Data Preparation Process for Construction Knowledge Generation through Knowledge Discovery in Databases

Lucio Soibelman, M.ASCE,[1] and Hyunjoo Kim[2]

Abstract: As the construction industry is adapting to new computer technologies in terms of hardware and software, computerized construction data are becoming increasingly available. The explosive growth of many business, government, and scientific databases has begun to far outpace our ability to interpret and digest the data. Such volumes of data clearly overwhelm the traditional methods of data analysis such as spreadsheets and ad-hoc queries. The traditional methods can create informative reports from data, but cannot analyze the contents of those reports. A significant need exists for a new generation of techniques and tools with the ability to automatically assist humans in analyzing the mountains of data for useful knowledge. Knowledge discovery in databases (KDD) and data mining (DM) are tools that allow identification of valid, useful, and previously unknown patterns so that the construction manager may analyze the large amount of construction project data. These technologies combine techniques from machine learning, artificial intelligence, pattern recognition, statistics, databases, and visualization to automatically extract concepts, interrelationships, and patterns of interest from large databases. This paper presents the necessary steps such as (1) identification of problems, (2) data preparation, (3) data mining, (4) data analysis, and (5) refinement process required for the implementation of KDD. In order to test the feasibility of the proposed approach, a prototype of the KDD system was developed and tested with a construction management database, RMS (Resident Management System), provided by the U. S. Corps of Engineers. In this paper, the KDD process was applied to identify the cause(s) of construction activity delays. However, its possible applications can be extended to identify cause(s) of cost overrun and quality control/assurance among other construction problems. Predictable patterns may be revealed in construction data that were previously thought to be chaotic.

DOI: 10.1061/(ASCE)0887-3801(2002)16:1(39)

CE Database keywords: Data processing; Databases; Neural networks; Construction industry; Data analysis.

Introduction

Currently, the construction industry is experiencing explosive growth in its capability to both generate and collect data. Advances in scientific data collection, the new generation of sensor systems, the introduction of bar codes for almost all commercial products, and computerization have generated a flood of data. Advances in data storage technology, such as faster, higher capacity, and less expensive storage devices (e.g., magnetic disks, CD-ROMS), better database management systems, and data warehousing technology have allowed the transformation of this enormous amount of data into computerized database systems. As the construction industry is adapting to new computer technologies in terms of hardware and software, computerized construction data are becoming more and more available. However, in most cases, these data may not be used, or even properly stored. Several reasons exist: (1) construction managers do not have sufficient time to analyze the computerized data; (2) complexity of the data analysis process is sometimes beyond the simple application; and (3) up to now, there was no well defined automated mechanism to extract, preprocess, and analyze the data and summarize the results so that the site managers could use it. On the other hand, it is obvious that valuable information can be obtained from an appropriate use of these data.

A knowledge discovery application that discovers valuable patterns on construction costs or activity durations from construction project data can be very beneficial. One discovered pattern might be ''the activity that has a pattern of 50% probability of delay.'' Patterns might also be much more complex, taking into account several variables and specifying the conditional probability of delay for a large and possibly exhaustive set of possible activity profiles. A plan of action might be to apply the pattern to the current activities and determine which are most likely to be delayed and then establish way(s) to avoid those delays. We can estimate the payoff of such a strategy by assuming an effectiveness rate. If the estimated payoff from the strategy was sufficiently high, a real program to avoid delays would be implemented, and its results would be measured.

In this paper, specific issues to consider during the knowledge discovery in databases (KDD) process are presented because the complexity of the KDD application (limited breadth or coverage, data outliers, diverse forms of data, high dimensionality, etc.) makes development of an appropriate KDD difficult. To test the feasibility of the proposed approach, a prototype of the KDD system was developed and tested with the database RMS (Resident Management System) provided by the U.S. Army Corps of Engineers (USACE). Obviously, true knowledge cannot be obtained from a database if data are collected inconsistently. RMS is an automated construction management/quality information sys-

[1] Assistant Professor, Dept. of Civil Engineering, Univ. of Illinois at Urbana-Champaign, Urbana, IL 61801.

[2] Doctoral Candidate, Dept. of Civil Engineering, Univ. of Illinois at Urbana-Champaign, Urbana, IL 61801.

Note. Discussion open until June 1, 2002. Separate discussions must be submitted for individual papers. To extend the closing date by one month, a written request must be filed with the ASCE Managing Editor. The manuscript for this paper was submitted for review and possible publication on November 30, 2000; approved on July 17, 2001. This paper is part of the ***Journal of Computing in Civil Engineering***, Vol. 16, No. 1, January 1, 2002. ©ASCE, ISSN 0887-3801/2002/1-39–48/$8.00+$.50 per page.

tem that is PC-based, LAN-compatible, and oriented to the daily requirements of USACE. The RMS database is considered to be consistent in that it stores data on contract administration, quality assurance/quality control, schedule, and cost from project planning and design to the completion of a construction project. RMS also has fully automated single-entry data exchange/communication capabilities with Corps-wide systems. Currently, RMS stores construction project data that includes construction project planning, contract administration, quality assurance, payments, correspondence, submittal management, safety and accident administration, modification processing, and management reporting.

KDD Implementation for Pattern Discovery

Historically, the notion of finding useful patterns in raw data has been given various names, including knowledge extraction, information discovery, information harvesting, data archeology, and data pattern processing (Fayyad et al. 1996). By the end of the 1980s, a new term, KDD, was coined to replace all of the old terms referring to methods of finding patterns and similarities in raw data. Artificial intelligence and machine learning practitioners quickly adopted KDD and used it to cover the overall process of extracting knowledge from databases. KDD can be considered an interdisciplinary field involving concepts from machine learning, database technology, statistics, mathematics, high-performance computing, and visualization. Although the main concern of database technologists has been to find efficient ways of storing, retrieving, and manipulating data, the machine learning and statistical community has been focused on developing techniques for learning knowledge from data. The visualization community, on the other hand, has been concerned with the interface between humans and electronically stored data. The complexity of the mining algorithms and the size of the data being mined make high performance computing an essential ingredient of successful and time-critical data mining. Fayyad et al. (1996) define KDD as the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data. John (1997) defines it as the process of discovering advantageous patterns in data. Because the use of automated methods distinguishes KDD from any traditional methods of data analysis, the authors of this paper define KDD as the partially automated process of finding potentially valid, useful, and previously unknown patterns from large data sets.

A pattern is an expression of describing facts in a subset of a set of facts. The expression is called a pattern if it is simpler than the enumeration of all facts in the subset of facts. For example, ''If the cost of any activity is less than a certain threshold then historical data proves that there is a high probability that the activity is on schedule'' would be one such pattern for an appropriate choice for construction cost. This pattern is illustrated graphically in Fig. 1. Whether a pattern is useful or interesting is in the eye of the user of the system (in this research, the construction manager). If he finds it useful, interesting, and previously unknown, then the pattern becomes a novel pattern. In this research, the novel pattern means that construction managers may manage their construction projects more efficiently in terms of schedule/cost management, etc. through the patterns found.

