5

ATM traffic prediction using FIR neural networks

Z. Fan and P. Mars
School of Engineering, University of Durham,
South Road, Durham, DH1 3LE, UK.
Tel: +44 191 3742559 Fax: +44 191 3743838
Email: (zhong.fan, philip.mars)durham.ac.uk

Abstract

ATM networks support a wide range of multimedia traffic. Various BISDN VBR sources generate traffic at significantly different rates. The traffic can often have time-varying characteristics which are not well understood currently. However, traffic management techniques require traffic parameters that can capture the various traffic characteristics and adapt to the changing network environment. In this paper, we present a novel neural network approach to characterize and predict the complex arrival process. The FIR multilayer perceptron model and its training algorithm are discussed in this paper. It is shown that the FIR neural network can adaptively predict the traffic by learning the relationship between the past and the future traffic variations. Based on the experimental results, we conclude that the FIR neural network is an attractive tool for traffic prediction and hence has an excellent potential for use in some congestion control schemes.

Keywords

ATM, traffic prediction, FIR neural networks

1 Introduction

Asynchronous Transfer Mode(ATM) has been recommended by CCITT as the transfer mode for the future broadband ISDN(BISDN). ATM networks are expected to support a diverse set of applications, such as data, voice and video, each having different traffic characteristics. Accurate characterization of the multimedia traffic is essential in order to develop a robust set of traffic descriptors. Such a set is required by the Usage Parameter Control(UPC) algorithm for traffic enforcement and the Connection Admission

D. D. Kouvatsos (ed.), *ATM Networks*

Control(CAC) algorithm for bandwidth allocation utilizing the statistical multiplexing gain. However, for the time being, there are no comprehensive measurements that permit designers to satisfactorily address the characteristics of various communication services in a realistically accurate manner. This is especially true for Variable Bit Rate(VBR) traffic.

During the duration of a connection, the period at which a source generates traffic is referred to as an *active* period, whereas a *silent* period corresponds to the time between the active periods during which no traffic is generated. Traffic generated by a VBR source either alternates between the active and silent periods, or is a continuous bit stream with varying rates. This traffic is highly bursty and correlated(in comparison to a Poisson process). Burstiness can be defined by the ratio of the peak bit rate to average bit rate or the squared coefficient of variation of the interarrival times of cells, c_1^2(variance divided by the square of the mean). For example, c_1^2 for the packet arrival process from a single voice source is 18.1, while c_1^2 for a Poisson process is 1 [Sriram 86, Heffes 86]. Although the aggregate packet arrival process with many components does behave like a Poisson process over relatively short time intervals, under heavy loads the congestion in the multiplexer is determined by the behaviour of the arrival over much longer time intervals, where it does not behave like a Poisson process. Accordingly, characterization of traffic from VBR sources is very difficult.

As mentioned above, the congestion control schemes(e.g., CAC and UPC) in ATM networks require specific knowledge of the statistical behaviour of the input traffic declared via its traffic descriptors. Parameters such as peak bit rate, average bit rate, and burst length are often used as a simple set of parameters characterizing the traffic. More complicated second-order time domain parameters(e.g., IDI, IDC) are also used to capture the burstiness and the correlation properties of the arrival stochastic process especially those of VBR video and voice sources [Habib 92]. In [Heffes 86], the aggregate arrival process from N voice sources is approximated by a nonrenewal process, i.e.,a two-state Markov Modulated Poisson Process(MMPP). In [Daigle 86], very complex mathematical models such as semi-Markov process and continuous-time Markov chain are used to characterize the voice traffic. Traffic descriptors using simple parameters will not accurately characterize very rapid changes in the bit rate time variations of the traffic over short intervals and often ignore the bursty nature of the traffic. On the other hand, those mechanisms using more sophisticated parameters are computationally expensive and impractical.

To solve this problem, a neural network based traffic prediction approach is proposed in this paper. The neural network can predict the bit rate variations of a complex stochastic process and capture the probability density function(pdf) of the traffic. It is shown that the neural prediction is accurate enough to characterize the actual traffic and can be used in policing and CAC functions. It can also be used in some feedback control schemes. It has been argued that traditional reactive congestion control is not suitable for ATM networks due to the effects of high-speed channels. Recently, Amenyo et al. [Amenyo 91] have proposed a new congestion control scheme called proactive control. Underlying its feasibility and effectiveness are traffic predictions of correlated input traffic streams into network nodes. These predictions are used to obviate the problem of propagation delays.

