복잡한 예측문제에 대한 이차학습방법: Video-On-Demand에 대한 사례연구

김형관* · 주종형*

Second-Order Learning for Complex Forecasting Tasks: Case Study of Video-On-Demand

Steven H. Kim* and Jonghyung Joo*

ABSTRACT

To date, research on data mining has focused primarily on individual techniques to support knowledge discovery. However, the integration of elementary learning techniques offers a promising strategy for challenging applications such as forecasting nonlinear processes.

This paper explores the utility of an integrated approach which utilizes a second-order learning process. The approach is compared against individual techniques relating to a neural network, case based reasoning, and induction. In the interest of concreteness, the concepts are presented through a case study involving the prediction of network traffic for video-on-demand.

Key words: Neural networks, case based reasoning, induction.

INTRODUCTION

An intelligent supervisory system should be able to predict the dynamic behavior of a target

* Graduate School of Management, Korea Advanced Institute of Science and Technology.

system as it reacts to transient environmental conditions. To this end, extensive research has been conducted over the past few decades on individual learning techniques from artificial intelligence. However, the integration of multiple techniques offers a promising path to improve performance even further(Kim, 1994a, 1994b; etc.). In particular, an approach we call *second-order* learning involves an integrated architecture where an adaptive module builds on the results of multiple outputs from a preceding stage.

In the interest of concreteness, the concepts are presented through a case study involving the prediction of network traffic involving flows of digitized video. Video-on-demand(VOD) is an interactive service in which programs(such as movies, home shopping, etc.) are provided to users through a network. VOD services are expected to represent a major segment of transmission capacity in broadband integrated digital networks(Kim et al., 1996, Petit et al., 1994).

This paper presents a comparative study of explicit and implicit methods to predict network traffic for VOD services. The traffic patterns are characterized by stochastic as well as chaotic behavior. The evaluated models include case based reasoning, induction, and a neural network, as well as a second-order learning module.

METHODOLOGY

An integrated learning system should have the ability to accommodate a diversity of adaptive techniques among its component subsystems. A number of the most promising techniques to employ in an integrated architecture is as follows.

Neural Networks.

Neural networks are characterized by robustness and graceful degradation (Hopfield, 1982; Sejnowski, 1985). The most common type of neural network and training procedure takes the form of backpropagation (BPN). Once trained, a backpropagation neural network responds to a given input pattern with exactly the same output pattern every time a particular input pattern is presented (Grossberg, 1974; Rumelhart et al., 1986).

Case Based Reasoning.

A learning system should make increasingly useful decisions as it accumulates experience. This is the express goal of the work in case based reasoning (CBR). The CBR methodology can be effective even if the knowledge base is imperfect. Certain techniques of automated learning, such as explanation-based learning, work well only if a strong domain theory exists. In contrast, CBR can use many examples to overcome the gaps in a weak domain theory while still taking advantage of the fragmentary knowledge (Porter et al., 1990). CBR can also be used when the descriptions of the cases, as well as the domain theory, are incomplete (Sycara and Navinchandra, 1991). The approach has been utilized in a number of forecasting applications (Kim, 1995a, 1995b; etc.). Figure 1 presents the predictive procedure through case based reasoning using composite neighbors.

1. Begin with current case $x(t)$.

2. Seek the L neighboring cases $x(t_i)$ in the past which are closet to $x(t)$ according to the euclidean distance function:

$$d_i = [\sum (x(t_i) - x(t))^2]^{\frac{1}{2}}$$

3. Compute the sum of distances:

$$d_{tot} = \sum_{i=1}^{L} d_i$$

4. Determine the relative weight of i^{th} neighbor:

$$W_i = \frac{1}{L - F}[1 - \frac{Fd_i}{d_{tot}}]$$

where F is a weighting factor.

5. Find the successor $x(t_i + 1)$ of each case $x(t_i)$ in the set of neighbors.

6. Calculate the forecast for $t + 1$ as the weighted sum of successors:

$$y(t+1) = \sum_{i=1}^{L} W_i\, x(t_i + 1)$$

Figure 1. Predictive procedure through standard case based reasoning using composite neighbors.

Inductive learning.

In this approach, a set of training examples is collected as input to the inductive learning program. During the training phase, the description of a class is induced from individual examples. Each example takes the form of a case and is described by a vector of attribute values. Concept descriptions are generated to describe each class.

