Discovering Trends in Large Datasets Using Neural Networks

Khosrow Kaikhah, Ph.D. and Sandesh Doddameti
Department of Computer Science
Texas State University
San Marcos, Texas 78666

Abstract. A novel knowledge discovery technique using neural networks is presented. A neural network is trained to learn the correlations and relationships that exist in a dataset. The neural network is then pruned and modified to generalize the correlations and relationships. Finally, the neural network is used as a tool to discover all existing hidden trends in four different types of crimes (murder, rape, robbery, and auto theft) in US cities as well as to predict trends based on existing knowledge inherent in the network.

1 Introduction

Large datasets encompass hidden trends, which convey valuable knowledge about the dataset. The acquired knowledge is helpful in understanding the domain, which the data describe. The hidden trends, which can be expressed as rules or correlations, highlight the associations that exist in the data. Therefore, discovering these hidden trends, which are specific to the application, is extremely helpful and vital for analyzing the data.[1], [2]

We define a machine learning process that uses artificial neural networks to discover trends in large datasets. A neural network is trained to learn the inherent relationships among the data. The neural network is then modified via pruning and hidden layer activation clustering. The modified neural network is then used as a tool to extract common trends that exist in the dataset as well as to predict trends. The extraction phase can be regulated through several control parameters.

The extraction process defines a method for analyzing the knowledge acquired by the neural network that is encoded into its architecture, connections and weights. Our goal is to define an analytical approach using neural networks based on hidden unit activations and neural network training. The novelty of our process lies in relating the strength/predictability of ‘occurrence trends’ to the frequency of neural activations. The knowledge stored in the neural network is extracted only when the rigorous frequency and consistency requirements are satisfied thus providing a sound method for selecting the knowledge/rules from the vast possible combinations of data.

2 Rule Extraction Techniques

Andrews et al. in [3] discuss the difficulty in comprehending the internal process of how a neural network learns a hypothesis. According to their survey, rule extraction methods have been categorized into decompositional, pedagogical, and eclectic techniques. The distinguishing characteristic of the decompositional approach is that the focus is on extracting rules at the level of individual (hidden and output) units within the trained neural network. In the pedagogical approach to rule extraction, the trained neural network is treated as a black-box, in other words, the view of the underlying trained artificial neural network is opaque. The eclectic approach combines decompositional and pedagogical techniques. They describe several rule extraction methods and conclude that no single rule extraction/rule refinement technique is currently in a dominant position to the exclusion of all others.

The Subset method uses the decompositional technique to extract rules at each individual hidden and output unit.[4] These rules are then aggregated to form the composite rule base for the neural network as a whole. The basic idea is to search for subsets of incoming weights to each hidden and output unit which exceed the bias on the unit. The key underlying assumption is that the corresponding 'signal strength' associated with each connection to the unit is zero or one, in other words, 'maximally active' or 'minimally active'. This assumption is achieved by judicious selection of the unit's activation function. The Subset method is a 'general purpose' procedure that does not appear to be specific to a particular problem domain. However, since the solution time of the algorithm increases exponentially with the number of input units, it is suitable only for simple networks or so-called 'small' problem domains.

The M-of-N method uses the decompositional technique.[5] The M-of-N method is described as follows. For each hidden and output unit, identify groups of similarly-weighted links. Set link weights of all group members to the average of the group. Eliminate any groups that do not significantly affect whether the unit will be active or inactive. Holding all link weights constant, optimize biases of all hidden and output units using the back-propagation algorithm. Form a single rule for each hidden and output unit, the rule consists of a threshold given by the bias and weighted antecedents specified by the remaining links. Where possible, simplify rules to eliminate superfluous weights and thresholds. The M-of-N method is designed as ‘general purpose’ rule extraction method and not limited to any particular class of problem domain.

The RULEX method uses the decompositional technique.[6] It is designed to exploit the manner of construction and consequent behavior of a particular type of multilayer perceptron, the Constrained Error Back-Propagation (CEBP) MLP which is a type of local response neural network similar in performance to a Radial Basis Function (RBF) network. The hidden units of the CEBP network are sigmoid-based locally responsive units (LRUs) that have the effect of partitioning the training data into a set of disjoint regions, each region represented by a single hidden layer unit. Each LRU is composed of a set of ridges, one ridge for each dimension of the input. A ridge will produce an appreciable output only if the value presented as input lies within the active range of the ridge. The LRU output is the thresholded sum of the activations of the ridges. In order for a vector to be classified by an LRU, each component of the input vector must lie within the active region of its corresponding ridge. RULEX performs rule extraction by direct interpretation of the weight parameters as rules.

The VIA (Validity Interval Analysis) method uses the pedagogical technique to extract rules that map inputs directly to outputs.[7] VIA uses a generate-and-test procedure to extract symbolic rules from standard back-propagation trained neural network which have not been specifically constructed to facilitate rule extraction. The approach is similar to sensitivity analysis in that it characterizes the output of the trained ANN by systematic variations in the input patterns and examining the changes in the network classification. The validity interval of a unit specifies a maximum range for its activation. VIA is designed as a ‘general purpose’ rule extraction procedure and does not seem to be limited to any particular problem domain.

The BRAINNE system uses the pedagogical technique.[8] It extracts rules from an ANN trained using back-propagation. It requires a specialized training regime that tracks an initialized trained network with m inputs and n outputs and transforms it into a network with $m+n$ inputs and n outputs. This transformed network is then retrained. The next phase in the process is to perform a pair-wise comparison of the weights for the links between each of the original m input units and the set of hidden units with the weights from each of the n additional input units and the corresponding hidden units. The smaller the difference between the two values, the greater the contribution of the original input unit to the output. A major innovation of the BRAINNE system is the capability to deal with continuous data as input without first having to employ a discretising phase.