KDD for Discovering Causal Patterns

The writers intend to apply the KDD process to the provided RMS database with the goal of developing a casual analysis. According to Suchman (1967), causality can be applied when one input appears to imply the occurrence of output, and a causal relationship can be inferred by analyzing how input and output are related or associated. Suchman's model can be applied to a construction project (Fig. 2) for intervening in the chain of events in three different ways: primary intervention (enables prevention at the root); secondary intervention (enables reduction of effects); and tertiary intervention (enables rehabilitation or reduction in the consequences). If the relationship between the chain of events and the interventions is understood, many undesirable outcomes can be prevented. Therefore, by applying KDD to a large database in an attempt to identify the patterns between preconditions and causes in a construction project, it is possible to avoid unexpected consequences.

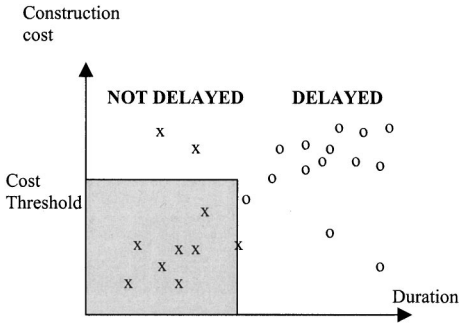

Fig. 1. Simple data set with two classes for finding patterns

KDD Applications

The development of KDD has been powered by the desire to find meaningful, new patterns in real-world databases. Retail data are mined to determine sales and inventory patterns (Anand and Kahn 1992), credit card data are mined for suspect fraudulent activity (Blanchard 1993), molecular sequence data are mined for finding structural motifs (Hofacker et al. 1996), satellite image data are mined for earthquake patterns (Shek et al. 1996), and even basketball statistics are mined to help identify key matchups in upcoming games (Bhandari et al. 1995). In the construction industry, however, there have been relatively few KDD applications (Yang et al. 1998; Simoff 1998), and Buchheit et al. (2000) established a framework on KDD process with a case study of measuring HVAC performance and energy use. Although some important suggestions have been provided by each researcher, the effectiveness of such research still remains unanswered.

Case Study

In this section, a case study by KDD on a large construction database is presented with the sequence of five steps shown in

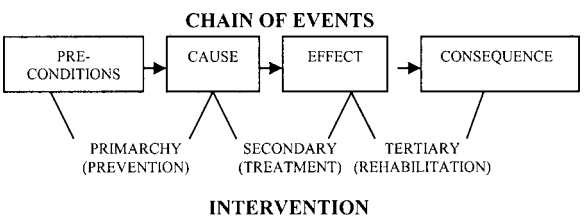

Fig. 2. Suchman's Causal Models (Suchman 1967)

J. Comput. Civ. Eng. 2002.16:39-48.

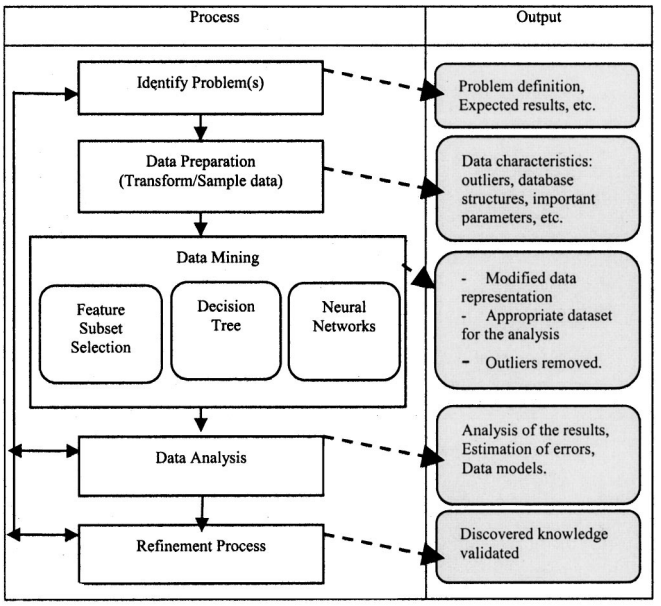

Fig. 3. The main processes of discovery approach

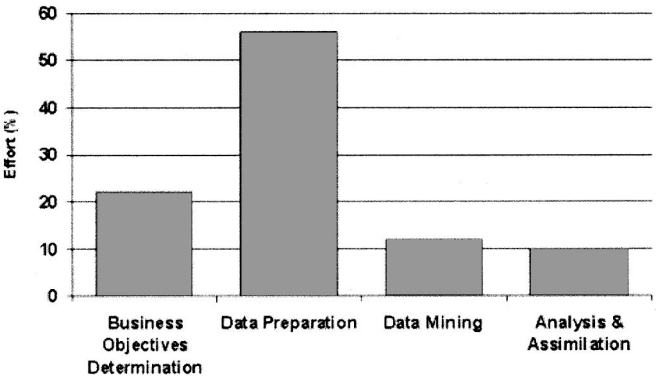

Fig. 4. Required effort for each knowledge discovery in databases step (Cabena et al. 1998)

Fig. 3, where detailed steps are presented during the KDD process and the type of outputs that could be generated with the available tools. The initial data survey for a project in Fort Wayne, Indiana, provided by USACE demonstrated that one activity called ''installation of drainage pipelines'' was behind schedule in 54% of the instances. Identifying causal relationships of construction schedule delay present several challenges for the KDD process. First, construction data usually has limited breadth or coverage. In order to resolve this problem, it is essential to have good data preparation techniques. In addition, mining all the data in the RMS database may not be feasible because there are so many variables and data sets to be dealt with.

To reduce or eliminate possible delays, it is important for the contracting parties, including owners and contractors, to have a clear understanding of causal relationships of schedule delays in a project. This case study presents the patterns found in delays for the activity of a drainage pipeline installation in Fort Wayne, Ind. and provides suggestions for the contracting parties to allow the selection of proactive actions in order to reduce or eliminate any possible delays.

