MEASURMENT BASED TRAFFIC PREDICTION USING FUZZY LOGIC

Qingshan Jiang, * Raj Srinivasan, ** Dean Slonowsky
Manifold Data Mining Incorporated,
501 Alliance Avenue, Suite 205, Toronto, Ontario, Canada M6N 2J1
**Department of Mathematics & Statistics,*
University of Saskatchewan, Saskatoon, Saskatchewan, Canada S7N 5E6
***Department of Statistics & Institute of Industrial Mathematical Sciences,*
University of Manitoba, Winnipeg, Manitoba, Canada R3T 2N2
jiang@manifolddatamining.com, raj@math.usask.ca, dean_slonowsky@umanitoba.ca

Abstract

We investigate algorithms, based on the theory of fuzzy logic systems, which uses on-line traffic measurements to adaptively learn packet arrival patterns, leading to "model-free" traffic prediction. The main predictor, and several novel variants thereof are developed and tested on diverse data streams. For autoregressive-type traffic sources, our algorithms are on par with standard time-series predictors. More significantly, our algorithms provide reliable predictions for highly variable, non-stationary MPEG data; a situation where standard methods are poorly suited.

Keywords: *traffic prediction; fuzzy logic; clustering techniques; learning algorithms.*

1 Introduction

Consider the *time series*, $\{t_k : k = 0, 1, 2, \ldots\}$ where t_k represents the total number of data packets (or bits) generated by a traffic source during the k^{th} time interval. In a high-speed network, critical functions such as traffic shaping, policing, call admission control and bandwidth allocation require an accurate description of the various traffic sources. Indeed, if one can characterize the data streams reasonably well, one can make reliable predictions of future values of t_k on the basis of packet arrival histories. By incorporating such a prediction mechanism, a network can "pre-adapt" to significant changes before the changes happen, hence greatly improving the overall performance of the network.

There are a variety of situations where data streams follow well-known *statistical models* (see [5] and [10], for example). Although driven by random processes, such time series have sufficient structure which, when properly identified, lends itself to dependable traffic predictions. See [1] for a detailed treatise on standard prediction procedures in these cases.

However, today's networks are increasingly faced with traffic which may exhibit a host of complex characteristics such as *long-range dependence* and *non-stationarity* (see [2]). In cases such as these, the task of identifying an underlying model (for the purpose of prediction) through "classic" (e.g., statistical) means is daunting and often times impractical.

For this reason, we investigate adaptive, "model-free" traffic characterization methods which rely only on on-line measurements. These mechanisms "learn" from packet arrival histories; adapting to, and recognizing signs of future change in packet arrival patterns.

In Section 2, we begin by describing the fuzzy logic systems used to approximate the underlying "predictor function". Included is an "adaptive version" which uses nearest-neighborhood clustering to reduce computation. The adaptive version was investigated earlier by Pang *et al.* [6]. Beyond [6], we design, incorporate and test a novel method for training the "dimension parameter" in the predictor.

In Section 3, we test our fuzzy traffic predictors on various traces of real and simulated data. When the traces are such that standard time series methods are known to be reliable, our fuzzy methods yield predictions of comparable quality. The positive effect of training n is clearly demonstrated. In situations where standard time series methods tend to fail, our predictors continue to yield impressive results.

Proceedings of the 2002 IEEE Canadian Conference on Electrical & Computer Engineering
0-7803-7514-9/02/$17.00 © 2002 IEEE

2 Fuzzy Logic Systems

In this section, we define a special class of *fuzzy logic systems*; the key ingredient in our fuzzy traffic predictors. In general, fuzzy logic systems provide a natural means by which to approximate the "dynamics" of a system which can be observed, but is too complex to be directly modeled by classical analytical techniques.