The Rule-Extraction-as-Learning method uses the eclectic technique.[9] The core idea is to view rule extraction as a learning task where the target concept is the function computed by the network and the input features are simply the network's input features. The key is the procedure used to determine if a given rule agrees with the network. This procedure accepts a class label *c* and a rule *r*, and returns 'true' if all instances covered by *r* are classified as members of class *c* by the network. The method is designed as a 'general purpose' rule extraction procedure and its applicability is not limited to any specific class of problem domains.

The DEDEC method uses the eclectic technique.[10] It provides a general method for disgorging the information contained in existing trained neural network solutions already implemented in various problem domains. It extracts symbolic rules efficiently from a set of individual cases. The task of rule extraction is treated as a process similar to that of identifying the minimal set of information required to distinguish a particular object from other objects. In order to search the solution space in as optimum a fashion as possible, the DEDEC method ranks the cases to be examined in order of importance. This is achieved by using the magnitude of the weight vectors in the trained ANN to rank the input units according to the relative share of their contribution to the output units. The focus is on extracting rules from those cases that involve what are deemed to be the most important input units. This method also employs heuristics to terminate the process either as soon as a measure of stability appears in the extracted rule set or the relative significance of an input unit, selected for generating cases to be examined, falls below some threshold value.

3 Related Research

Setiono in [11] uses the decompositional technique and applies feedforward multilayer neural networks to the data mining classification problem. The overall process is as follows: a) Train a neural network for a classification problem using the dataset. The network will be trained to the desired accuracy. b) The network is then pruned to obtain a minimal architecture to improve generalization and to decrease the complexity. c) The acquired knowledge is then extracted in the form of rules. An example which classifies persons based on their age and income is described in [11]. In addition, Setiono demonstrates the applicability and accuracy of the process on the Wisconsin breast cancer dataset in [12]. The rules extracted are in the if <conditions> then <result> form. The overall process is defined for classification problems with three layer networks (1 input, 1 hidden and 1 output layer). There are no significant control parameters for data analysis.

Gupta et al. in [13] propose an algorithm (GLARE) to extract classification rules from feedforward and fully connected neural networks trained by back-propagation. The major characteristics of the GLARE algorithm are: (a) its analytic approach for rule extraction, (b) its applicability to standard network structure and training method, and (c) its rule extraction mechanism as a direct mapping between input and output neurons. This method is designed for a neural network with only one hidden layer. This approach uses the significance of connection strengths based on their absolute magnitude and uses only a few important connections (highest absolute values) to analyze the rules.

Wang et al. in [14] present an example of biological data mining using neural networks. They train a Bayesian neural network with protein sequences to classify them as belonging to a particular super family. The input to the network is the protein sequence encoded in a special form and the output is a single neuron, which is activated if the protein sequence is in a particular class. Bayesian neural networks use probability to represent biases, sampling distributions, noise, and other uncertainties. The Bayesian approach incorporates external knowledge (biases) about the target function in the form of prior probabilities of different hypothesis functions. In this work, no rule extraction is performed and the process is a simple classification on the test data. The relevance of this article to our work lies in applicability of neural networks to learn the hypothesis encompassed in the dataset.

Our knowledge discovery process is both decompositional and pedagogical. It is decompositional in nature, since we examine the weights for pruning and clustering the hidden unit activation values. It is pedagogical, since we use the neural network as a black-box for knowledge discovery.[15] Our approach is neither limited by the complexity of the hidden layer, nor by the number of hidden layers. Therefore, our approach can be extended to networks with several hidden layers. Our process also provides control parameters for data analysis. These parameters provide a mechanism to control the probability of occurrences and accuracy of rules which are similar to ‘Support and Confidence’ framework of the associative data mining. However, there are no data mining or statistical techniques that are comparable to our knowledge discovery process.

4 Discovering Trends

We have developed a novel process for discovering trends in datasets, with *m* dimensional input space and *n* dimensional output space, utilizing neural networks. Our process is independent of the application. The significance of our approach lies in using neural networks for discovering knowledge, with control parameters. The control parameters influence the discovery process in terms of importance and significance of the acquired knowledge. There are four phases in our approach: 1) neural network training and filtering, 2) pruning and re-training, 3) clustering the hidden neuron activation values, and 4) rule discovery and extraction.

In phase one, the neural network is trained using a supervised learning method. The neural network learns the associations inherent in the dataset. In phase two, the neural network is pruned by removing all unnecessary connections and neurons. In phase three, the activation values of the hidden layer neurons are clustered using an adaptable clustering technique. In phase four, the modified neural network is used as a tool to extract and discover hidden trends. These four phases are described in more detail in the next four sections.

4.1 Neural Network Training and Filtering

Neural networks are able to solve highly complex problems due to the non-linear processing capabilities of their neurons. In addition, the inherent modularity of the neural network’s structure makes them adaptable to a wide range of applications.[16] The neural network adjusts its parameters to accurately model the distribution of the provided dataset. Therefore, exploring the use of neural networks for discovering correlations and trends in data is prudent.

Neural networks learn to approximate complex functional relationships between input and output patterns. An example of a mapping problem space is depicted in Figure 1.