Identifying Problems

The first step of KDD implementation is the identification of problems. Therefore, the domain information of construction schedule delays was obtained before any data were analyzed. In general, domain knowledge can be obtained from two different sources: (1) Experts such as project managers can provide the domain knowledge such as history data or description language and can evaluate the results of generated knowledge learning. In interviews with site managers, it was found that they considered the main cause for schedule delays to be weather related problems. Thus, a KDD prototype was developed to search for the activity delay causes and to see if the managers were working with the right assumptions. (2) Domain knowledge can be retrieved from a careful literature survey. On the basis of literature reviews, research on the development of project control and scheduling has been conducted at many different institutions in order to determine activity durations, develop alternative logic for schedules, integrate cost control and scheduling, and improve existing scheduling techniques (Jaafari 1984; Barrie 1985; Yates 1993). Statistics has been an important tool in finding the correlation or uncertainty of schedule delays by many researchers. Most statistics assume either a normal, chi-square, binomial, or Poisson distribution. However, it is not uncommon to collect data where the underlying frequency distribution is not known or is intractable. The ability of a particular test to remain valid if one or more of its assumptions is violated is known as its robustness. Tests that are robust are less biased than tests that are sensitive to their assumptions. Because one can avoid any hypothetical assumptions by using machine learning instead of statistics, one may consider using KDD techniques when the planned statistical procedure is not robust. The future use of statistical methods in KDD certainly lies in closely coupling tier use with machine learning and database technologies in the form of exploratory data analysis, noise modeling, knowledge validation, and significance testing.

Data Preparation

To date, most modern KDD tools have focused almost exclusively on building models (Brachman et al. 1997). However, data preparation is a very important process because data itself may have been collected in an ad hoc manner, unfilled fields in records may be found, or mistakes in data entry may have been made. As a result, the KDD process cannot succeed without a serious effort to prepare the data. Without the data discovery phase, the analyst will have no idea if the data quality can support the task at all. Once the quality and details are assessed, serious work is usually needed to get the data in shape for analysis. In Fig. 4, Cabena et al. (1998) presents a broad outline of the steps in the KDD process and the relative effort typically associated with each of them. As shown, 60% of the time goes into preparing the data for mining, thus highlighting the critical dependency on clean, relevant data. The actual mining step typically constitutes about 10% of the overall effort. Thus, the process of data preparation is one of the most important parts of the entire process and one of the most time consuming and difficult. Like any other real world applications, RMS data also has several problems such as missing parameter values, improper data types, out-of-range data, incomplete records or instances, and unavailable data. This section reviews the problems in preparing data and presents important procedures of how to use the solutions to get the most out of the data. Proper data preparation can cut preparation time enormously, depending on the quality of the original data, allowing the analyst to

J. Comput. Civ. Eng. 2002.16:39-48.

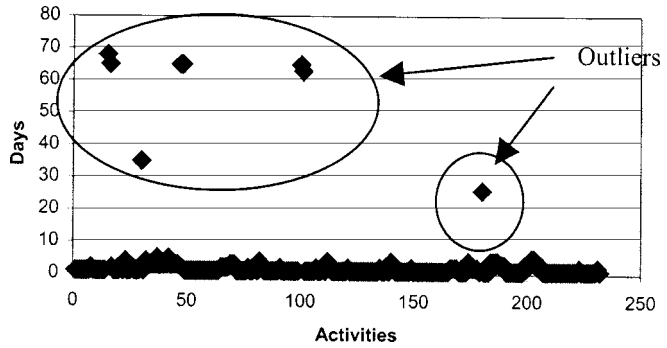

Fig. 5. Outliers in activity duration of 0.15–1.07m drainage pipeline

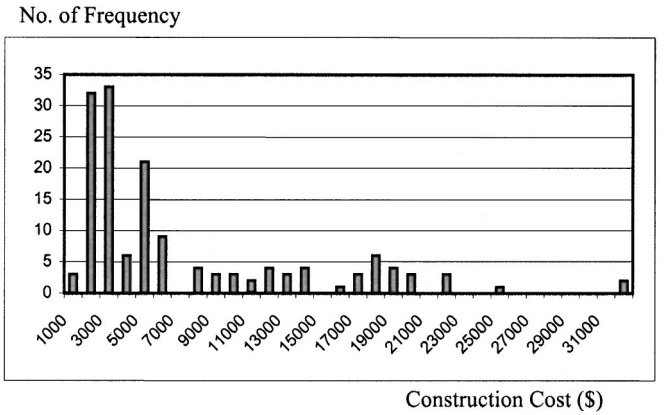

Fig. 6. Distribution of entire dataset for activity costs

produce better models in less time. When data are properly prepared, the analyst gains understanding and insight into the content, range of applicability, and limits of the data.

One may just consider data and not data sets in building a data representation. The data representation can be regarded from two perspectives: as data and as a data set. Data imply that the variables are to be considered as individual entities, and their interactions or relationships to other variables are secondary. The data set implies that not only are the variables themselves considered, but the interactions and interrelationships have equal or greater import. Preparation for mining involves looking at the variables individually as well as looking at the data set as a whole. The following section describes the four types of actions that need to be taken to prepare for data representation and some of the problems that need to be addressed both with data and data sets.

Removing Variables

The number of distinct values and the frequency count of each distinct value are the basic information in KDD applications. From this information, it is determined if a variable is empty. If so, the variable may be discarded. Removing variables becomes more problematic when most of the instance values are empty, but occasionally a value is recorded. The changing value does indeed present some information, but if there are not many actual values, the information density of the variable is low. This circumstance is described as sparsity (Pyle 1999). In general, mining tools deal very poorly with highly sparse data.

Because there are a large number of variables in RMS, which has 72 tables with each table containing more than 98 attributes, removing variables is an important task to be considered. Forty percent (calendar, warranty, holiday, accident, payroll, real property, etc.) of the 72 tables were almost empty and were removed from the data set. Also a table titled, ''contractors'' contained single values and was deleted from the list because the lack of variation in content carried no information for modeling purposes.

Outliers

An outlier is a single or very low frequency occurrence of the value of a variable that is far away from the bulk of the values of the variable. As a general rule of thumb, if it can be established that it is a mistake, it can be rectified. The problem is what to do if it cannot be pinpointed as an error. Outliers in this case study are illustrated in Fig. 5. First it must be determined that outliers are not due to error. The normal duration for a 0.15–1.07m drainage pipeline installation is from 1 to 4 days. However, according to the RMS, some instances took from 30 to 65 days, although they were originally planned to be finished in less than 4 days. With the help of the site managers in the construction project, it was confirmed that the activity had not taken more than 10 days, and those instances, considered as input mistakes, were eliminated from the dataset. The number of outliers removed was 7 out of 231 instances.

Variability and Confidence

If data are displayed as a histogram, a pattern is easier to see. Each column in a histogram can show the number of instances of a particular value. Fig. 6 shows a histogram representing RMS data. The column positions represent the magnitude of each of the values. The height of each column represents the count of instance values of the appropriate measured value. The histogram in Fig. 6 certainly makes some sort of pattern easier to see. The range of construction costs of the activity, ''installation of pipelines,'' is between $1,000 and $33,000, and most instances are gathered around $4,000.