2.1 Optimal fuzzy logic systems

Suppose we are given N *input-output pairs*, $(\mathbf{x}^l, y^l)$, $l = 1, \ldots, N$, representing N observations of a non-linear "black-box"-type system with an n-dimensional input, $\mathbf{x}^l = \langle x_1^l, \ldots, x_n^l \rangle \in \mathcal{R}^n$ and a one-dimensional output $y^l \in \mathcal{R}$. The *optimal fuzzy logic system* (OFLS), $f : \mathcal{R}^n \to \mathcal{R}$ determined by these pairs has the functional form

$$f(\mathbf{x}) = \frac{\sum_{l=1}^{N} y^l \exp\left(-\frac{|\mathbf{x}-\mathbf{x}^l|^2}{\sigma^2}\right)}{\sum_{l=1}^{N} \exp\left(-\frac{|\mathbf{x}-\mathbf{x}^l|^2}{\sigma^2}\right)} \qquad (1)$$

where $\mathbf{x} = \langle x_1, x_2, \ldots, x_n \rangle \in \mathcal{R}^n$ and

$$\exp\left(-\frac{|\mathbf{x}-\mathbf{x}^l|^2}{\sigma^2}\right) = \prod_{i=1}^{n} \exp\left(-\frac{|x_i - x_i^l|^2}{\sigma^2}\right).$$

Our choice of these particular fuzzy logic systems—and the core prediction method which follows—is governed by a previous application to a similar problem in [6].

With the proper choice of σ (the *smoothing parameter*), the OFLS in (1) can approximate the underlying dynamics of the system on the set of input-output pairs to any given degree of accuracy, ϵ. In particular, from [7] we have

Theorem *For any $\epsilon > 0$, there is a $\sigma^* > 0$ such that the fuzzy logic system in (1) with $\sigma = \sigma^*$ has the property: $|f(\mathbf{x}^l) - y^l| < \epsilon$ for all $l = 1, \cdots, N$.*

Comments. (a) If the sole purpose of this fuzzy logic system is to match all given input-output pairs to arbitrary accuracy, then the smaller the value of σ, the better. Indeed, as $\sigma \downarrow 0$, $f(\mathbf{x}^l) \to y^l$ for each $l = 1, \ldots, N$. However, since the intendend use of these fuzzy logic systems is to predict future traffic intensities, the parameter σ (which can be trained) should be large enough so that the influence of each component of the input vector is reasonably significant.

(b) The above class of OFLSs can be described as those fuzzy logic systems with *singleton fuzzifiers, product inference, centroid defuzzifier* and *Gaussian membership functions.* For the definitions of these terms, and a discussion of general fuzzy logic systems, see [7].

2.2 The main traffic predictor

Fix a "step-ahead" value, L. Assume we have observed (and kept record of) the initial trace $\{t_k : k = 0, 1, \cdots, M-1\}$ of packet arrivals. Let the *dimension* n denote the number of successive observations required for a reliable description of the packet arrivals L-steps into the future. That is, for each $k = 1, 2, \ldots, M-n-L+1$, assume that packet arrivals, $t_{k+n-2+L}$ in the $(k+n-2+L)^{\text{th}}$ time interval can be adequately described by

$$t_{k+n-2+L} = F_L(t_{k-1}, t_k, \cdots, t_{k+n-2})$$

where $F_L : \mathcal{R}^n \to \mathcal{R}$ is a deterministic but unknown *predictor function.*

To approximate F_L by an OFLS f, we create as many input-output pairs as possible from the initial trace, namely:

$$\mathbf{x}^l = \langle t_{l-1}, t_l, \cdots, t_{l+n-2} \rangle\,; \;\; y^l = t_{l+n-2+L}$$

for $l = 1, 2, \ldots, M-n-L+1$. In this way, the initial M data points are reduced to

$$N = M + 1 - (n + L)$$

input-output pairs.

For $r \geq M$, let $\widehat{t}_r$ denote the predicted value of t_r (number of packets or bits sent in the r^{th} time interval). Using the approximation, $f \approx F_L$ we can make L L-step ahead predictions by taking

$$\widehat{t}_{M+j} = f(\mathbf{x}^{M-n-L+2+j})\,; \;\; j = 0, 1, \cdots, L-1$$

where $\mathbf{x}^{M-n-L+2+j} = \langle t_{M-n-L+1}, \cdots, t_{M-L} \rangle$. Note that the final prediction is

$$\widehat{t}_{M+L-1} = f(t_{M-n}, t_{M-n+1}, \ldots, t_{M-1}),$$

so that all available data is used.