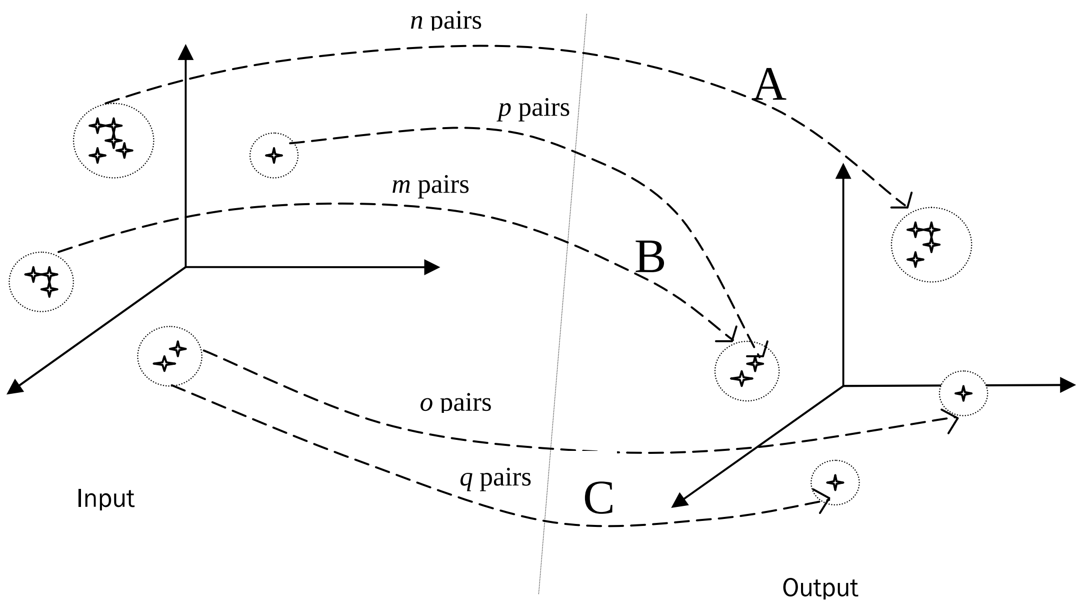


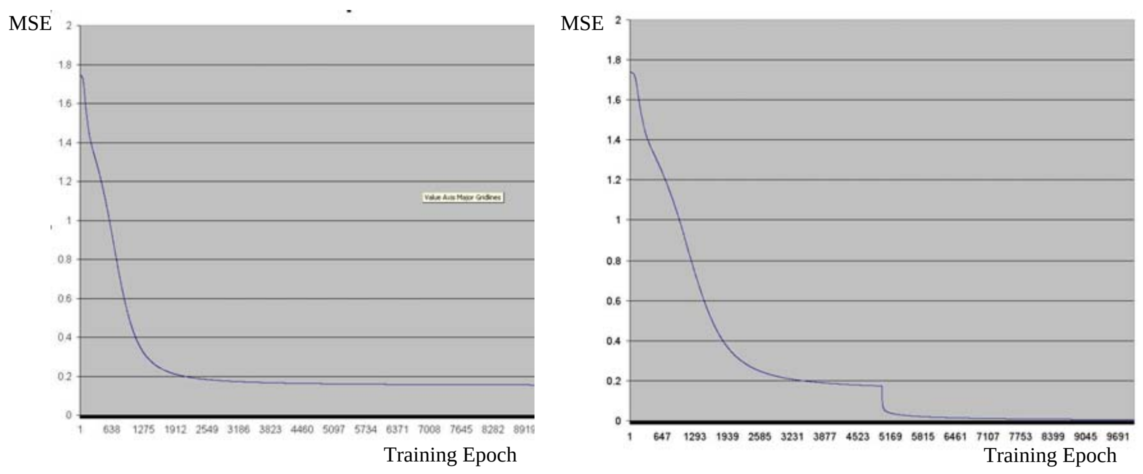
Figure 1: Types of mapping that exist in a given dataset

In any dataset, there are three types of mappings. Type A mapping is from similar input patterns to similar output patterns. They are said to have consistent mapping, i.e. close input pairs (closely grouped in input space) map to close output pairs (closely grouped in output space). Type B mapping is from different regions in the input space to similar regions in the output space. Type C mapping is from one region in the input space to various regions in the output space. Type C is inconsistent mapping, since similar inputs are mapped to different outputs. In the example above, there are *n* pairs of type A mapping, *p* + *m* pairs of type B mapping, and *o* + *q* pairs of type C mappings.

Consistent and frequent patterns of type A and B strengthen the learning. The frequencies of these patterns largely affect the learning and generalization capability of the neural network.

However, inconsistent patterns of type C weaken the learning. In these cases, the inconsistent pairs with the lowest frequencies should be filtered out in favor of the higher frequency pairs. The inconsistency of a particular pattern can be measured in terms of its error with respect to the mean squared error. For patterns in Figure 1, the filtering process removes either set *o* or set *q* patterns, whichever that has a lower frequency, or both sets, if both sets have relatively low frequencies with respect to the whole dataset.

By filtering the scattered and inconsistent data during the training phase, the neural network can achieve a higher performance rate. Figure 2 depicts the error rates obtained for a sample dataset with and without filtering.



Mean Squared Error for a Sample Set s
Without Filtering

Mean Squared Error for a Sample Set s
With Filtering

Figure 2: Effects of Filtering

The input and output patterns may be real-valued or binary-valued. If the patterns are real-valued, each value is discretized and represented as a sequence of binary values, where each binary value represents a range of real values. For example, in a credit card transaction application, an attribute may represent the person's age (a value greater than 21). This value can be discretized into 4 different intervals: (21-30],(30-45],(45-65], and (65+]. Therefore [0 1 0 0] would represent a customer between the ages of 31 and 45. The number of neurons in the input and output layers are determined by the application, while the number of neurons in the hidden layer are dependent on the number of neurons in the input and output layers.