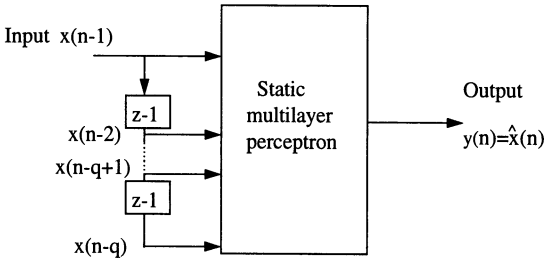

Figure 1: Static multilayer perceptron used as a nonlinear predictor.

So we can apply our neural prediction method to this framework as well.

2 FIR Neural Network

It is well known that neural networks are capable of performing nonlinear mappings between real-valued inputs and outputs. A three-layered feedforward neural network (multilayer perceptron), with sigmoidal units in the hidden layer, is able to approximate an arbitrary nonlinear function to any desired degree of accuracy [Funahashi 89, Hornik 89]. This kind of neural network is trained with the backpropagation(BP) algorithm. One limitation of the standard BP algorithm is that it can only learn an input-output mapping that is *static*. This form of static input-output mapping is well suited for pattern recognition applications, where both the input and output vectors represent *spatial* patterns that are independent of time [Haykin 94].

The standard BP algorithm may also be used to perform nonlinear prediction on a stationary time series [Lapedes 87]. We may use a static multilayer perceptron, as depicted in Figure 1, where the input elements labeled z^{-1} represent unit delays. The input vector $\mathbf{x}$ is defined in terms of the past samples $x(n-1), x(n-2), ..., x(n-q)$ as follows:

$$\mathbf{x} = [x(n-1), x(n-2), ..., x(n-q)]^T \tag{1}$$

where q is the prediction order. Thus the scalar output $y(n)$ of the multilayer perceptron equals the one-step prediction $\hat{x}(n)$, as shown by

$$y(n) = \hat{x}(n) \tag{2}$$

The actual value $x(n)$ of the input signal represents the desired response.

However, if we want to capture the *dynamic* properties of the time-varying signals, we have to extend the design of a multilayer perceptron so as to represent *time* in it. One of the methods is the so-called Time Delay Neural Network(TDNN), which was first used in

[Lang 88] to perform speech recognition. The TDNN is a multilayer feedforward network in which the ouputs of a layer are buffered several time steps and then fed fully connected to the next layer. It was devised to capture explicitly the concept of time symmetry as encountered in the recognition of an isolated phoneme using a spectrogram.

The TDNN topology is in fact embodied in a multilayer perceptron in which each synapse is represented by a Finite Impulse Response (FIR) filter. This latter neural network is referred to as a FIR multilayer perceptron, which can be trained with an efficient algorithm called *temporal backpropagation* [Wan 94]. It can be shown that the TDNN and the FIR network are functionally equivalent. However, the FIR network is more easily related to a standard multilayer network as a simple temporal or vector extension. The FIR representation also leads to a more desirable adaptation scheme. So in this paper, we adopt this kind of FIR network as our traffic predictor.

2.1 FIR Network Model

As mentioned above, the traditional model of a multilayer perceptron forms a static mapping; there are no internal dynamics. A modification of the basic neuron is accomplished by replacing each synaptic weight by a FIR linear filter. By FIR we mean that for an input excitation of finite duration, the output of the filter will also be of finite duration. For this filter, the output $y(k)$ equals a weighted sum of past delayed values of the input:

$$y(k) = \sum_{n=0}^{T} w(n)x(k-n) \tag{3}$$

On the basis of Eq. 3, we may formulate the model of a FIR neuron as follows. Let $w_{ji}(l)$ denote the weight connected to the lth tap of the FIR filter modeling the synapse that connects the output of neuron i to neuron $j (i = 1, 2, ..., p)$. The index l ranges from 0 to M, where M is the total number of delay units built into the design of the FIR filter. Let $y_j(n)$ denote the ouput signal of neuron j and $x_i(n)$ the input signal. Hence, we have

$$v_j(n) = \sum_{i=1}^{p} \sum_{l=0}^{M} w_{ji}(l)x_i(n-l) - \theta_j \tag{4}$$

$$y_j(n) = \varphi(v_j(n)) \tag{5}$$

where $v_j(n)$ is the net activation potential of neuron j, θ_j is the externally applied threshold and $\varphi(\cdot)$ is the nonlinear activation function of the neuron.