Inductive inference is a problem-solving process in which solutions (inductive concept descriptions) are obtained through a process of search and generalization (Shaw and Gentry, 1990). This paper uses a tree-induction technique utilizing the metric of entropy[1] from information theory (Quinlan, 1986).

Integrated models and second-order learning.

There are many ways to integrate elementary learning techniques. For instance, one might use the output from a CBR module in conjunction with raw data as inputs into a neural module, or employ the converse architecture (Kim and Oh, 1996; etc.).

Another approach to integration involves a multistrategy technique which may be called *second-order* learning. This approach refers to the selection of an elementary model based on its anticipated performance. In this paper, inductive reasoning is used to select among forecasts from two other learning techniques.

The elementary techniques representing first-order learning relate to CBR and backpropagation. Subsequently the second stage employs inductive reasoning to determine the conditions under which CBR or backpropagation performs better than the other. Then the forecast from the superior method is selected on a case-by-case basis to determine the output of the overall system. In other words, the second stage serves as a metalevel predictive system to discern which of two elementary modules (CBR or backpropagation) will perform better.

CASE STUDY

The utility of an integrated approach to forecasting was investigated through a case study in-

1) Entropy is used in information theory as a measure of the amount of information in a message based on the average number of bits needed to encode all possible messages in an optimal coding (Shannon and Weaver, 1949).

volving the total waiting time for a request in a VOD network. The overall architecture for the simulation software is presented in Figure 2.

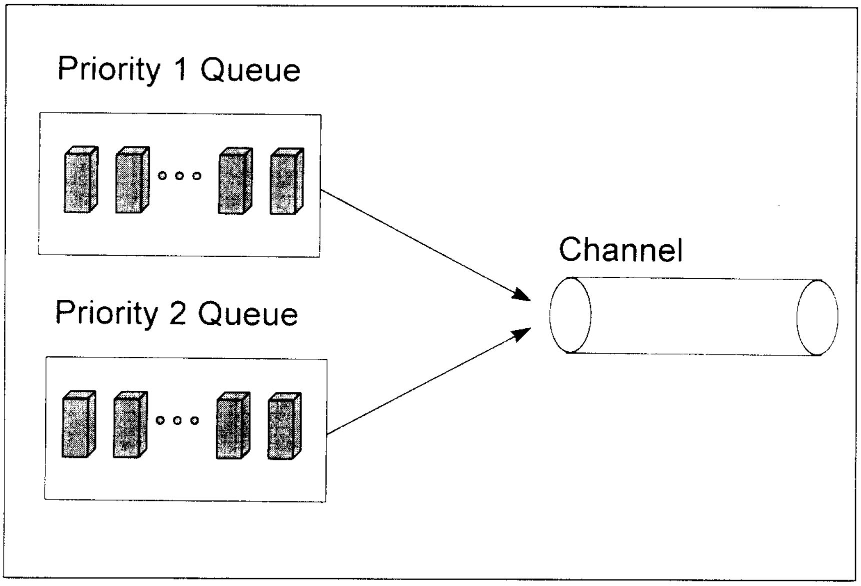

Figure 2. High-level structure of the simulation environment. The service policy is first-in-first-out within priority; that is, a priority 1 request supersedes a priority 2 request. Pre-emption is disallowed: a new job must await the completion of a job in process. In particular, a priority 1 job must wait the completion of a priority 2 job if the latter is already in midstream.

The inputs into the predictive system consisted of three sources: the interarrival time (x), the transmission time (y), and the priority (r) of requests. We assumed that x would follow the Henon model and y the Lorentz process [2], while r would be a random process, as indicated in Table 1.

2) The Henon and Lorentz models represent chaotic processes which are difficult to predict. The two processes are often utilized in chaotic analyses due to the simplicity of the models, which nevertheless yield such complex behavior.
However, as in simulation in general, the exact nature of the process is not critical for our investigation. The important point is the need for a fixed stream of data points to use across trials in order to compare different forecasting models.

Table 1. List of variables. The aggregate waiting time w depends on the interarrival time, transmission time, and the priority of request.

Variable	Description	Generation
x	interarrival time	Henon
y	transmission time (size of file)	Lorentz
r	priority of request	Random
w	aggregate waiting time	*Simulation output*

The total waiting time w_t is the longest wait at time t among all the jobs currently in the system. More precisely, let w_{jt} be the waiting time for job j at time t; that is, the time elapsed since its arrival into the system. Then the total waiting time w_t is defined as the cumulative delay before the last job begins service:

$$w_t \equiv \sum_j w_{jt}$$

The total waiting time depends on variables x, y, and r.