Remark. The proper choice of n and σ is crucial in controlling the prediction error. The smoothing parameter σ, being a continuous quantity, can be trained via a standard *back-propagation scheme* (see [7], [8]). Roughly speaking, we iterate

$$\sigma_{k+1} = \sigma_k - \zeta \cdot \left.\frac{dE}{d\sigma}\right|_{\sigma_k} \;\; ; \;\; k = 0, 1, 2, \ldots$$

until $|\sigma_{k+1} - \sigma_k|$ is sufficiently small. Here, ζ is the *learning rate* (a small number) and E is an *error-of-fit measure* defined in terms of the training data and the OFLS, f in (1). For example, we can use the *sum of squared-errors*,

$$E = E(\sigma) = \sum_{l=1}^{N} (f(\mathbf{x}^l) - y^l)^2.$$

A learning algorithm for the dimension, n will be discussed in Section 2.5.

2.3 An adaptive fuzzy logic system and associated traffic predictor

The OFLS in (1) associates one "fuzzy rule" (corresponding to one summand) to each input-output pair generated by the *training set*, $\{t_k : k = 0, 1, \cdots, M-1\}$. If there is sufficient determinism to imply that similar input vectors will yield sufficiently similar outputs, it is reasonable to associate just one fuzzy rule to an entire class of input-output pairs with similar input vectors. (Determinism is merely used to motivate the following procedure; it is not a requirement on the underlying time series.) In this way, we remove redundancies in the fuzzy rule base.

To determine these "similarity classes", we employ a modified version of the *nearest neighborhood clustering scheme* proposed in [7]:

1) Starting with the first data pair $\{\mathbf{x}^1, y^1\}$, establish a cluster center C^1 at $\mathbf{x}^1$ and set $A_1(1) = y^1$ and $B_1(1) = 1$. Select a reasonably small radius $r > 0$ (see Remark (b)).

2) Consider the second input-output pair, $\{\mathbf{x}^2, y^2\}$. Compute the distance between $\mathbf{x}^2$ and C^1.

   (i) If $|\mathbf{x}^2 - C^1| > r$, then we take $\mathbf{x}^2$ as the center, C^2 of the second cluster and set $A_2(2) = y^2$, $B_2(2) = 1$, $A_1(2) = A_1(1)$ and $B_1(2) = B_1(1)$.

   (ii) Otherwise, let $A_1(2) = A_1(1) + y^2$ and $B_1(2) = B_1(1) + 1$.

3) Suppose that, just prior to classifying the k^{th} input-output pair, $\{\mathbf{x}^k, y^k\}$, there are K_k clusters with centers located at C^l, $l = 1, 2, \cdots, K_k$. Find the nearest cluster center, C^{l_k} to $\mathbf{x}^k$. Then,

   (i) If $|\mathbf{x}^k - C^{l_k}| > r$, establish a new cluster center $C^{K_k+1} = \mathbf{x}^k$ and set $A_{K_k+1}(k) = y^k$, $B_{K_k+1}(k) = 1$ and set $A_l(k) = A_l(k-1)$, $B_l(k) = B_l(k-1)$ for each $l = 1, 2, \cdots, K_k$.

   (ii) Otherwise, set $A_{l_k}(k) = A_{l_k}(k-1) + y^k$, $B_{l_k}(k) = B_{l_k}(k-1)+1$ and set $A_l(k) = A_l(k-1)$, $B_l(k) = B_l(k-1)$ for each $l = 1, 2, \cdots K_k$ with $l \neq l_k$.

3) If $k < N$, return to Step 3.