We may rewrite Eq. 4 and Eq. 5 in matrix form by introducing the following definitions for the state vector and weight vector for synapse i, respectively:

$$\mathbf{x}_i(n) = [x_i(n), x_i(n-1), ..., x_i(n-M)]^T \tag{6}$$

$$\mathbf{w}_{ji} = [w_{ji}(0), w_{ji}(1), ..., w_{ji}(M)]^T \tag{7}$$

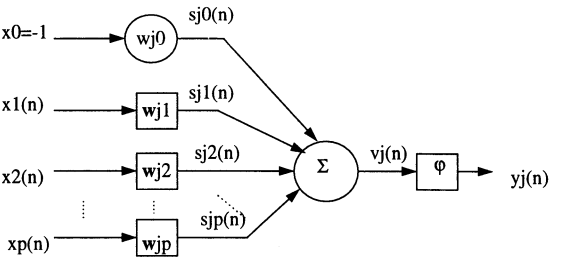

Figure 2: Dynamic model of a neuron, incorporating synaptic FIR filters.

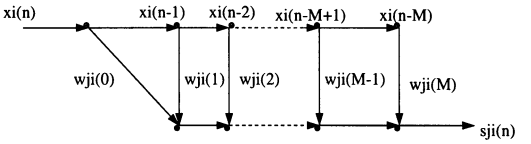

Figure 3: Signal-flow graph of a synaptic FIR filter.

We may thus express the output $y_j(n)$ of neuron j by the following equation:

$$y_j(n) = \varphi(\sum_{i=1}^{p} \mathbf{w}_{ji}^T \mathbf{x}_i(n) - \theta_j) \tag{8}$$

This FIR model of a single artificial neuron is shown in Figure 2, where the weight w_{j0} connected to the fixed input $x_0 = -1$ represents the threshold θ_j. The signal-flow graph representation of a FIR filter is shown in Figure 3.

We may construct a multilayer perceptron whose hidden and output neurons are all based on the above FIR model. Such a neural network structure can be referred to as a FIR multilayer perceptron. The difference between the FIR multilayer perceptron and the standard one is that the static forms of the synaptic connections between the neurons in the various layers of the network are replaced by their dynamic versions (i.e., scalars are replaced by vectors and multiplications by vector products).

2.2 Temporal Backpropagation Learning

Given an input sequence $x(k)$, the network produces the output sequence $y(k) = \mathcal{N}[W, x(k)]$, where W represents the set of all filter coefficients in the network. Define the instantaneous error $e^2(k) = \|d(k) - y(k)\|^2$ as the squared Euclidean distance between the network output $y(k)$ and the desired output $d(k)$. Therefore the objective of training corresponds to minimizing over W the cost function:

$$C = \frac{1}{2}\sum_{k=1}^{K} e^2(k)$$

where the sum is taken over all K points in the training sequence. In [Wan 94], an algorithm called temporal backpropagation is proposed to minimize C. The weight-update equation is shown by the following pair of relations:

$$\mathbf{w}_{ji}(k+1) = \mathbf{w}_{ji}(k) - \eta \frac{\partial C}{\partial v_j(k)} \frac{\partial v_j(k)}{\partial \mathbf{w}_{ji}(k)} = \mathbf{w}_{ji}(k) + \eta \delta_j(k) \mathbf{x}_i(k) \tag{9}$$

$$\delta_j(k) = \begin{cases} e_j(k)\varphi'(v_j(k)), & \text{neuron } j \text{ in the output layer} \\ \varphi'(v_j(k)) \sum_{m \epsilon \mathcal{A}} \mathbf{\Delta}_m^T(k) \mathbf{w}_{mj}, & \text{neuron } j \text{ in a hidden layer} \end{cases} \tag{10}$$

where η is the learning-rate parameter, $\mathcal{A}$ is defined as the set of all neurons whose inputs are fed by neuron j in a forward manner and $\mathbf{\Delta}_m(k)$ is defined as follows:

$$\mathbf{\Delta}_m(k) = [\delta_m(k), \delta_m(k+1), ..., \delta_m(k+M)]^T \tag{11}$$