The Henon model is a discrete system which takes the following form.

$$x_{t+1} = 1 - Ax_t^2 + y_t$$
$$y_{t+1} = Bx_t$$

The Henon model exhibits chaotic behavior for parameter values $A = 1.4$ and $B = 0.3$. The corresponding observations are depicted in Figure 3.

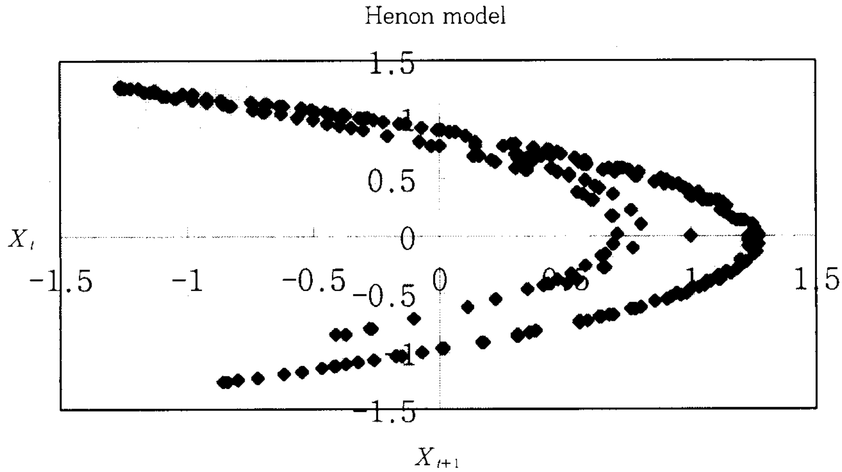

Figure 3. Plot of the general Henon model.

The discrete version of the Lorentz model takes the following form.[3)]

$$x_{t+1} = (1 - \sigma\triangle t)x_t + (\sigma\triangle t)y_t$$
$$y_{t+1} = (1 - \triangle t)y_t + (R - z_t)\triangle t x_t$$
$$z_{t+1} = (1 - b\triangle t)z_t + x_t y_t \triangle t$$

Chaotic behavior arises for the following parameter values: $\sigma = 10$, $\triangle t = 0.025$, $R = 28$, and $b = 8/3$. The associated trajectory is portrayed in Figure 4.

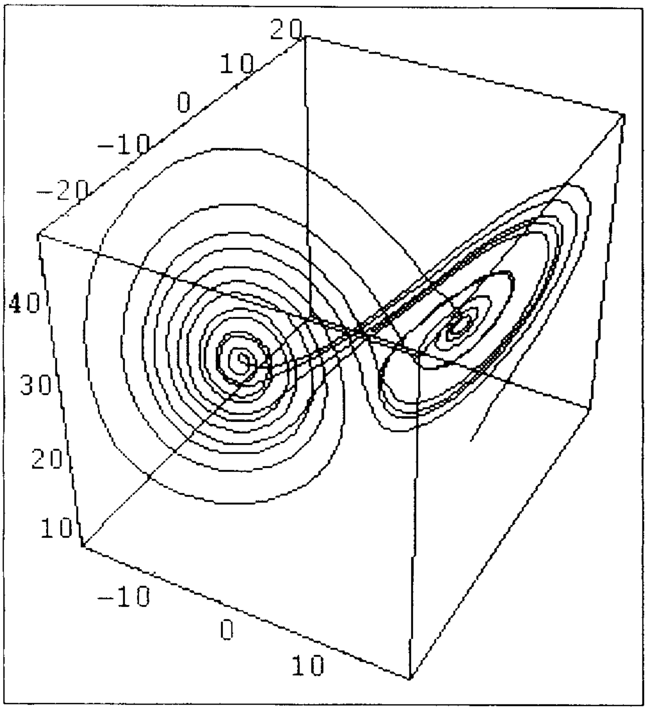

Figure 4. Plot of the general Lorentz model.

In this paper, the Henon and Lorentz data were rescaled to the range between 0 and 1. The resulting process along a single dimension is shown in Figures 5 and 6, respectively. The Henon

3) The roots of the Lorentz process lie in a meterological model(Lorentz, 1963). The model, however, can - and often is - regarded as a generic description of a chaotic process. As indicated in the previous footnote, the interpretation of the 3 variables in the model is of no significance in employing the model to generate a nonlinear data series. In particular, any of the 3 variables may be regarded as a quasi-random variable to yield a chaotic sequence.

series was used as input values for the interarrival time x, and the Lorentz series for the transmission time y.