We use an augmented gradient descent approach to train and update the connection strengths of the neural network. The gradient descent approach is an intelligent search for the global minima of the energy function. We use an energy function, which is a combination of an error function and a penalty function [5]. The total energy function to be minimized during the training process is:

$$\theta(w,v) = E(w,v) + P(w,v) \quad (1)$$

The error function to be minimized is the mean squared error.

$$E(w,v) = \frac{1}{L}\sum_{l=i}^{L}\sum_{k=1}^{n}(o_{lk} - d_{lk})^2 \tag{2}$$

where L is the number of patterns
n is the number of output neurons
o_{lk} is the output of the k^{th} output neuron for pattern l
d_{lk} is the expected output of the k^{th} output neuron for pattern l

The addition of the penalty function drives the associated weights of unnecessary connections to very small values while strengthening the rest of the connections. Therefore, the unnecessary connections and neurons can be pruned without affecting the performance of the network. The penalty function is defined as:

$$P(w,v) = \rho_{decay}\left(P_1(w,v) + P_2(w,v)\right) \tag{3}$$

$$P_1(w,v) = \varepsilon_1\left(\sum_{j=1}^{h}\sum_{i=1}^{m}\frac{\beta w_{ij}^2}{1+\beta w_{ij}^2} + \sum_{j=1}^{h}\sum_{k=1}^{n}\frac{\beta v_{jk}^2}{1+\beta v_{jk}^2}\right)$$

$$P_2(w,v) = \varepsilon_2\left(\sum_{j=1}^{h}\sum_{i=1}^{m}w_{ij}^2 + \sum_{j=1}^{h}\sum_{k=1}^{n}v_{jk}^2\right)$$

m is the number of input neurons
h is the number of hidden neurons
n is the number of output neurons
w_{ij} is the i^{th} input to j^{th} hidden layer connection strength
v_{jk} is the j^{th} hidden to k^{th} output layer connection strength
ε_1 is scaling factor typically set at 0.1
ε_2 is the scaling factor typically set at 0.00001
β is the scaling factor typically set at 10
ρ_{decay} is the scaling factor typically set to a value between 0.03 and 0.05

The network is trained till it reaches a recall accuracy of 99% or higher.

4.2 Pruning and Re-Training

The neural network is trained with an energy function, which includes a penalty function. The penalty function drives the strengths of unnecessary connections to approach zero very quickly. The insignificant connections having very small values can safely be removed without considerable impact on the performance of the network.

For each input to hidden layer connection (w_{ij}), if $\max_k \left| v_{jk} w_{ij} \right| < 0.1$ remove w_{ij}, and for each hidden to output layer connection (v_{jk}), if $\left| v_{jk} \right| \leq 0.1$ remove v_{jk}.

After removing all weak connections, any input layer neuron having no outgoing connections can be removed. In addition, any hidden layer neuron having no incoming or outgoing connections can safely be removed. Finally, any output layer neuron having no incoming connections can be removed. Removal of input layer neurons corresponds to having irrelevant inputs in the data model; removal of hidden layer neurons reduces the complexity of the network and the clustering phase; and removal of the output layer neurons corresponds to having irrelevant outputs in the data model. Pruning the neural network results in a less complex network while improving its generalization.

Once the pruning step is complete, the network is trained with the same dataset in phase one to ensure that the recall accuracy of the network has not diminished significantly. If the recall accuracy of the network drops by more than 2%, the pruned connections and neurons are restored and a stepwise approach is pursued. In the stepwise pruning approach, the insignificant incoming and outgoing connections of the hidden layer neurons are pruned, one neuron at a time, and the network is re-trained and tested for recall accuracy.

4.3 Clustering the Hidden Layer Neuron Activation Values

The activation values of each hidden layer neuron are dynamically clustered and re-clustered with a cluster radius and confidence radius, respectively. The clustering algorithm is adaptable, that is, the clusters are created dynamically as activation values are added into the cluster-space. Therefore, the number of clusters and the number of activation values in each cluster are not known *a priori*. The centroid of each cluster represents the mean of the activation values in the cluster and can be used as the representative value of the cluster, while the frequency of each cluster represents the number of activation values in that cluster. The centroid and frequency of a cluster c are denoted by G_c and $freq_c$ respectively. By using the centroids of the clusters, each hidden layer neuron has a minimal set of activations. This helps with getting generalized outputs at the output layer. The centroid is adjusted dynamically as new elements e_c^i are added to the cluster.

$$G_c^{new} = \frac{(G_c^{old} \cdot freq_c) + e_c^i}{freq_c + 1} \tag{4}$$

$Dist\left(G_c, e\right)$ is the numerical distance of an element e from the centroid G_c.

$$Dist\left(G_c, e\right) = \left|G_c - e\right| \tag{5}$$

The radius of a cluster defines the distance of the farthest element to the centroid.

$$\left|G_c - e_c^i\right| \le r_c \quad \text{for any cluster } c \tag{6}$$

Cluster c is a region having a radius of r_c which includes elements e_c^i, where $1 \le i \le n_c$ and n_c is the number of elements in the cluster. Clusters may be overlapping or disjoint. The cluster radius must be less than a predetermined upper bound in order to maintain the desired accuracy of the network. The upper bound for the cluster radius defines a range for which the hidden layer neuron activation values can fluctuate without compromising the network performance. The actual activation value of the jth hidden layer neuron is defined as:

$$S_j = Sig\left(\sum_{l=1}^{n} x_l \cdot w_{lj}\right) \tag{7}$$

where $Sig(x) = \dfrac{1}{1 + e^{-a \cdot x}}$

Each group of activation values are represented by a centroid value which in the worse case is defined as:

$$G_{c_j} = S_j \pm r_{c_j} \tag{8}$$

The output of the kth output layer neuron is defined as:

$$Z_k = Sig\left(\sum_{j=1}^{m} G_{c_j} \cdot v_{jk}\right) \tag{9}$$

For maintaining the accuracy of the network, the following must hold.

$$\left|Z_k - Z_k^*\right| \leq \rho \tag{10}$$

Z_k^* is the desired value, and ρ (the tolerance factor) is typically set to a small value such as 0.01. The upper bound for the cluster radius is derived from (8), (9), and (10) as:

$$\left|r_c\right| \leq \frac{\ln\left(\frac{1}{\rho} - 1\right)}{\max_k \left|\sum_{j=1}^{m} v_{jk}\right|} \tag{11}$$

Since dynamic clustering is order sensitive, once the clusters are dynamically created with a cluster radius, all elements will be re-clustered with a confidence radius of one-half the cluster radius.