After all N input-output pairs have been considered, there will be K cluster centers, $C^1, \ldots, C^K$, and coefficients $A_l(N)$ and $B_l(N)$ where $l = 1, 2, \cdots, K$ and $N = M - n - L + 1$. (Note that $K \leq N$.) We define the associated *adaptive fuzzy logic system* (AFLS) by

$$h(\mathbf{x}) = \frac{\sum_{l=1}^{K} A_l(N) \exp\left(-\frac{|\mathbf{x}-C^l|^2}{\sigma^2}\right)}{\sum_{l=1}^{K} B_l(N) \exp\left(-\frac{|\mathbf{x}-C^l|^2}{\sigma^2}\right)}. \qquad (2)$$

As was the case for f in Section 2.2, h approximates the predictor function, F_L. Therefore, we obtain L-step ahead predictions via

$$\widehat{t}_{M+j} = h(\mathbf{x}^{M-n-L+2+j}); \; j = 0, 1, \cdots, L-1$$

where $\mathbf{x}^{M-n-L+2+j} = \langle t_{M-n-L+1}, \cdots, t_{M-L} \rangle$.

Remarks. (a) Given a radius, $r > 0$, if we replace each $\mathbf{x}^l$ in (1) by the center of the cluster to which it belongs, then, gathering like-terms, (1) becomes (2). Indeed, $B_l(N)$ represents the number of input vectors in cluster l whereas $A_l(N)$ represents the sum of outputs of all input vectors in cluster l. Conversely, if r is less than the minimum distance between any two $\mathbf{x}^l$'s, the AFLS in (2) becomes the OFLS in (1).

(b) The radius r determines the complexity of the AFLS. The smaller the radius, the more clusters, resulting in a more sophisticated nonlinear approximation to F_L at the price of more computation.

(c) An appropriate choice of σ and n for an OFLS is not necessarily an appropriate choice for the corresponding AFLS. Therefore, if one switches from an OFLS predictor to an AFLS predictor, both parameters must be retrained, albeit by the same methods. Furthermore, the parameter r can also be trained.

2.4 Making successive predictions

As it stands, given an initial trace of data, the OFLS (or AFLS) can make only L L-step ahead predictions before "running out of data". To make consistently reliable predictions beyond this horizon, we propose two options:

Growing Window: As more data becomes available, construct additional input-output vectors to be used for fitting an updated OFLS/AFLSs. In this way, as time progresses, we base our predictions on a larger segment of packet arrival history.

Sliding Window: Clearly, as sample-size (time) increases, the training set in the growing window scheme will become quite large, significantly slowing the fitting and predicting process. For this reason, we may want to incorporate a *sliding information window* of a fixed size W. That is, even though we will observe

$\{t_0, t_1, \ldots, t_{M-1}\}$ by time $M-1$, we only use the information in the window, $\{t_{M-W}, t_{M-W+1}, \ldots, t_{M-1}\}$ $(W < M)$ to construct an OFLS/AFLS for the purpose of predicting $\widehat{t}_{M+j}$, $j = 0, 1, \ldots, L-1$.

Another advantage of the sliding window scheme is its ability to adapted to situations where the traffic is non-stationary, i.e., not homogeneous in time. Indeed, in such cases, information in the distant past might be misleading, so far as predicting the near future is concerned.

Under either scheme, the parameters n and σ should be periodically retrained. In the growing window scheme, this will accommodate for the ever increasing ensemble of input-output pairs. In the sliding window scheme, this will help the predictor adapt to non-stationarities in the time series.

2.5 Training the parameter n

Since the discrete parameter n is a fundamental quantity describing, to some degree, the evolution of the packet (or bit) stream, it stands to reason that it, too, can be trained using packet arrival histories. In particular, using the observations $\{t_k : k = 0, 1, \ldots, M-1\}$, we can employ the following *grid-based training scheme*:

1) Fix a "reliable" σ (perhaps a previously trained value), and select a grid for n (say, every m^{th} integer between 1 and M; m dependent on computational resources).

2) For each n in the above grid:

   (i) Use the first 90% of the observed data to construct an OFLS (or AFLS).

   (ii) Make L-step ahead predictions of the remaining 10% of data points, using the sliding window scheme if $L < \frac{M}{10}$.