It is obvious that the above equations represent a *vector generalization* of the standard backpropagation algorithm. In fact, if we replace the input vector $\mathbf{x}_i(n)$, the weight vector $\mathbf{w}_{mj}$, and the local gradient vector $\mathbf{\Delta}_m$ by their scalar counterparts, the temporal backpropagation algorithm reduces to the standard backpropagation for static networks. To calculate $\delta_j(k)$ for a neuron j located in a hidden layer, we filter the δ's from the next layer backwards through the FIR synapses for which the given neuron feeds(see Figure 4). Thus δ's are formed not by simply taking weighted sums, but by backward filtering. For each new input and desired response vector, the forward filters are incremented one time step and the backward filters one time step. The weights are then adapted on-line at each time increment.

Temporal backpropagation preserves the symmetry between the forward propagation of states and the backward propagation of error terms. The sense of parallel distributed processing is thereby maintained. Furthermore, each unique weight of synaptic filter is used only once in the computation of the δ's; there is no redundant use of terms experienced in the instantaneous gradient model.

However, careful inspection of the above equations reveals that the calculations for the $\delta_j(k)$'s are noncausal. We may formulate the causal form of the temporal backpropagation algorithm by a simple reindexing:

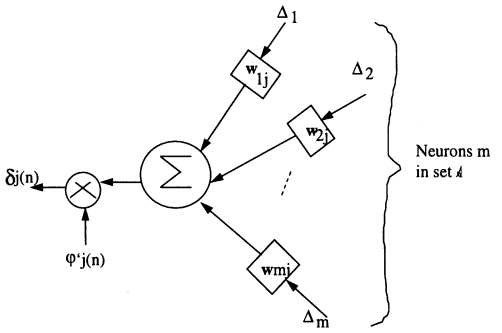

Figure 4: Backpropagation of local gradients through an FIR multilayer perceptron.

For neuron j in the output layer, compute

$$\mathbf{w}_{ji}(k+1) = \mathbf{w}_{ji}(k) + \eta\delta_j(k)\mathbf{x}_i(k) \tag{12}$$

$$\delta_j(k) = e_j(k)\varphi_j'(k) \tag{13}$$

For neuron j in a hidden layer, compute

$$\mathbf{w}_{ji}(k+1) = \mathbf{w}_{ji}(k) + \eta\delta_j(k-lM)\mathbf{x}_i(k-lM) \tag{14}$$

$$\delta_j(k-lM) = \varphi'(v_j(k-lM)) \sum_{m\epsilon A} \mathbf{\Delta}_m^T(k-lM)\mathbf{w}_{mj} \tag{15}$$

where M is the total synaptic filter length, and the index l identifies the hidden layer in question. Specifically, $l = 1$ corresponds to one layer back from the output layer; $l = 2$, two layers back from the output layer; and so on.

3 ATM Traffic Prediction Using FIR Networks

Neural networks have adaptation capability that can accommodate nonstationarity. Their generalization capability makes them flexible and robust when facing new and noisy data patterns. Once the training is completed, a neural network can be computationally inexpensive even if it continues to adapt on-line. Actually, neural networks have been used in call control, switch control and routing [Morris 94, Hiramatsu 90]. Here we use a FIR neural network as a multimedia traffic predictor in ATM networks. The role of the neural network is to capture the unknown complex relation between the past and future values of the traffic.