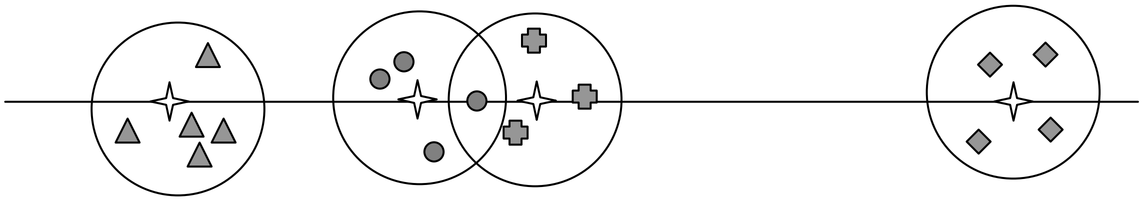


Figure 3: Effects of Clustering

The benefits of re-clustering are twofold: 1) Due to order sensitivity of dynamic clustering, some of the activation values may be misclassified. Re-clustering alleviates this deficiency by classifying the activation values in appropriate clusters. 2) Re-clustering with confidence radius (one-half the cluster radius) eliminates any possible overlaps among clusters. For example, an element which was determined to belong to a cluster in Figure 3, after re-clustering, is determined to belong to a different cluster in Figure 4. In addition during re-clustering, the frequency of each confidence cluster is calculated, which will be utilized in the extraction phase.

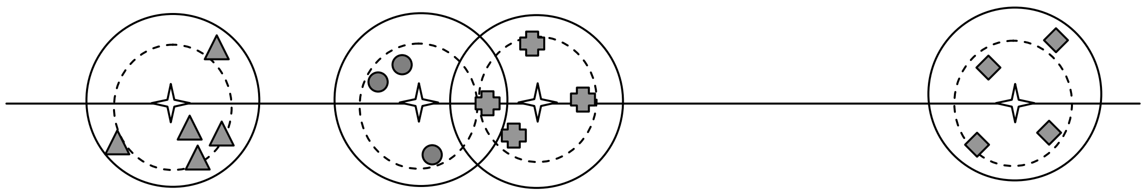


Figure 4: Effects of Re-Clustering

Clustering of hidden layer neuron activation values helps to identify regions of activations along with the frequency of such activities. In addition, clustering generates representative values for such regions by which we can retrieve generalized outputs. Since we utilize a desired confidence frequency, we can examine the level of activities in all regions across the entire hidden layer. Only patterns which satisfy the desired confidence frequency across the entire hidden layer are considered. This ensures that inconsistent patterns and those which fall within the regions with low level of activity are not considered.

4.4 Rule Discovery and Extraction

In the final phase of the process, the knowledge acquired by the trained and modified neural network is extracted in the form of rules. This is done by utilizing the generalization of the hidden layer neuron activation values as well as control parameters. The novelty of the extraction process is the use of the hidden layer as a filter by performing vigilant tests on the clusters. Clusters identify common regions of activations along with the frequency of such activities. In addition, clusters provide representative values (the mean of the clusters) that can be used to retrieve generalized outputs.

The control parameters for the extraction process include: a) cluster radius, b) confidence frequency, and c) hidden layer activation level. The cluster radius determines the coarseness of the clusters. The confidence radius, r_{conf} , is usually set to one-half of the cluster radius to remove any possible overlaps or misclassifications among clusters. The confidence frequency, $Freq_{Conf}$, defines the minimum required support, which reflects the required commonality among patterns. The hidden layer activation level defines the maximum level of tolerance for inactive hidden layer neurons.

Knowledge extraction is performed in two steps. First, the existing trends are discovered by presenting the input patterns in the dataset to the trained and modified neural network and by providing the desired control parameters.

For a given hidden layer neuron activation value e_j:

If $\min_n Dist(G_c, e_j) \le r_{conf}$ and $G_{freq} \ge Freq_{Conf}$

the hidden layer activation value would be G_c

Otherwise

the hidden layer neuron would be inactive

If the total number of inactive hidden layer neurons does not exceed the maximum allowable tolerance for inactive hidden layer neurons, the network produces an output.

The input patterns that satisfy the rigorous extraction phase requirements and produce an output pattern represent generalization and correlations that exist in the dataset. The level of generalization and correlation acceptance is regulated by the control parameters. This ensures that inconsistent patterns, which fall outside confidence regions of hidden layer activations, or fall within regions with low levels of activity, are not considered. There may be many duplicates in these accepted input-output pairs. In addition, several input-output pairs may have the same input pattern or the same output pattern. Those pairs having the same input patterns will be combined, and, those pairs having the same output patterns will be combined. This post-processing is necessary to determine the minimal set of trends. Any input or output attribute not included in the discovered trend corresponds to irrelevant attributes in the dataset.

Second, the predicated trends are extracted by providing all possible permutations of input patterns, as well as the desired control parameters. Any additional trends discovered in this step constitute the predicated knowledge based on existing knowledge. This step is a direct byproduct of the generalizability of neural networks.

5 Discovering Trends in Crimes in US Cities

We compiled two datasets consisting of the latest two annual demographic and crime statistics for 6100 US cities. The data are derived from three different sources: 1) US Census; 2) Uniform Crime Reports (UCR) published annually by the Federal Bureau of

Investigation (FBI); and 3) Unemployment Information from the Bureau of Labor Statistics.