Figure 5. Partial plot of 100 interarrival times for a Henon process.

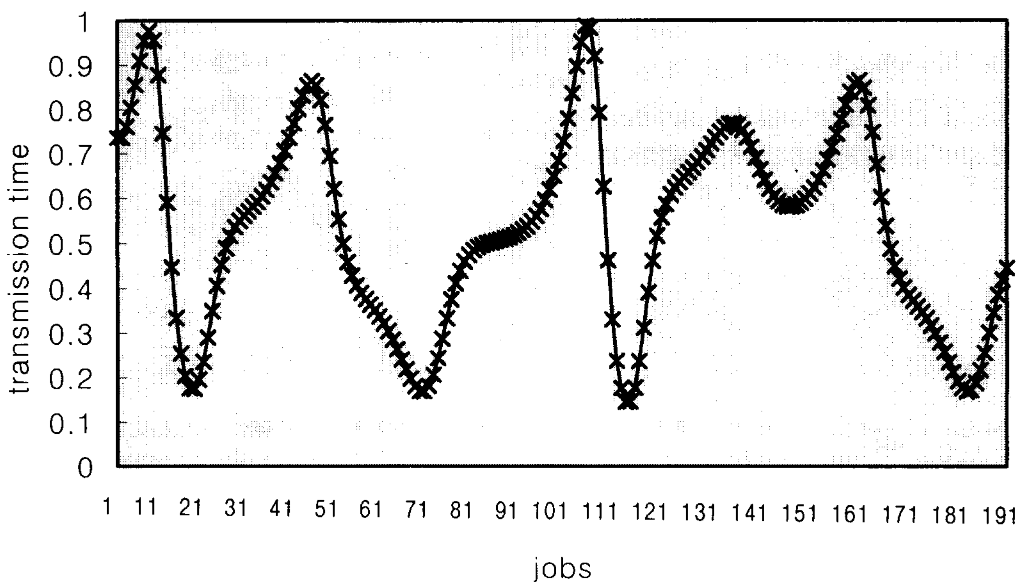

Figure 6. Partial plot of 200 transmission times for a Lorenz process.

A random number generator was utilized to determine the priority r of each request or job: 1 for high priority, and 2 for low priority. A partial plot of the generated priority values is presented in Figure 7.

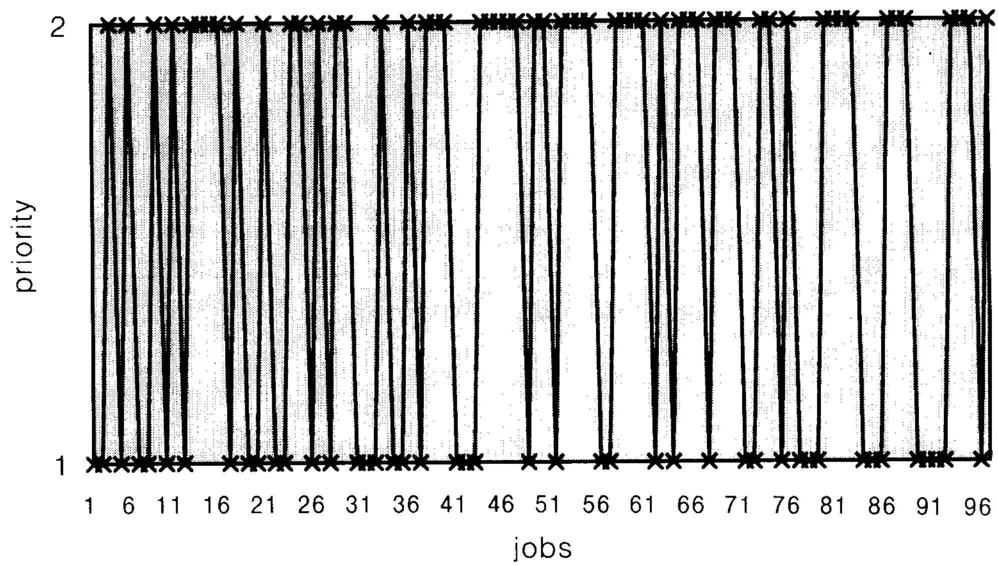

Figure 7. Partial plot of 100 priority values through random number generation.