3.1 Predictor Training Configuration

Consider a scalar time series denoted by $x(n)$, which is described by a nonlinear regressive model of order q as follows [Haykin 94]:

$$x(n) = f(x(n-1), x(n-2), ..., x(n-q)) + \varepsilon(n) \tag{16}$$

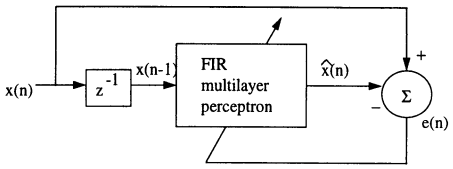

Figure 5: Training scheme of the FIR network.

where f is a nonlinear function of its arguments and $\varepsilon(n)$ is a residual. It is assumed that $\varepsilon(n)$ is drawn from a white Gaussian process. The nonlinear function f is unknown, and the only thing that we have available to us is a set of observables: $x(1), x(2), ..., x(N)$, where N is the total length of the time series. We may use a FIR multilayer perceptron as a one-step predictor of some order q to model the time series, as shown in Figure 5. Specifically, the network is designed to make a prediction of the sample $x(n)$, given the past q samples $x(n-1), x(n-2), ..., x(n-q)$, as shown by

$$\hat{x}(n) = F(x(n-1), x(n-2), ..., x(n-q)) + e(n) \tag{17}$$

The nonlinear function F is the approximation of the unknown function f, which is computed by the FIR multilayer perceptron. The actual sample value $x(n)$ acts as the desired response. Hence the FIR multilayer perceptron is trained so as to minimize the squared value of the prediction error:

$$e(n) = x(n) - \hat{x}(n), \qquad q+1 \leq n \leq N. \tag{18}$$

In the neural network literature the above training scheme is referred to as *teacher forcing*, while in the control and signal processing literature, it is referred to as *equation-error adaptation.*

In our application, the FIR multilayer perceptron is designed as a 1-5-1 fully connected feedforward network with 3:3 taps per layer. Selection of these dimensions is based mostly on trial and error. In general, selection of dimensions for neural networks remains an open question in need of further research. The FIR network is trained with the causal form of temporal backpropagation and the *mean-squared error*(MSE) is used as a performance measure. To increase the rate of learning and yet avoid the danger of instability, a momentum term is added to the weight-update equation, i.e.,

$$\Delta \mathbf{w}_{ji}(k) = \alpha \Delta \mathbf{w}_{ji}(k-1) + \eta \delta_j(k) \mathbf{x}_i(k) \tag{19}$$

where α is a positive number called the momentum constant. The learning rate η and momentum constant α are set at 0.1 initially. It has been found that the BP learning algorithm may learn faster when the sigmoidal activation function built into the neuron model of the network is asymmetric than when it is nonsymmetric. So we adopt the hyperbolic tangent activation function in the hidden layer, which is defined by

$$\varphi(v) = a \tanh(bv)$$

where $a = 1.716$ and $b = 2/3$. In some of our experiments, we have also used some heuristics to accelerate the convergence of backpropagation learning through learning rate adaptation [Haykin 94]. Simulations have been performed to obtain the neural network data set for both training and testing(cross-validation) purposes. Since we use the logistic function $\varphi(v) = 1/(1 + \exp(-v))$ for the output neuron, we have to normalize the traffic data so that all the values fall between 0 and 1.

3.2 Traffic Models

In this section, we briefly describe the models for video arrival process and voice arrival process used in our experiments.

3.2.1 Video Arrival Process Model

Video is presented to users as a series of frames in which the motion of the scene is reflected in small changes in sequentially displayed frames. Video frames are generated at a constant rate defined by the playout rate. As the amount of data transmitted per frame varies due to intraframe and interframe coding, video applications generate traffic in a continuous manner at varying rates. Video is a relatively new service in communication networks and its traffic characteristics are not well understood. It is also quite different from voice or data in that its bit streams exhibit various types of correlations between consecutive frames.

The characteristics of the video signal depends primarily on two factors: 1) the nature of the video scene, and 2) the type of VBR coding technique employed(e.g., motion-compensated discrete cosine transform, interframe DPCM, etc.). For the purpose of simplicity, in this paper, we focus on video services with uniform activity level scenes, i.e., the change in the information content of consecutive frames is not significant [Onvural 94]. A typical application of this type is video telephone where the screen shows a person talking. In general, correlations in video services with uniform activity levels last for a short duration and decay exponentially with respect to the time. The simulation model used to generate this kind of video coded traffic is a continuous-state discrete-time stochastic process. A first-order autoregressive(AR) Markov model is proposed in [Maglaris 88], which estimates the bit rate at the nth frame from the bit rate at the $(n-1)$st frame to be