We used the first dataset to discover existing and predicted trends in crimes with respect to the demographic characteristics of the cities and used the second dataset to verify the accuracy of the predicted trends. We divided the datasets into three groups in terms of the population of cities: a) cities with populations of less than 20k (4706 cities), b) cities with populations of greater than 20k and less than 100k (1193 cities), and c) cities with populations of greater than 100k (201 cities). We divided the datasets into three groups in terms of city population, since otherwise, small cities (cities less that 20k) would dominate the process due to their high overall percentage. We then trained a neural network for each group of cities and each of four types of crimes (murder, rape, robbery, and auto theft) using the first dataset. Table 1 includes the demographic characteristics and crime types we used for the knowledge discovery process.

Table 1: The Categories for the Process

POP :	*City Population*
SINGP% :	*Percentage of Single-Parent Households*
MINOR% :	*Percentage of Minority*
YOUNG% :	*Percentage of Young People(between the ages of 15 and 24)*
HOMEOW% :	*Percentage of Home Owners*
SAMEHO% :	*Percentage of People living in the Same House for Over 5 years*
UNEMPL% :	*Percentage of Unemployment*
Murder:	*Number of Murders*
Rape:	*Number of Rapes*
Robbery:	*Number of Robberies*
Auto-Theft:	*Number of Auto Thefts*

Each category is discretized into several intervals to define the binary input/output patterns. The size of each interval is data dependent. The data should be discretized as to preserve the integrity of the data. Therefore, the regions with high data density should be discretized into smaller intervals to characterize the data more accurately. In general, the granularity of data analysis increases as the size of intervals decreases. For each category, we discretized the regions having high data density into smaller intervals to preserve data integrity. Table 2 represents the discrete intervals for each category. Each interval of each category is represented by a single neuron.

Table 2: Discrete Intervals

Categories	Neurons	Intervals
POP(*small*)	5	[0-4k],(4k-8k],(8k,12k],(12k-16k],(16k-20]
POP(*medium*)	5	(20k-40k],(40k-60k],(60k-80k],(80k-90k],(90k-100k]
POP (*large*)	5	(100k-130k],(130k-160k],(160k-200k],(200k-500k],500k+
SINGP%	7	[0-5],(5-7],(7-9],(9-11],(11-14],(14-20],(20-100]
MINOR%	6	[0-5],(5-10],(10-20],(20-40],(40-70],(70-100]
YOUNG%	7	[0-12],(12-13],(13-14],(14-15],(15-17],(17-25],(25-100]
HOMEOW%	7	[0-40],(40-50],(50-60],(60-70],(70-80],(80-90],(90-100]
SAMEHO%	6	[0-45],(45-50],(50-55],(55-60],(60-65],(65-100]
UNEMPL%	6	[0-4],(4-6],(6-8],(8-12],(12-20],(20-100]
Murder	4	0, (1-5],(5-10],10+
Rape	5	0, (1-5],(5-10],(10-70],70+
Robbery	5	0, (1-5],(5-10],(10-100],100+
Auto-Theft	5	[0-10],(10-100],(100-500],(500-1000],1000+

5.1 Neural Network Architecture

For each crime type, three different feedforward neural networks with a single hidden layer are trained, using the modified back-propagation algorithm, for the three groups of cities (small, medium, and large) to an accuracy of 99% or higher. There are twelve neural networks in total. Each network consists of 44 input layer neurons, 60 hidden layer neurons, and 4 or 5 output layer neurons. The number of hidden layer neurons is arbitrary. We experimented with a wide range of values and chose the value that resulted in best network recall performance. After the training phase, the networks are pruned and clustered. Although, for each network, about 30% of the connections as well as about 5% of the hidden layer neurons were pruned, none of the input neurons were pruned. This reflects the importance of all demographic categories we used for discovering trends in crimes. After phase two and three, all twelve networks maintain an accuracy rate of 99% or higher. The networks were then used as tools to discover the existing, as well as predicted trends.

5.2 Discovering Existing Trends

The existing trends are those that are present in the existing dataset. Therefore, all input patterns in the first dataset are used as input to the trained and modified networks to extract existing trends. For each type of crime, several trends were discovered. Some of the discovered trends are represented in the following three tables. For each type of crime, either a lower-bound or an upper-bound percentage of population is calculated. The support percentage for each discovered trend represents the percentage of patterns in the first dataset that are similar to the trend. This is used for verification of the discovered trends and reflects the strength of each trend.

The dataset for small cities consists of 4706 records. Table 3 includes some of the existing trends discovered for small cities. For example, cities having population between 4000 and 8000, minority between 0% and 10%, unemployment less than 6%, single-parent households less than 9%, people living in the same house for more than five years less than 55%, young people between 12% and 13%, and home owners between 60% and 70%, have an annual robbery rate of 1 to 5. This trend personifies 30% of the patterns in the first dataset.

Table 3: The Existing Trends for Small Cities

Support%	30%	30%	30%	30%
POP	0-4k	0-4k	4k-8k	4k-12k
MINOR %	0-5	20-40	0-10	0-5
UNEMPL %	0-4	8-12	0-6	0-4
SINGP %	7-9	7-9	0-9	7-11
SAMEHO %	50-55	60-65	0-55	55-60
YOUNG %	14-15	0-12	12-13	13-14
HOMEOW %	70-80	70-80	60-70	60-70
	Murder	Rape	Robbery	Auto-Theft
	0	0	1-5	1-5
Population%	0%	0%	≤0.00125%	≤0.00125%

The dataset for medium cities consists of 1193 records. Table 4 includes some of the existing trends discovered for medium cities. For example, cities having population between 20000 and 40000, minority less than 5%, unemployment of 4% to 6%, single-parent households less than 5%, people living in the same house for more than five years between 65% and 100%, young people between 12% and 13%, and home owners between 80% and 90%, have an annual auto theft rate of 10 to 100. This trend personifies 30% of the patterns in the first dataset.