Given a sample set of cases, the common practice is to randomly divide the cases into train-and-test sets. In this example, the whole data set of 224 instances was divided into a training data set of 150 cases (2/3 split) and a testing data set of 74 cases (1/3 split) as recommended by Weiss and Kulikowski (1991). One problem in variability is that until a representative sample is obtained and known to be representative, it is impossible to know if the pattern in some particular random sample represents the true variability of the population. Getting a representative sample can be resolved by a phenomenon called ''convergence.'' If the sample distribution is recalculated with each additional instance added, it will resemble a low number of instances in the sample; that is, each addition will make a large impact on the shape of the curve. However, when the number of instances in the sample is modestly large, the overall shape of the curve will have settled down and will change little as new instance values are added. This settling down of the overall curve shape is the key to deciding the ''convergence'' between two different data sets. Fig. 7 represents the histogram of the sample data set, and Fig. 6 represents the whole set of data sets. Because the two histograms look similar, the two data sets are considered as being in the state of convergence. Another approach to measure the consistency between two data sets is to calculate standard deviation ($\sqrt{[\Sigma(x-m)^2/(n-1)]}$ where x is the instance of value; m is the mean; and n is the number of instance). In many statistical texts, variability is often described in terms of how far the individual instances of the sample are from the mean of the sample. In RMS, means of the whole data set and the sample data set were 8,013 and 7,956, and the standard deviations of the whole data set and

J. Comput. Civ. Eng. 2002.16:39-48.

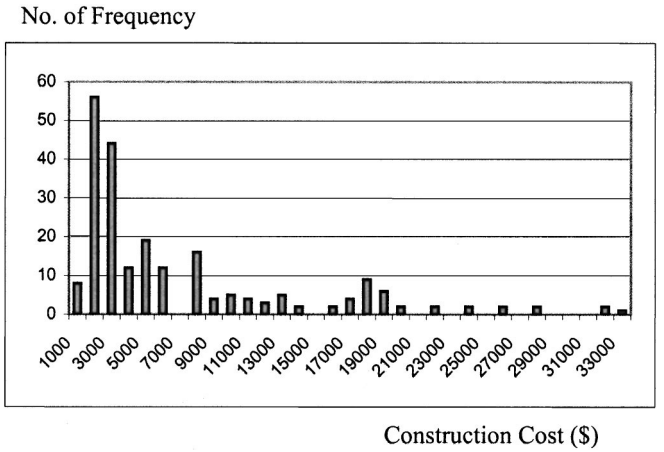

Fig. 7. Distribution of sample data set for activity costs

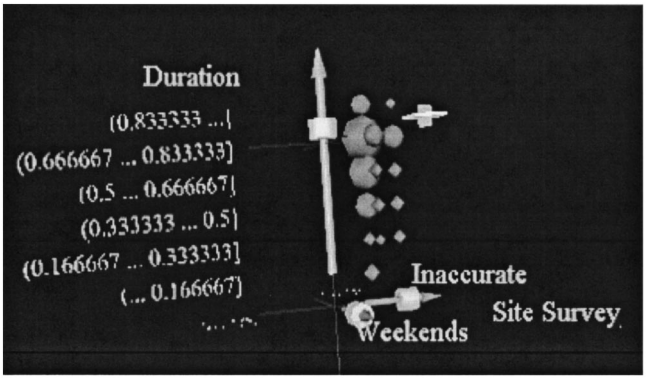

Fig. 9. Capturing variability in 3D for sample dataset of 150 instances

the sample data set were 12,534 and 11,962, respectively. Thus, the two data sets are considered to be in consistency because means and standard deviations are within 5% difference (confidence of 95%).

As a larger and larger random sample is taken, the variability of the sample tends to fluctuate less and less between the smaller and larger samples. This reduction in the amount of fluctuation between successive samples as sample size increases makes the number measuring variability converge toward a particular value. Fig. 8 shows what happens to the standard deviation as the number of instances in the sample increases, as measured along the bottom of the graph. Fig. 8 shows incremental samples in RMS, starting with a sample size of zero, and increasing the sample size by one each time. By simply looking at the graph, intuition suggests that the variability will end up somewhere about 0.55, no matter how many more instances are considered.

The focus of Figs. 9 and 10 is more on the dataset rather than on data itself. The dataset places emphasis on the interactions between the variables; whereas, data focus on individual variables and their instance values. When a sample dataset is obtained, it is important to note that the distribution of the whole dataset is similar to that of the sample dataset. Differently sized, randomly selected samples from the whole population will have different variability measures. As a larger and larger random sample is taken, the variability of the sample tends to be gathered into a certain shape as sample size increases. In capturing variability for an individual variable, a two-dimensional graph is used to measure the variability of each attribute (Fig. 8). By extending the 2D graph to 3D, it is possible to measure the variability among variables in a dataset. Fig. 9 (complete dataset of 224 instances) and Fig. 10 (sample dataset of 150 instances) show similar variability in the dataset through a 3D manifold (x value of inaccurate site survey, y value of duration, and z value of weekends). Therefore, it is noted that when the number of instances in the sample data set is around 150 instances (2/3 split), adding another instance barely makes any difference at all to the overall shape.

Handling Nonnumerical Variables

Because all tools can handle numerical data but some tools cannot handle alpha data, the data analyst needs a method of transforming alpha values into appropriate numerical values. What must be avoided at all costs is an arbitrary assignment of numbers to alpha labels. The initial stage in numerating alphas is for the miner to replace them with a numeration that has some rationale, if possible.

The usual problem in analyzing large data sets is in reducing the dimensionality. There are some circumstances where the dimensionality of a variable needs to be increased when the types of variables, which are almost always categorical, carry information that is best exposed in more than one dimension. In RMS, the field name of FEATURE, for example, includes multiple information in one field from which ''broadscope'' and more than one activity detail can be derived. Thus, broadscope can deliver the information of problem descriptions while activity details represent information in detailed level reports. For the attributes of