The data sets were used as inputs into the simulation architecture shown in Figure 2. The output of the simulation model was the aggregate waiting time w_t for a job which entered the system at time t. Figure 8 depicts the resulting sequence of observations.

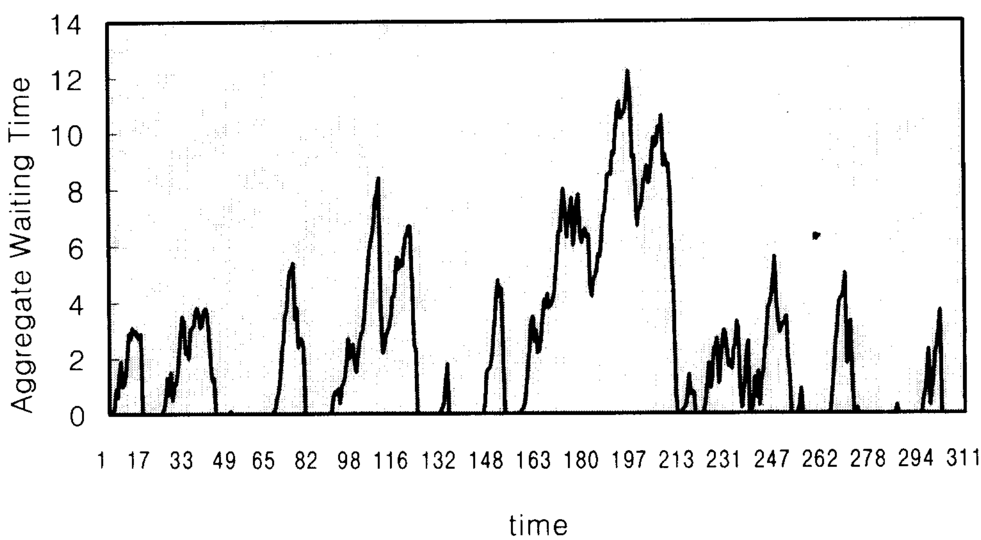

Figure 8. Aggregate waiting time as a function of time.

The induction process embodied in the ID3 algorithm generated a sequence of discrete classifications. Given the nature of the ID3 procedure as well as the genetic algorithm, forecasts from these techniques involved a bipolar output.

Since the number and size of each job in the system was known at any time t, the delay to be experienced by a new job could be easily determined. In other words, w_t was known at each period t.

On the other hand, the arrival time and size of the subsequent job would still be unknown. Hence w_{t+1} could only be estimated. In this context, the forecasting task was to determine whether or not the expected value of w_{t+1} would exceed the current value of w_t. That is, whether

$$E(w_{t+1}) \geq w_t \qquad \text{(Eq. 1)}$$

where E denotes the expectation operator. If Eq. (1) held, then the output was 1; otherwise it was 0. In other words, the predictive task was to anticipate whether the total waiting time at the next period would be longer or shorter than at time t. For consistency, the outputs from the neural network and CBR were also converted into binary form.

For the predictive task, the input vector consisted of a contiguous list of 4 lagged variables. In other words, the objective at each period t was to forecast w_{t+1} based on observed values w_t, w_{t-1}, and w_{t-2}. The data set was partitioned into three consecutive samples: Segments A, B, and C. For the first stage of the investigation, the elementary learning techniques were trained on Segment A and tested on Segment B.

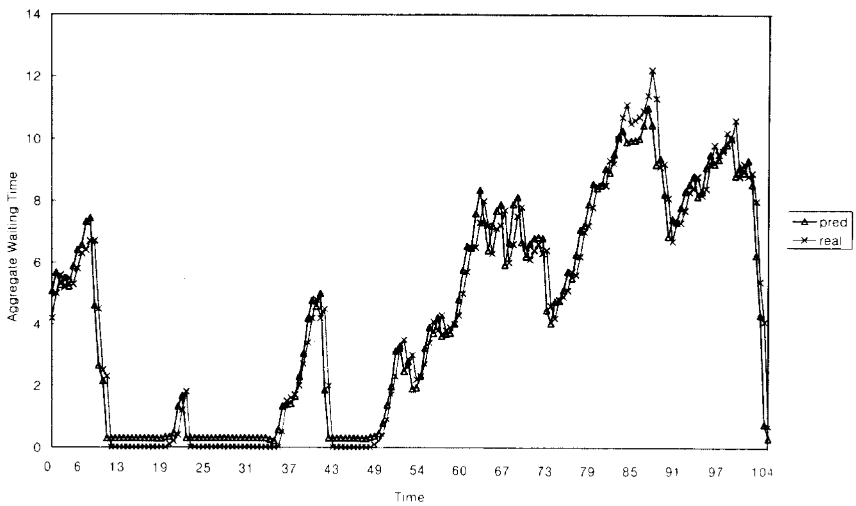

Figure 9. Actual aggregate waiting time versus forecasts using NN.
The architecture involved 3 input nodes, 5 hidden nodes, and 1 output node.