   (iii) Calculate the *noise to signal ratio*, SNR^{-1} over the predictions. (SNR^{-1} is defined in Section 3.)

4) Choose n which minimizes the SNR^{-1}.

Once n has been trained, we retrain σ using 100% of the data and the trained value of n.

3 Simulation and Discussion

In this section, we test the effectiveness of the OFLS and the AFLS traffic predictors. To measure the quality of predictions, we use *root mean-square error*,

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{k\in T}\left(\widehat{t}_k - t_k\right)^2}$$

and the *noise to signal ratio*,

$$\text{SNR}^{-1} = \frac{\sum_{k\in T}\left(\widehat{t}_k - t_k\right)^2}{\sum_{k\in T} t_k^2}$$

where T denotes the set of times for which we are making predictions. We now consider two diverse and representative situations.

3.1 Simulated VBR data

In [5], a *continuous-state autoregressive Markov model* was proposed for multiplexed encoded *variable bit rate* (VBR) data. In statistical terms, this model is defined by the one-step recursion,

$$t_k = 0.8781\, t_{k-1} + 0.1108\, w_k \qquad (3)$$

where $\{w_k : k = 0, 1, 2, \ldots\}$ is a sequence of independent normal random variables, with common mean, 0.572 and variance, 1. Each realization of this *stochastic process*, yields a typical data stream. Here, t_k represents the bit rate (in bits/pixel) in the k^{th} frame, where 1 frame represents $\frac{1}{30}$ of a second. Since there are 250 000 pixels per frame, the units of t_k can be converted to bits/frame by multiplying by this factor.

Using a procedure written in the C-language, a trace of such data was simulated. We fit an OFLS and AFLS using a window of W values, $\{t_{300-W}, t_{300-W+1}, \ldots, t_{299}\}$, i.e., $\frac{W}{30}$ seconds of training data, for $W = 100, 200$ and 300. The resulting predictors were then applied, yielding 5 five-step ahead predictions for the bit rates in frames 300 to 304. For the sake of reference, we state the actual (albeit simulated) bit rates:

t_{300}	t_{301}	t_{302}	t_{303}	t_{304}
0.373	0.338	0.342	0.311	0.114

For various fixed values of n, we trained the unknown parameter σ. The effect of varying n, measured in terms of RMSE and SNR^{-1}, are recorded in Tables 1 and 2. In the AFLS, we used the mean plus one standard deviation of the training data as the radius for clustering.

One key observation can be made. While offering comparable prediction results (with respect to RMSE

and SNR^{-1}), the AFLSs in Table 2 are significantly simpler than the corresponding OFLSs. For example, when $W = 100$ and $n = 4$, the AFLS defined in (2) uses $K = 3$ fuzzy rules, whereas the corresponding OFLS in (1) uses $N = W + 1 - (n + L) = 92$ fuzzy rules.

Table 1: Predictions using an OFLS.

W	n	σ	SNR^{-1}	RMSE
100	4	0.036	0.579	0.236
100	10	0.094	0.542	0.228
100	15	0.094	0.607	0.241
100	20	0.094	0.730	0.265
200	4	0.025	0.404	0.197
200	10	0.113	0.507	0.221
200	15	0.116	0.590	0.238
200	20	0.116	0.587	0.237
300	4	0.020	0.219	0.145
300	10	0.107	0.554	0.231
300	15	0.116	0.588	0.238
300	20	0.116	0.586	0.237

Table 2: Predictions using an AFLS.