Table 4: The Existing Trends for Medium Cities

Support%		30%	25%	25%	30%
POP		20k-40k	20k-40k	20k-40k	20k-40k
MINOR %		0-5	5-10	5-10	0-5
UNEMPL %		0-6	4-6	0-4	4-6
SINGP %		9-11	11-14	7-9	0-5
SAMEHO %		45-50	45-50	0-45	65-100
YOUNG %		12-13	14-15	17-25	12-13
HOMEOW %		60-90	40-50	60-70	80-90
		Murder	**Rape**	**Robbery**	**Auto-Theft**
Population%		**0-5 ≤0.00025%**	**1-10 ≤0.0005%**	**5-10 ≤0.0005%**	**10-100 ≤0.005%**

The dataset for large cities consists of 201 records. Table 5 includes some of the existing trends discovered for large cities. For example, cities having population between 200000 and 500000, minority between 20% and 40%, unemployment of 8% to 12%, single-parent households between 11% and 14%, people living in the same house for more than five years between 50% and 55%, young people between 15% to 17%, and home owners between 50% and 60%, have an annual murder rate of 5 to 10. This trend personifies 25% of the patterns in the first dataset.

Table 5: The Existing Trends for Large Cities

Support%		25%	25%	30%	20%
POP		200k-500k	200k-500k	130k-200k	100k-130k
MINOR %		20-40	40-70	20-70	10-20
UNEMPL %		8-12	8-12	6-8	0-4
SINGP %		11-14	14-20	11-14	5-7
SAMEHO %		50-55	50-55	45-50	0-45
YOUNG %		15-17	15-17	15-25	0-12
HOMEOW %		50-60	50-60	50-60	70-80
		Murder	**Rape**	**Robbery**	**Auto-Theft**
Population%		**5-10 ≤0.00005%**	**10-70 ≤0.00035%**	**100+ ≥0.000769%**	**500-1000 ≤0.01%**

5.3 Discovering Predicted Trends

The predicted trends are those that reflect the generalization property of the trained neural networks. Therefore, all possible permutations of input categories are used as input to the trained and modified networks to extract predicted trends based on existing correlations. Since all possible permutations of input categories include the existing dataset, naturally the discovered trends in this step include the existing trends that were discovered in the previous step. After removing the already discovered trends, the remaining trends constitute the predicted trends. The following three tables include predicted trends for three types of cities (small, medium, and large). The predicted trends describe cities having certain demographic characteristics and their crime rates. These cities did not exist in the first dataset, however if such cities were to exist, they can expect to have such crime rates. We used the second dataset to verify the validity of the predicted trends. The predicted trends are reasonable expectations that result from the generalizability property of trained neural networks. The consistency percentage indicates the percentage of patterns in the second dataset (future data) that are similar to the predicted trend. A predicted trend will be consistent if all necessary conditions of the trend become available. On the other hand, if the consistency of a predicted trend is 0%, this indicates that the necessary conditions required for this trend are not currently present. However, it is still possible for this predicted trend to be consistent if required conditions become available. Therefore, the predicted trends can be used for monitoring the environment.

Table 6 includes some of the predicted trends discovered for small cities. For example, if cities having population less than 4000, minority between 10% and 20%, unemployment of 4% to 6%, single-parent households of 9% to 11%, people living in the same house for more than five years between 65% to 100%, young people between 17% and 25%, and home owners between 40% and 50% were to exist, they would have an annual robbery rate of 1 to 5. This prediction is consistent with 5% of the patterns in the second dataset.

Table 6: The Predicted Trends for Small Cities

Consistency%	30%	10%	5%	0%
POP	0-4k	4k-8k	0-4k	4k-8k
MINOR %	0-5	5-10	10-20	5-10
UNEMPL %	4-6	4-6	4-6	20-40
SINGP %	5-7	5-7	9-11	7-9
SAMEHO %	45-50	60-65	65-100	60-65
YOUNG %	25-40	15-17	17-25	17-25
HOMEOW %	60-70	40-50	40-50	60-70
	Murder	Rape	Robbery	Auto-Theft
	0	1-5	1-5	1-5
Population%	0%	≤0.00125%	≤0.005%	≤0.00125%

Table 7 includes some of the predicted trends discovered for medium cities. For example, if cities having population between 20000 and 40000, minority between 70% and 100%, unemployment of 6% to 8%, single-parent households between 5% and 7%, people living in the same house for more than five years between 50% and 55%, young people between 13% and 14%, and home owners between 40% and 50% were to exist, they would have an annual rape rate of 1 to 10. This prediction is consistent with 12% of the patterns in the second dataset. It is worth noting that, for medium cities, no trend was predicted for auto theft.

Table 7: The Predicted Trends for Medium Cities

Consistency%	25%	12%	0%	
POP	20k-40k	20k-40k	20k-40k	
MINOR %	0-5	70-100	5-10	
UNEMPL %	0-6	6-8	8-12	
SINGP %	5-7	5-7	11-14	
SAMEHO %	50-60	50-55	50-55	
YOUNG %	12-13	13-14	14-15	
HOMEOW %	80-90	40-50	40-50	
	Murder	Rape	Robbery	Auto-Theft
	0	1-10	10-100	No strong predictor
Population%	0%	≤0.0005%	≤0.005%	

Table 8 includes some of the predicted trends discovered for large cities. For example, if cities having population between 200000 and 500000, minority between 40% and 100%, unemployment of 8% to 12%, single-parent households between 11% and 14%, people living in the same house for more than five years between 50% and 55%, young people less than 12%, and home owners between 60% and 70% were to exist, they would have an annual auto theft rate of more than 1000. This prediction is consistent with 20% of the patterns in the second dataset.