Figure 9 presents a comparison of actual observations against forecasts from the neural network. A similar set of results is shown for CBR in Figure 10. Moreover, the induction algorithm produced a decision tree which is portrayed in Figure 11.

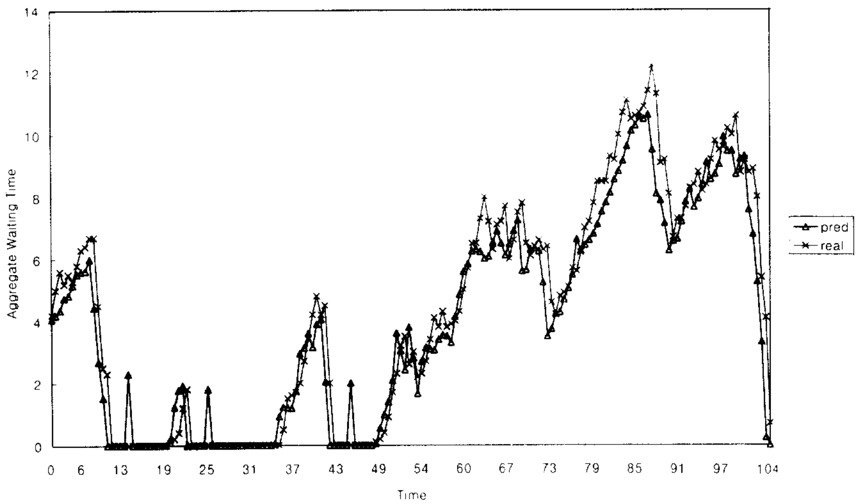

Figure 10. Actual aggregate waiting time versus forecasts using CBR.

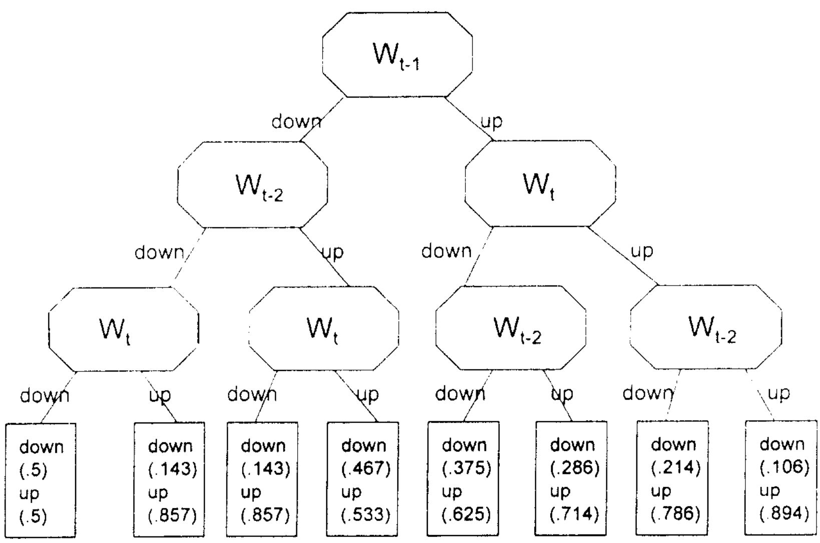

Figure 11. Induction tree by ID3 algorithm.

The elementary algorithms were compared against the metric of hit rate: the proportion of correct forecasts. According to this measure, CBR and BPN outperformed the induction algorithm.

Consequently, CBR and BPN were selected as the first stage of the integrated architecture. The second stage involved induction, based on the data and outputs from the first stage of the investigations.

For the integrated architecture, the inductive module was trained on the data and results associated with Segment B of the data set. The objective of the second stage was to ascertain whether the output from CBR would be more accurate than from BPN, and vice versa.

The training and testing procedures are depicted in Figure 12. After the training phase, the second-stage module of the integrated architecture produced a decision tree as depicted in Figure 13.