W	n	K	σ	SNR^{-1}	RMSE
100	4	3	0.072	0.588	0.238
100	10	9	0.748	0.748	0.268
100	15	18	0.423	0.219	0.145
100	20	29	0.411	0.127	0.111
200	4	3	0.362	0.493	0.218
200	10	9	0.228	0.492	0.217
200	15	30	0.267	0.514	0.222
200	20	48	0.127	0.333	0.179
300	4	3	0.203	0.368	0.188
300	10	13	0.679	0.455	0.209
300	15	38	0.429	0.440	0.206
300	20	76	0.347	0.397	0.195

Since this data originates from a stationary statistical model, it provides a good opportunity to make comparisons between the performance of our fuzzy predictors and established statistical techniques. To this end, using the same trace used in Tables 1 and 2, we applied the flexible ARAR forecasting technique using the ITSM software accompanying [1]. The RMSE for the 1-step ahead predictions of t_{300} to t_{304} are presented in Table 3. Unlike the OFLS and AFLS used in Tables 1 and 2, both n and σ were trained. Clearly, there is no significant difference in performance between OFLS, AFLS and ARAR with respect to RMSE. Given the reliability of the ARAR method, this result affirms the effectiveness of our fuzzy predictors when *both* n and σ are trained.

Table 3: Comparison between fuzzy predictors and ARAR forecasts.

method	W	n	K	σ	RMSE
OFLS	100	52	–	0.094	0.110
AFLS	100	20	29	0.411	0.110
ARAR	100	–	–	–	0.134
OFLS	200	85	–	0.116	0.140
AFLS	200	21	52	0.194	0.132
ARAR	200	–	–	–	0.133
OFLS	300	89	–	0.116	0.132
AFLS	300	89	207	0.116	0.132
ARAR	300	–	–	–	0.135

3.2 Real MPEG-1 data

The Star Wars data set, produced by Garrett and Vetterli [3] at Bellcore contains 171 000 frames of MPEG-1 data encoding the movie Star Wars. This can be written as a time series, $\{t_k : k = 0, 1, 2, \ldots, 170\,999\}$ where t_k denotes the number of bits in frame k. Here, there are 24 frames per second.

Unlike the previous example, this time series possesses many properties which make prediction via classical methods extremely difficult. Among other things, it's "bursty", non-stationary, heavy-tailed, and exhibits long-range dependence (see [3]).

Using an $M = 200$ frame training set (8.33 seconds of on-line measurements), we constructed a sliding window version of the OFLS with $n = 100$, and used this OFLS to made 100 ten-step ahead predictions. Figure 1 depicts several aspects of the prediction experiment: (a) The 200 frames of training data; (b) the 100 ten-step ahead predictions (bold/red graph represents actual data; lighter/green graph represents predictions); (c) the graph of *relative errors*,

$$\frac{|\widehat{t}_k - t_k|}{|t_k|}$$

for $k = 200, 201, \ldots, 300$. (Here, $k = 0$ represents a randomly selected "initial frame", not the actual initial frame of the overall data set.) Of particular mention:

- Over the 100 predictions, $\text{SNR}^{-1} = 0.157$.
- In Fig. 1 (a), the strong determinisitic (albeit non-periodic) character of the time series is quite evident.
- In Fig. 1 (c), more than 80% of relative errors are less than 0.5. This is a quite an encouraging result, given the complex features of the data. Indeed, this result clearly demonstrates the fuzzy predictors ability to "learn" from similar (but distinct) patterns embedded in the training data.