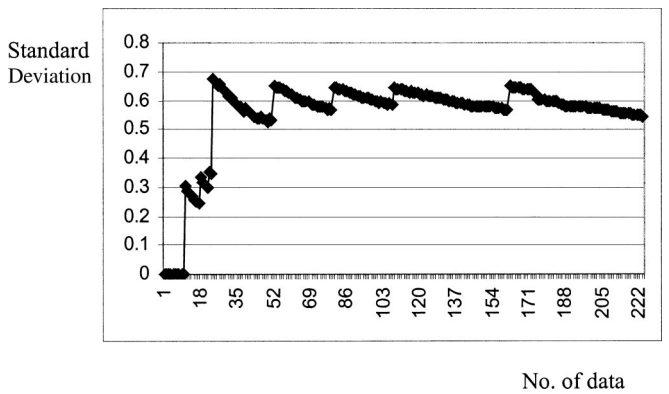

Fig. 8. Measuring variability of duration

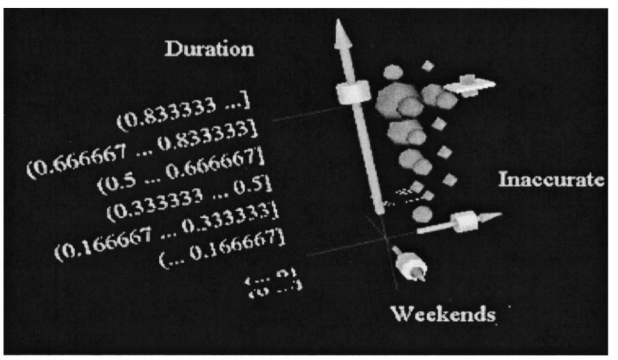

Fig. 10. Capturing variability in 3D for entire dataset of 224 instances

J. Comput. Civ. Eng. 2002.16:39-48.

FEATURE, such as 0.61 m PVC PIPE, 21.3 m PIPE AND GATEWELL, BASKETBALL COURT, CATCH BASINS, and CLEANOUTS, it involves creating 43 new binary pseudo-variables. The numerical value is set to one, indicating the presence of the relevant particular label value and zero otherwise. There are 43 variables, only one of which is ''on'' at any one time. Only one ''on'' of the 43 possible gives ''one-of-*n*'' where, in this case *n* is 43.

Data Mining

Given the application characteristics presented in the previous section, our goal of data mining in this research is to develop an overall data analysis mechanism that can be applied to find patterns that explain or predict any behaviors in construction projects. In data mining, feature subset selection was first used to calculate the relevance of features. Then, decision tree extracted rules from the data sets. Rules from decision tree made the input selection for the neural network a simple task and the understanding of outputs of neural network easier. Finally, neural networks were used to make predictions of the future trends in a construction project.

Feature Subset Selection

The technique of feature subset selection was required in the case study because many different attributes were available in the data set, and it was not obvious which features could be useful for the current problem. Also, practical machine learning algorithms are known to degrade in performance when faced with many features that are not necessary for predicting the desired output. The feature subset selection algorithm conducts a search for a good subset using the induction algorithm as part of the evaluation function. The accuracy of the induced classifiers is estimated using accuracy estimation techniques. The wrapper approach (Kohavi and John 1997) is well known in the machine learning community because of its accurate evaluation and was used in this application. There are two well-known induction algorithms such as the decision tree and the naïve-bayes. In C4.5 decision tree (Quinlan 1993), the tree is constructed by finding the best single-feature test to conduct at the root node of the tree. The Naïve-Bayesian classifier (Domingos and Pazzani 1997) uses Bayes' rules to compute the probability of each class given the instance, assuming the features are conditionally independent.

Decision Trees

A decision tree is a tree-based knowledge representation methodology used to represent classification rules. The leaf nodes represent the class labels while nonleaf nodes represent the attributes associated with the objects being classified. The branches of the tree represent each possible value of the decision node from which they originate. Decision trees are useful, particularly for solving problems that can be cast in terms of producing a single answer in the form of a class name. On the basis of answers to the questions at the decision nodes, one can find the appropriate leaf and the answer it contains. An example of what uses the algorithms above is C4.5. The first stage of C4.5 generates a decision tree. Each level of the decision tree represents a split of the data set. This split is chosen by examining each possible split of the data on each attribute and choosing the one that best splits the data (according to an information theoretic measure of the distribution of classes in each subset). This continues for each level of the decision tree until there is no benefit from further segmenting the data. Once this has been done, rules are generated by traversing each branch of the tree and collecting the conditions at each branch of the decision tree. Each generated rule has a confidence percentage associated with the class it predicts. The uncertainty is caused by the generalization process, as some leaves on a tree may no longer contain single labels.

Neural Networks (NN)

The foundation of the neural networks paradigm was laid in the 1950s, and NN have gained significant attention in the past decade because of the development of more powerful hardware and neural algorithms (Rumelhart 1994). Artificial neural networks have been studied and explored by many researchers where they have been used, applied, and manipulated in almost every field. For example, they have been used in system modeling and identification (Narendra and Parthasarathy 1990), control (Werbos 1989; Jordan and Jacobs 1990), pattern recognition (Renals et al. 1992; LeCun et al. 1990), speech pronounciation ''NETalk'' (Sejnowski and Rosenberg 1987), system classifications (Haykin and Bhattacharya 1992; Casselman et al. 1991), medical diagnosis (Harrison et al. 1990), and they have been applied in prediction, computer vision, and hardware implementations. As in civil engineering applications, neural networks models have been employed in different studies. Some of these studies cover the mathematical modeling of nonlinear structural materials, damage detection, nondestructive analysis, earthquake classification, dynamical system modeling, system identifications, and structural control of linear and nonlinear systems (Bani-Hani and Ghaboussi 1998; Tsai et al. 1999; Moselhi 2000).

Among the numerous artificial neural networks that have been proposed, backpropagation networks have been extremely popular for their unique learning capability (Widrow et al. 1994). Backpropagation networks (Rumelhart et al. 1986) are layered, feed-forward models. Activations flow from the input layer through the hidden layer, then to the output layer. A backpropagation network typically starts out with a random set of weights. The network adjusts its weights each time it sees an input-output pair. Each pair is processed at two stages, a forward pass and a backward pass. The forward pass involves presenting a sample input to the network and letting activations flow until they reach the output layer. During the backward pass, the network's actual output is compared with the target output and error estimates are computed for the output units. The weights connected to the output units are adjusted in order to reduce the errors (a gradient descent method). The error estimates of the output units are then used to derive error estimates for the units in the hidden layer. Finally, errors are propagated back to the connections stemming from the input units, and the backpropagation network updates its weights incrementally until the network stabilizes.

Data Analysis

Results from C4.5 Decision Tree

Fig. 11 shows each level of the decision tree built with data from the project in Fort Wayne, Indiana. Interesting patterns can be found as follows. Each box in the tree in Fig. 11 represents a node. The top node is called the root node. A decision tree grows from the root node, so the tree can be thought as growing upside down, splitting the data at each level to form new nodes. The resulting tree comprises many nodes connected by branches. Nodes that are at the end of branches are called leaf nodes and play a special role when the tree is used for prediction. In Fig. 11, each node contains information about the number of instances and percentages at that node and about the distribution of dependent