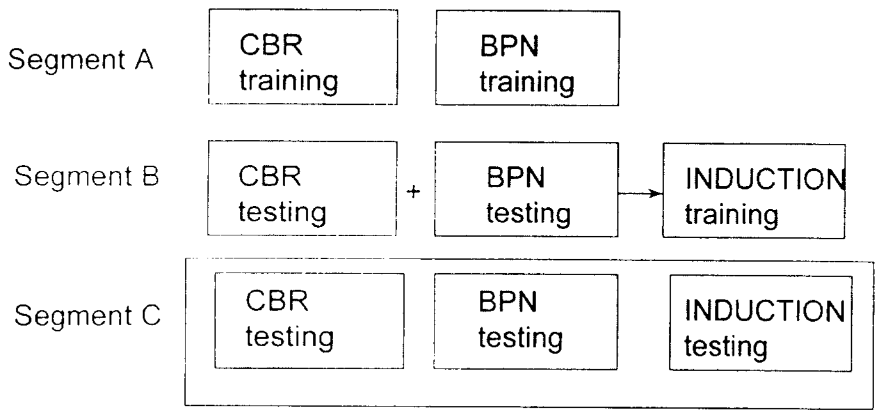

Figure 12. Segmentation of data sets to investigate second-order learning.

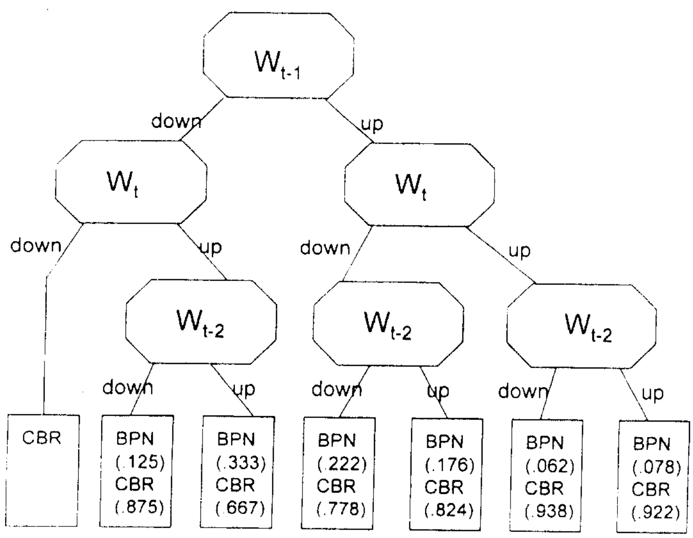

Figure 13. Induction tree generated by second-order learning.

Finally, all 4 architectures were tested against Segment C of the data set. The graphic results in Figure 14 indicate that the induction algorithm performed the worst, while the integrated architecture bested the others. The hit rates are also presented numerically in Table 2.

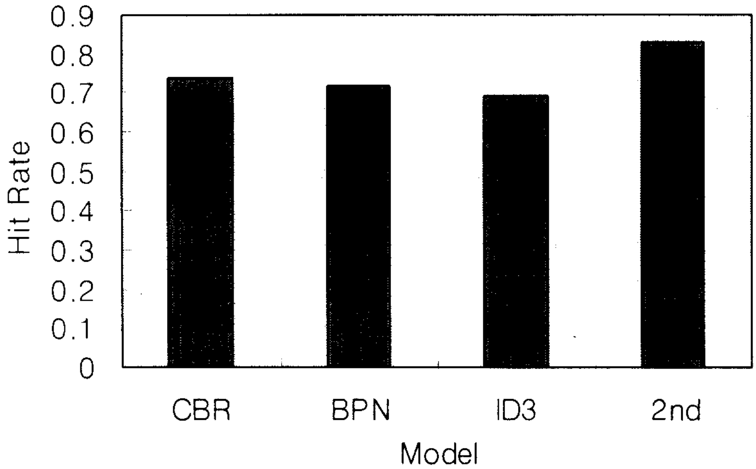

Figure 14. Hit rates resulting from forecasts of the aggregate waiting time for Segment C of the training data.

The collection of pairwise t-tests in Table 3 reveals that the superiority of the integrated method was statistically significant. Moreover, the other 3 techniques did not differ from each other in a statistically convincing fashion.

Table 2. Comparison of predictive methods.

Method	Hit Rate
CBR	0.737
BPN	0.719
Induction	0.690
Second-order	0.831

Table 3. Pairwise tests for proportions using the metric of hit rate in predicting the aggregate waiting time. The first entry in each cell presents the z-value while the significance level is shown in parentheses.