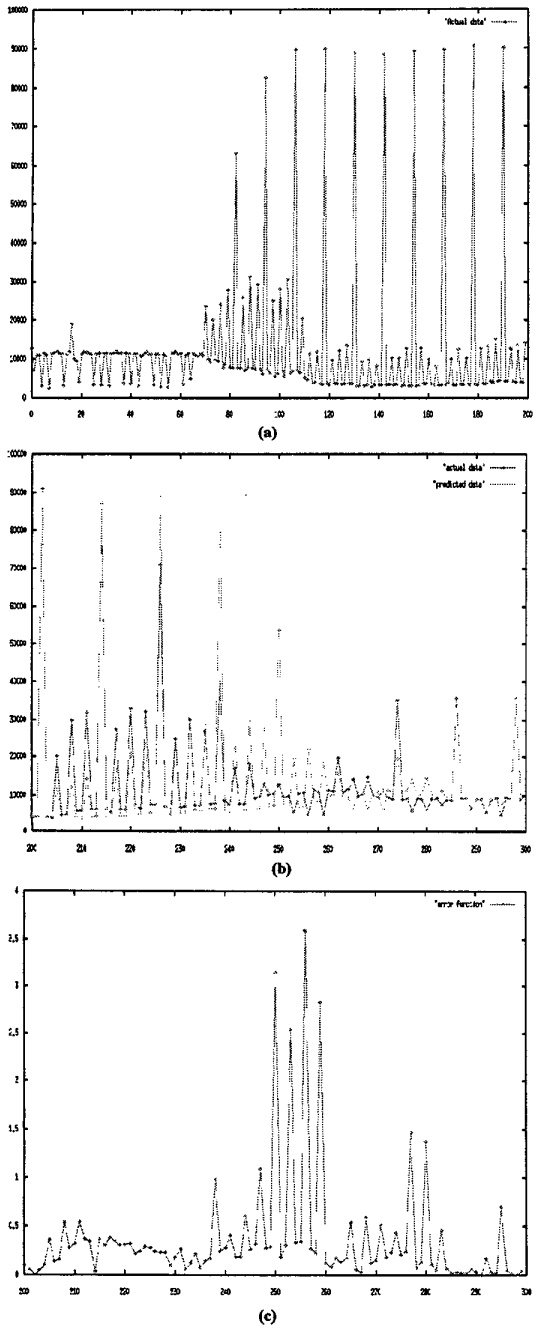

Figure 1: Prediction of Star Wars data.

4 Conclusions

In this paper, we investigated the effectiveness of fuzzy time series predictors in predicting packet arrival patterns. In a situation where classic forecasting methods are known to perform well, our fuzzy predictor, incorporating a novel training scheme for the dimension parameter, performed as well as these methods. In a situation where classic forecasting methods would encounter difficulties, our fuzzy predictor continued to perform well, basing its predictions on past experiences, not on any underlying model. Future work includes additional testing and "fine-tuning" of the various predictors. Our methods will also be applied to the forecasting of other network statistics.

Acknowledgment

This work was partially funded by a grant from MITACS, a Network of Centres of Excellence, and by Nortel Networks. We would like to thank Don Dawson (Carleton University, Canada), Tadeusz Drwiega (Nortel Networks) and James Yan (Nortel Networks) for their many helpful insights throughout this project.

References

[1] P. J. Brockwell, R. A. Davis, *Introduction to Time Series and Forecasting.* Springer 1996.

[2] S. Floyd, V. Paxson, "Difficulties in simulating the internet," to appear in *IEEE/ACM Transactions on Networking.*

[3] M. W. Garrett, W. Willinger, "Analysis, modeling and generation of self-similar VBR video traffic," *Proceedings of ACM SIGCOMM'94*, London, UK; August 1994, pp. 269–280.

[4] J. R. Jang, C. Sun, "Predicting chaotic time series with fuzzy IF-THEN rules," *2nd IEEE International Conference on Fuzzy Systems*, San Francisco, vol. 2, pp. 1079–1084, 1993.

[5] B. Maglaris, D. Anastrassiou, P. Sen, G. Karlsson, J. D. Robbins, "Performance models of statistical multiplexing in packet video communications," *IEEE Transactions on Communications*, vol. 36, no. 7, pp. 834–844, 1988.

[6] Q. Pang, S. Cheng, P. Zhang, "Adaptive fuzzy traffic predictor and its applications in ATM networks," *Proceedings of ICC'98*, pp. 1759–1763, 1998.

[7] L. X. Wang, *Adaptive Fuzzy Systems and Control.* Prentice-Hall 1994.

[8] L. X. Wang, J. M. Mendel, "Back-propagation fuzzy systems as nonlinear dynamic system identifiers," *IEEE International Conference on Fuzzy Systems*, San Diego, pp. 1409–1418, 1992.

[9] L. X. Wang, J. M. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, no. 6, pp. 1414–1427, 1992.

[10] S. Xu, Z. Huang, "A gamma autoregressive video model on ATM networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 7, pp. 138–142, 1988.