J. Comput. Civ. Eng. 2002.16:39-48.

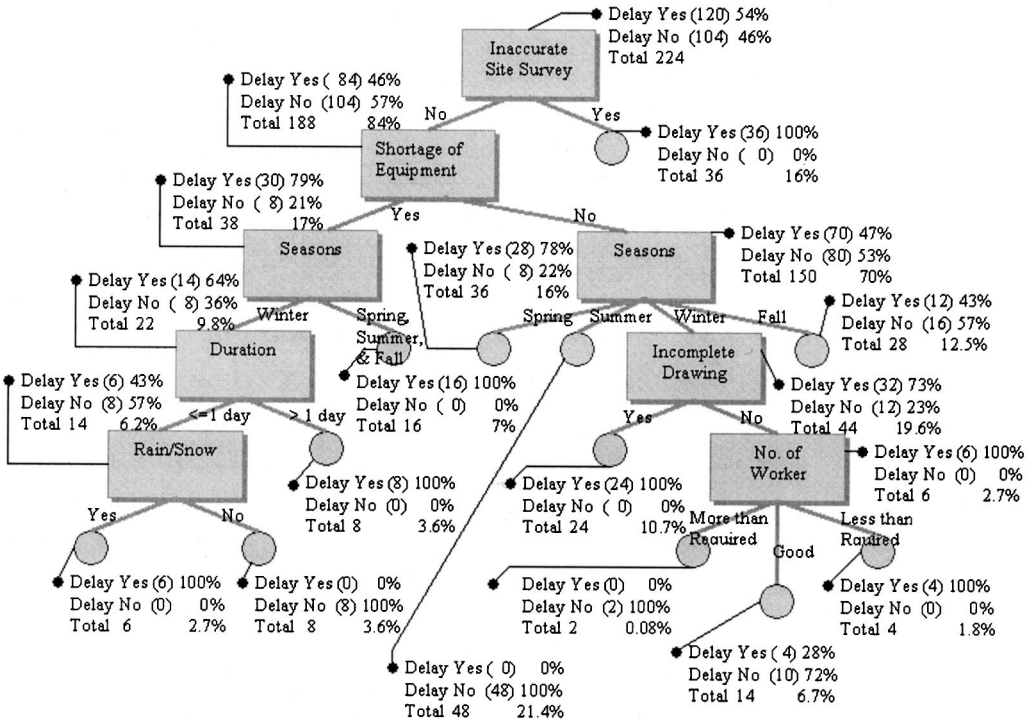

Fig. 11. Decision trees of schedule delays on drainage pipeline activities

variable values. The instances at the root node are all of the instances in the training set. This node contains 224 instances, of which 54% are instances of delay and 46% are of no delay. Below the root node (parent) is the first split that, in this case, splits the data into two new nodes (children) based on whether inaccurate site survey is yes or no. The rightmost node (yes of inaccurate site survey that has three values such as ''yes'', ''no'', or ''unknown'' in RMS database where yes means that a problem in terms of inaccurate site survey was reported during the construction) resulting from this split contains 36 instances, all of which are associated with schedule delays. Because all instances have the same value of the dependent variable, this node is considered pure and will not be split further. The leftmost node in the first split contains 188 instances, 46% of which are schedule delays. The leftmost node is then further split on the basis of the value of shortage of equipment. A decision tree algorithm determines the order of the splits, inaccurate site survey, shortage of equipment, etc.

A tree that has only pure leaf nodes is called a pure tree, a condition that is not only unnecessary but is usually undesirable. Most trees are impure; that is, their leaf nodes contain cases with more than one outcome. Fig. 11 reveals the following interesting patterns:

- Weather that is considered responsible for delays by site managers appears not to be the most important cause in determining delays.
- Instances of pipeline installation activity with inaccurate site surveys are always delayed in the schedule.
- Shortage of equipment, seasons, and incomplete drawing are very significant factors in determining activity delay because the induction algorithm tried to prioritize its splits by choosing the most significant split first.

Once the decision tree is built, the tree can be used for predicting a new case by starting at the root (top) of the tree and following a path down the branches until a leaf node is encountered. The path is determined by imposing the split rules on the values of the independent variables in the new instance. Navigating a tree to produce predicted values can become cumbersome as trees increase in size and complexity. It is possible to derive a set of rules for a tree with one rule for each leaf node simply by following the path between the root and that leaf node. The rules for the leaf nodes in Fig. 11, taken top to bottom and left to right, are as follows:

- If inaccurate site survey=yes then delay=yes.
- If inaccurate site survey=no, shortage of equipment=yes, and season=summer then delay=yes.
- If inaccurate site survey=no, shortage of equipment=yes, season=winter, and duration >1 day then delay=yes.
- If inaccurate site survey=no, shortage of equipment=yes, season=winter, and duration <=day, rain/snow=yes then delay=yes.

Prediction of Trends through Neural Networks

The nine input variables (inaccurate site survey, number of workers, incomplete drawing, change order, shortage of equipment, duration, season, weekends, rain/snow) were selected. For example, the data inaccurate site survey was stored into the RMS when construction personnel finds something unexpected during the construction. Then, types of each variable were converted to numbers. The output value is the delay for an activity. The training cycle is repeated for each case in the training set, with small adjustments being made in the weights after each case. When the entire training set has been processed, it is processed again. In deciding the appropriate number of hidden layers and the best learning rate, a great number of NN were run. In this case study, it was found that the best result was given with a 1% learning rate and a three layer backpropagation NN architecture.

As shown in Fig. 12, the training error always decreases with an increase in the number of cycles. In contrast, the testing error does not have a continuously decreasing trend where a minimum

J. Comput. Civ. Eng. 2002.16:39-48.

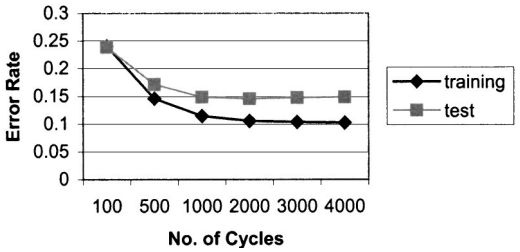

Fig. 12. Comparison of error rates between training and testing sets

value is found on the curve. Overfitting is the phenomenon where in most cases a network gets worse instead of better after a certain point during training. This is because such long training may make the network memorize the training patterns, including all of their peculiarities. However, one is usually interested in the generalization of the network. Learning the peculiarities of the training set makes the generalization worse. The network should only learn the general structure of the examples.

As the number of cycles increase, the training error rate should generally decrease. The error rate on test cases should begin to decrease, and then it will eventually turn upward. This corresponds to the dynamics of underfitting and then overfitting. Fig. 12 shows the comparison between plots of training and testing errors as the number of cycles increases. In NN, one would like to train on the training data set and estimate the true error rate with the test data set. Fig. 12 shows the optimum point around 2,000 cycles where the training set error rate continues to decrease but where the test set error rate is bouncing back. Thus, it is noted in Fig. 12 that the most appropriate number of cycles in the RMS data set is 2,000, where the training error rate is 10.56% and the testing error rate is 14.55%. NN results are described in the next section.

Refinement Process—Comparison between Traditional and KDD Predictions

There is the essential need to validate results from KDD models because the inability to adequately evaluate KDD models may become the limiting factor in our ability to utilize KDD technologies. In this research, the validation was conducted by comparing the results from the KDD process and publications or any project-control software that the construction expert heavily relies on in the industry.