	BPN	ID3	2nd
CBR	0.365	0.957	-2.120
	(0.716)	(0.339)	(0.034)
BPN		0.593	-2.478
		(0.554)	(0.014)
ID3			-3.058
			(0.002)

CONCLUSIONS

This paper has presented a comparative study of knowledge-based methods for predicting the dynamic behavior of complex systems incorporating chaotic and random characteristics. The approach was validated through a case study involving video-on-demand.

The neural network, case based reasoning, and inductive methods yielded no significant differences among themselves. However, second-order learning bested the others at a statistically significant level.

This result was especially remarkable given that the second-order architecture employed the inductive procedure, which had yielded the poorest performance among elementary techniques during the first round of experiments. The effectiveness of the second-order architecture indicates that superior results can emerge through a synergistic interaction of inferior components.

In brief, the results indicate that the complexity inherent in developing a learning system for practical applications can be addressed by the judicious use of a spectrum of methodologies from data mining. Moreover, an integrated method involving second-order learning can significantly outperform more elementary approaches.

REFERENCES

1. Grossberg, S., "Classical and Instrumental Learning by Neural Networks." *Progress in Theoretical Biology,* Vol. 3, Academic Press, NY, (1974), 51-141.
2. Hopfield, J. J., "Neural Networks and Physical Systems with Emergent Collective Computational Abilities." *Proc. Nat. Acad. Sciences USA,* Vol. 79, No. 8, (1982), 2554-2558.
3. Kim, S. H., "Learning Systems for Process Automation through Knowledge Integration." *Proc. Second World Congress on Expert Systems,* Lisbon, Portugal, (1994a).
4. Kim, S. H., *Learning and Coordination,* Dordrecht, Netherlands: Kluwer, 1994b.
5. Kim, S. H., "Integrated Knowledge Discovery for Investment prediction: Critical Issues and System Design Implications." Work Paper, Management Information Systems, KAIST, Seoul, Korea, 1995a.
6. Kim, S. H., "Knowledge Based Prediction of Nonlinear Phenomena." Work Paper, Management Information Systems, KAIST, Seoul, Korea, 1995b.
7. Kim, Y. K., J. Y. Kim and S. S. Kang, "A Tabu Search For Designing A Non-Hierarchical Video-On-Demand Network Architecture." *Proc. 20th Int. Conf. on Computers & Industrial Engineering,* Kyongju, Korea, Vol. 2, (1996), 245-48.

8. Kim, S. H. and H. S. Oh., "A Multistrategy Learning System to Support Predictive Decision Making." *J. of Financial Studies*, Vol. 3, No. 2, (1996), 267-279.
9. Lorentz, E. N., "Deterministic non-periodic flow.", *J. Atmos. Sci.*, Vol. 20, (1963), 130-141.
10. Petit, G. H., D. Deloddere and W. Verbiest, "Bandwidth Resource Optimization in Video-On-Demand Network Architectures." *IEEE GLOBECOM* '94, (1994), 91-97.
11. Porter, B. W., E. R. Bareiss and R.C. Holte, "Concept Learning and Heuristic Classification in Weak-Theory Domains." *Artificial Intelligence*, Vol. 45, No. 1-2, (1990), 229-63.
12. Quinlan, J. R., "Induction of Decision Trees." *Machine Learning*, Vol. 1, (1986), 31-106.
13. Rumelhart, D. E., G. E. Hinton and R. J. Williams, "Learning Internal Representations by Back Propagation." In D. E. Rumelhart, J. L. McClelland and the PDP Research Group, *Parallel Distributed Processing*, Vol. 1, M.I.T. Press, Cambridge, MA, 1986.
14. Sejnowski, T. J., "Parallel Networks that Learn to Pronounce English Text", *Complex Syst.*, Vol. 1, (1987), 145-168.
15. Shannon, C. E. and W. Weaver, *The Mathematical Theory of Communication*, University of Illinois, Urbana, 1949.
16. Shaw, M. J. and J. A. Gentry, "Inductive Learning for Risk Classification." *IEEE EXPERT*, (1990), 47-53.
17. Sycara, K. and D. Navinchandra, "Index Transformation Techniques for Facilitating Creative Use of Multiple Cases." *Proc. 12th Int. Joint Conf. on AI*, Morgan Kaufman, Los Altos, CA, (1991), 347-52.