Table 8: The Predicted Trends for Large Cities

Consistency%	**11%**	**8%**	**10%**	**20%**
POP	200k-500k	200k-500k	200k-500k	200k-500k
MINOR %	20-40	70-100	40-70	40-100
UNEMPL %	8-12	20-40	6-8	8-12
SINGP %	11-14	11-14	11-14	11-14
SAMEHO %	50-55	50-55	45-50	50-55
YOUNG %	15-17	13-14	17-25	0-12
HOMEOW %	50-60	60-70	50-60	60-70
	Murder	**Rape**	**Robbery**	**Auto-Theft**
	5-10	**70+**	**100+**	**1000+**
Population%	**≤0.00005%**	**≥0.00035%**	**≥0.0005%**	**≥0.005%**

6 Conclusions

For each group of cities (small, medium, large), we were able to discover the existing trends for each type of crime (murder, rape, robbery, auto theft). These trends represent the hidden knowledge and are based on the high level of commonalty inherent in the dataset. The desired level of commonalty can be regulated through the control parameters. We were able to demonstrate the validity of existing trends by determining the percentage of patterns in the first dataset that support each trend. In addition, by using the generalizability feature of neural networks, we were able to discover predicted trends. These predicted trends describe the demographic characteristics and crime rates of cities that do not exist in the dataset but are possible to exist. Once again, we were able to verify the validity of the predicted trends by determining the percentage of patterns in the second dataset (future data) that is consistent with each predicted trend.

The knowledge discovery technique offers two unique features that are not available in other knowledge discovery techniques. First, the control parameters provide a means to set the desired level of confidence for extracting existing and predicted trends. Second, the predicted trends provide reasonable expectations that can be used for monitoring the environment. The knowledge discovery technique can be applied to any application domain that deals with vast amounts of data such as medical, military, business, and security. In medical fields, the data gathered from cancer patients can be used to discover the dominating factors and trends for the development of cancer. In military fields, the data gathered from the enemy can be used to predicate their future movements. In business environments, the data gathered from customers can be used to model the transaction activities of the customers. In security applications, the data gathered can be used to predicate and prevent potential intrusions.

7 References

[1] Joseph P. Bigus, *Data Mining With Neural Networks: Solving Business Problems from Application Development to Decision Support*, McGraw-Hill, NY, 1996

[2] Jiawei Han and Micheline Kamber, *Data Mining: Concepts and Techniques,* Morgan Kaufmann, San Francisco, California, 2001

[3] Robert Andrews, Joachim Diederich, and Alan Tickle, "A Survey and Critique of Techniques for Extracting Rules from Trained Artificial Neural Networks", *Knowledge-Based Systems,* Vol.8, No.6, pp. 373-389, 1995

[4] L. Bochereau and P. Bourgine, "Extraction of semantic features and logical rules from a multilayer neural network" *International Joint Conference on Neural Networks Washington DC* Vol. 2, pp. 579-582, 1990

[5] G. Towell and J. Shavlik, "The Extraction of Refined Rules From Knowledge Based Neural Networks" *Machine Learning* Vol. 131, pp. 71-101, 1993

[6] R. Andrews and S. Geva, "Rule extraction from a constrained error back propagation MLP" *Proc. 5th Australian Conference on Neural Networks Brisbane Queensland*, pp. 9-12, 1994

[7] S.B. Thrun, "Extracting Provably Correct Rules From Artificial Neural Networks" *Technical Report IAI-TR-93-5 Institut fur Informatik III Universitat Bonn*, 1994

[8] S. Sestito and T. Dillon, “Automated knowledge acquisition of rules with continuously valued attributes” *Proc. 12th International Conference on Expert Systems and their Applications (AVIGNON’92)*, pp 645-656, May 1992

[9] M. W. Carven and J.W. Shavlik, “Using sampling and queries to extract rules from trained neural networks” *Machine Learning: Proceedings of the Eleventh International Conference*, pp. 176-183, 1994

[10] A.B. Tickle, M. Orlowski, and J. Diederich, “DEDEC: decision detection by rule extraction from neural networks” *QUT NRC*, September 1994

[11] Rudy Setiono, H. Lu, H. Liu, “Effective data mining using neural networks”, *IEEE Transactions on Knowledge and Data Engineering,* Vol. 8, No. 6, pp. 957-961, December 1996

[12] Rudy Setiono, “Extracting Rules from Pruned Neural Networks for Breast Cancer Diagnosis”, *Artificial Intelligence in Medicine*, Vol. 8, No. 1, pp. 37-51, February 1996

[13] Amit Gupta, Sang Park, and Siuva M. Lam, “Generalized Analytic Rule Extraction for Feedforward Neural Networks”, *IEEE transactions on knowledge and data engineering*, vol. 11, pp. 985–991, 1998

[14] Jason T. L. Wang, Qicheng Ma, Dennis Shasha, and Cathy H.Wu, “Application of Neural Networks to Biological Data Mining: A Case Study in Protein Sequence Classification”, *The Sixth ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 305-309, August 20-23, 2000 Boston, MA, USA

[15] Khosrow Kaikhah and Sandesh Doddameti, “Knowledge Discovery Using Neural Networks” *Seventeenth International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems, (IEA/AIE)*, pp. 20-28, May 2004

[16] Kishan Mehrotra, Chilukuri K. Mohan, and Sanjay Ranka, *Elements of Artificial Neural Networks (Complex Adaptive Systems)*, Cambridge, MA: MIT Press, 1997