Validation with Estimating Data Provided by RSMeans

RSMeans provides simple and up-to-date labor and material costs, cost index, productivity rates, and so on. When one wants an answer for a specific situation of a project from RSMeans, however, it does not provide detailed information, because of the fact that RSMeans deals with mean values only and does not allow users to access information by type of project, specific location, or contracting parameters. Each of these parameters is important for proper project planning and control. The NN demonstrates the importance of considering all the possible factors that might affect schedule durations. According to RSMeans (CMD Group 1999), it takes 3.2 days for 320 units of an activity of 0.15–1.07m pipe line installation with 10 workers a day; whereas, it will take 4.96 (most optimistic inputs) to 6.86 (most pessimistic inputs) days according to the NN built with the Fort

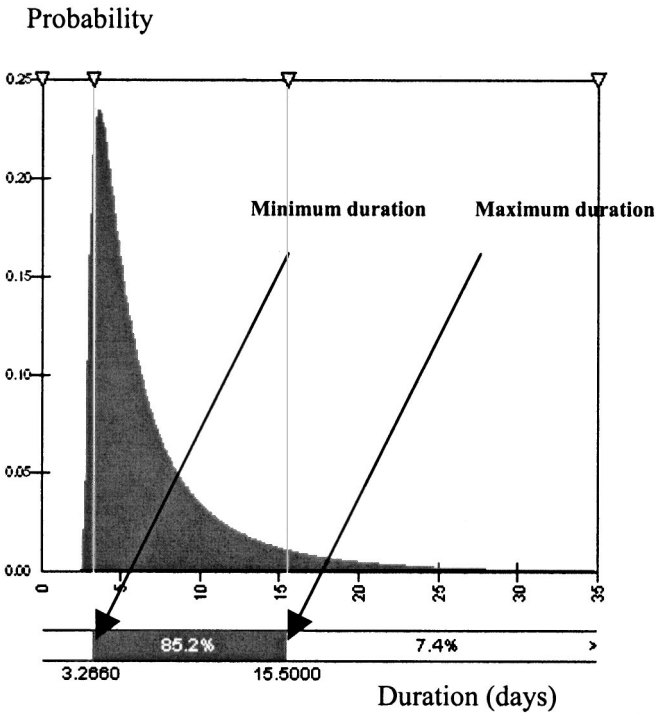

Fig. 13. Monte Carlo simulation for an activity of 320 units with 10 workers a day in flood control project (confidence level of 85.2%)

Wayne data to do the same amount of work with the same 10 workers, where NN considers all the possible causes such as inaccurate site survey, shortage of equipment, rain/snow, seasons, and so on. Therefore, it is found that RSMeans does not consider all the specific situations in a project that might cause any potential problems during a construction project.

Validation with Monte Carlo Simulation

Most of the existing project estimation tools such as Monte Carlo simulation help construction managers estimate when a construction project is to be completed by sampling its cumulative distribution function. In many scheduling tools, construction managers assign activity codes for three durations: minimum, maximum, as well as most likely. The simulation package assigns a duration to each task that is randomly generated somewhere in the range between the minimum and maximum figures assigned to each task. The result of a Monte Carlo simulation is a range of end dates, each of which has a number attached to it indicating the probability of completing the project by that date. According to a Monte Carlo simulation (Inverse Gaussian distribution, shown in Fig. 13), it takes 3.27 (minimum) to 15.5 (maximum) days for 320 units of an activity of 0.15–1.07m pipeline installation with 10 workers a day (the confidence was set at 85.2% in order to compare its result to NN's result. The confidence rate of NN −14.8% in the previous section). Because the duration in the Monte Carlo is given in a probabilistic range, the construction manager's past experience plays a very important role in choosing the most likely duration between minimum and maximum durations. Compared to the result of Monte Carlo, the predictions of NN give a more realistic estimation of the construction duration because it considers all the specific situations that might cause any potential problems during a construction phase. Knowledge generated by the tools presented in this paper would help project

J. Comput. Civ. Eng. 2002.16:39-48.

managers estimate the values for the minimum, maximum, and most likely values for future simulations without relying solely on his own experience.

Cost/Benefit

According to the preliminary results of this case study, the main cause of schedule delays in the flood control project at Fort Wayne was inaccurate site survey rather than the weather related problems initially assumed by site managers. Discussions with site managers on the construction project confirmed the importance of equipment, such as ground penetration radar, to make the site surveys more accurate. Ground penetrating radar (GPR) is equipment used to locate existing underground pipelines and construction structures and is considered to be very cost-effective equipment in a flood control project because one of the most frequent problems in schedule delay was due to inaccurate site survey. The potential savings to be obtained after buying a GPR was calculated for the 4th phase of the Fort Wayne project. We were able to predict a significant amount of savings if compared to the $5,000 investment it costs to buy GPR equipment. The $587,391 savings were obtained by the multiplication of daily construction cost by the expected number of instances (75) for the activity of drainage pipeline installation during the 4th stage and by the number of days to be saved by using the GRP (0.83 days) [$587,391=9,436×75×0.83]. The number of days to be saved by the GPR was obtained from running the NN with the weights learned from previous projects. In addition, expected savings would increase even more if schedule related penalties for delays are considered.

Conclusions

With the use of very large database, this research utilizes KDD technologies that reveal predictable patterns in construction data that were previously thought to be chaotic. To date, the use of KDD in civil engineering is only in its infancy where the sizes of the databases were quite limited and, hence, did not provide conclusive results. This research applied the KDD process to analyze the data to extract knowledge or patterns from the large RMS database so that the project manager may have a better understanding of causal relationships in a project. In this paper, the authors discussed an approach for discovering some useful knowledge from large amounts of data that were generated during a construction project. This paper introduced a knowledge discovery approach that is being developed for this real world application. This approach consists of five steps: (1) identification of problems; (2) data preparation; (3) data mining; (4) data analysis; and (5) refinement process. The proposed approach helps to guide the analysis through the application of diverse discovery techniques. Such a methodological procedure will help us to address the complexity of the domain considered and, therefore, to optimize our chance to discover valuable knowledge.

During the knowledge discovery approach, one of the most important, time-consuming, and difficult parts of the KDD process was data preparation. Domain knowledge and a good understanding of the data are key to successful data preparation.

In this research, one case study was conducted by the authors to identify the causes of schedule delays. But its possible applications can be extended to different areas such as identifying the causes of cost overrun, or quality control/assurance from the RMS database. The research of the KDD process to large construction data is continuously being refined, and more case studies are being followed. Eventually, knowledge discovery model-building framework developed by this research will be used to guide novice construction model builders through the process of creating models based on their own construction data.

References

Anand, S., et al. (1998). ''Discovering case knowledge using data mining,'' PAKDD, 25–35.

Anand, T., and Kahn, G. (1992). ''SPOTLIGHT: A data explanation system.'' *Proc. CAIA-92, Proc., 8th IEEE Conf.*, Piscataway, N.J., 2–8.

Bani-Hani, K., and Ghaboussi, J. (1998). ''Nonlinear structural control using neural networks.'' *J. Eng. Mech. Div.*, 124(3), 319–327.

Barrie, D. (1985). *Professional construction management*, McGraw-Hill, New York.

Bhandari, I., et al. (1995). ''Advanced scout: Data mining and knowledge discovery in NBA data.'' *Int. J. Data Mining Knowledge Discovery*, 1, 121–125.

Blanchard, D. (1993). ''Neural-based fraud detection system.'' MicroSoft Intelligence System Report, Atlanta, **2**, 28–35.

Brachman, et al. (1997). ''Visual data mining: Recognizing telephone calling fraud.'' *Data Mining and Knowledge Discovery*, 1, 33–46.

Buchheit, R. B., et al. (2000). ''A knowledge discovery case study for the intelligent workplace.'' *Proc., Computing in Civil Engineering*, Stanford, Calif., 914–921.

Cabena, P., et al. (1998). *Discovering data mining from concept to implementation*, Int. Technical Support Organization (IBM), Prentice Hall, Upper Saddle River, N.J., 42–44.

Casselman, F. L., Freeman, D. F., Kerringan, D. A., Lane, S. E., Millstrom, N. H., and Nichols Jr., W. G. (1991). ''A neural network-based passive sonar detection and classification design with a low false alarm rate.'' *IEEE Conf. Neural Networks for Ocean Engineering*, Washington, D.C., 49–55.

CMD Group. (1999). ''RSMeans—Building construction cost data.'' RSMeans, Inc., New York.

Domingos, P., and Pazzani, M. (1997). ''On the optimality of the simple bayesian classifier under zero-one loss.'' *Mach. Learn.*, 29, 103–130.

Fayyad, U., et al. (1996). *Advances in knowledge discovery and data mining*, AAAI Press/MIT Press, Cambridge, Mass., 1–34.

Harrison, R., Marshall, S., and Kennedy, R. (1991). ''The early diagnosis of heart attacks: A neurocomputational approach.'' *Int. Joint Conf. on Neural Networks*, Vol. **1**, Seattle, Wash., 1–5.

Haykin, S., and Bhattacharya, T. K. (1992). ''Adaptive radar detection using supervised learning networks,'' Computational Neuroscience Symposium, Indiana Univ.-Purdue Univ., Indianapolis, 35–51.

Hofacker, I., Huynen, M., Stadler, P., and Stolorz, P. (1996). ''Knowledge discovery in RNA sequence families of HIV using scalable computers.'' *Proc., 2nd Int. Conf. Knowledge Discovery and Data Mining*, Portland, Ore., 20–25.

Jaafari, A. (1984). ''Criticism of CPM for project planning analysis.'' *J. Constr. Eng. Manage.*, 110(2), 222–223.

John, G. (1997). ''Enhancements to the data mining process.'' PhD thesis, Stanford Univ., Stanford, Calif.

Jordan, M. I., and Jocobs, R. A. (1990). ''Learning to control an unstable system with forward modeling.'' *Advances in neural information processing systems 2*, Morgan Kaufmann, San Mateo, Calif., 324–331.

Kohavi, R., and John, G. (1997). ''Wrappers for feature subset selection.'' *Artificial Intelligence J.*, 97, 273–324.

LeCun, Y., Denker, J. S., Henderson, D., R. E., and Jackel, L. D. (1990). ''Handwritten digit recognition with a back-propagation network.'' *Advances in Neural Information Processing Systems*, Morgan Kaufmann, San Mateo, Calif., 396–404.

Moselhi, O., and Shehab-Eldeen, T. (2000). ''Classification of defects in sewer pipes using neural networks.'' *J. Infrastruct. Syst.*, 6(3), 97–104.

Narendra, K. S., and Parthasarathy, K. (1990). ''Identification and control of dynamical systems using neural networks.'' *IEEE Trans. Neural Netw.*, 1, 4–27.

J. Comput. Civ. Eng. 2002.16:39-48.

Pyle, D. (1999). *Data preparation for data mining*, Morgan Kaufman, San Mateo, Calif.
Quinlan, R. (1993). *C4.5 programs for machine learning*, Morgan Kaufmann, San Mateo, Calif.
Renals, S., Morgan, N., Cohen, M., Franco, H., and Bourlad, H. (1992). ''Improving statistical speech recognition.'' *Int. Joint Conf. Neural Networks*, Vol. 2, 302–307.
Rumelhart, D. E. (1986). ''Learning internal representations by error propagation.'' *Parallel distributed processing*, Vol. 1, MIT Press, Cambridge, Mass., 318–362.
Rumelhart, D. E. (1994). ''The basic ideas in neural networks.'' *Commun. ACM*, 37(3), 87–92.
Sejnowkski, T. J., and Rosenberg, C. R. (1987). ''Parallel networks that learn to pronounce english text.'' *Complex Syst.*, 1, 145–168.
Shek, E., Stolorz, P., Muntz, R., Mesrobian, E., and Ng, K. (1996). ''Fast spatio-temporal data mining of large geophysical datasets.'' *Proc., 1st Int. Conf. Knowledge Discovery and Data Mining*, AAAI Press, Montreal, 300–305.
Simoff, S., and Maher, M. L. (1998). ''Ontology-based multimedia data mining for design information retrieval.'' *Proc., Computing in Civil Engineering*, Boston, 212–223.
Suchman, L. (1967). *Casual analysis*, U.S. General Accounting Office, 6–17.
Tsai, C., and Lee, T.-L. (1999). ''Back-propagation neural network in tidal-level forecasting.'' *J. Waterw., Port, Coastal, Ocean Eng.*, 125(4), 195–202.
Weiss, S. and Kulikowski, C. (1991). *Computer systems that learn*, Morgan Kaufmann, San Mateo, Calif., 28–32.
Werbos, P. J. (1989). ''Backpropagation and neurocontrol: A review and prospectus.'' *Int. Joint Conf. Neural Networks*, Vol. 1, Washington, 209–219.
Widrow, B., et al. (1994). ''Neural Networks: Application in industry, business, and science.'' *Commun. ACM*, 37(3), 93–105.
Yang, M., Wood, W. H., and Cutcosky, M. R. (1998). ''Data mining for thesaurus generation in informal design retrieval.'' *ASCE Proc. Computing in Civil Engineering*, 189–200.
Yates, J. K. (1993). ''Construction decision support system for delay analysis.'' *J. Constr. Eng. Manage.*, 119(2), 226–244.

J. Comput. Civ. Eng. 2002.16:39-48.