$$\lambda(n) = a\lambda(n-1) + bw(n) \tag{20}$$

where $\lambda(n)$ denotes the bit rate of the nth frame in bits/pixel, a and b are constants and $w(n)$ is a Gaussian random variable with mean m and variance 1. There are about 250000 pixels per frame and 30 frames/s, thus 1 bit/pixel corresponds to 7.5 Mbits/s. The mean $E(\lambda)$, and the autovariance of the bit rate $C(n)$ are equal to

$$E(\lambda) = bm/(1-a) \tag{21}$$

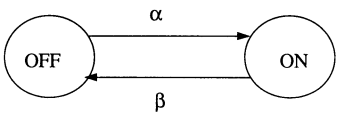

Figure 6: IPP model.

$$C(n) = b^2 a^n/(1-a^2) \tag{22}$$

which can be used to determine the two unknown variables a, b and m:

$$a = 0.8781, \qquad b = 0.1108, \qquad m = 0.572 \tag{23}$$

The model is found to be quite accurate compared with the actual measurements and is suitable for simulation studies.

3.2.2 Voice Arrival Process Model

A voice source alternates between talk spurts(active) and silent periods. To achieve higher resource utilization, a speech activity detection may be used at the VBR voice source so that voice packets are generated only when the source is active, thereby, increasing the transmission efficiency. The correlated generation of voice packets within a call can be modeled by an Interrupted Poisson Process(IPP). In an IPP model, each voice source is characterized by ON(corresponding to talk spurt) and OFF(corresponding to silence duration) periods, which appear in turn. During the ON period, the interarrival times of packets are exponentially distributed(i.e., in a Poisson manner), while no packets are generated during the OFF period. The transition from ON to OFF occurs with the rate β, and the transition from OFF to ON occurs with the rate α(see Figure 6). Hence the ON and OFF periods are exponentially distributed with the mean $1/\beta$ and $1/\alpha$. To specify this model completely, we assume that the packet-generation rate during the active period is 32kbps, the mean talk spurt is $1/\beta = 352$ms and the mean silence period is $1/\alpha = 650$ms.

4 Simulations

In this section, we demonstrate the effectiveness of the neural network used as a traffic predictor. Extensive simulations have been performed. The packet arrival process is generated from packetized video sources or/and packetized voice sources according to the models discussed in the previous section. We have used different data for the training

FIR network	MSE for the traing set	MSE for the test set
1-5-1	0.00414	0.00423
1-10-1	0.00410	0.00415

Table 1: MSE of the experiment for video traffic.

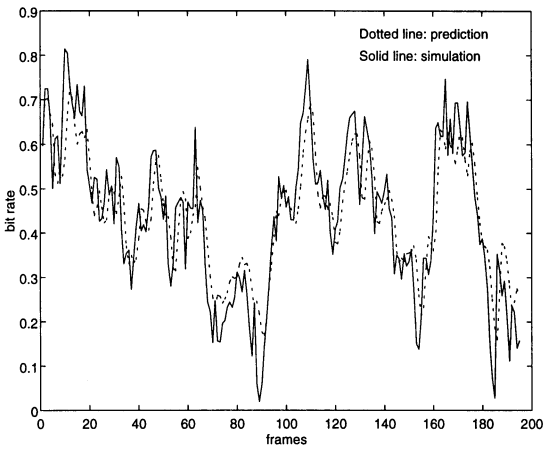

Figure 7: Prediction results for the bit rate of the video traffic.

sets and test sets by choosing different initial values of the arrival process or different seeds of the random number generator. For example, for the video arrival process, we have generated 2000 traffic data elements, starting with $\lambda(0) = 0.6$. The first 400 elements have been chosen to be the training set, and the next 1600 to be the test set. We have also tried a more complicated 1-10-1 network model for the same data sets, but no significant performance improvement is observed. The values of MSE of the above experiment are summarized in Table 1. Other prediction results for the test sets will be reported in the following.

Experiment 1: In this experiment, we use three video sources. The FIR network is used to predict the bit rate of the superposition video arrival process over the next frame. Therefore the lag time is 1/30 sec, which is the frame generation rate. The prediction results are shown in Figure 7 and Figure 8, illustrating that the predicted traffic has almost the same statistical characteristics as those of the actual traffic.

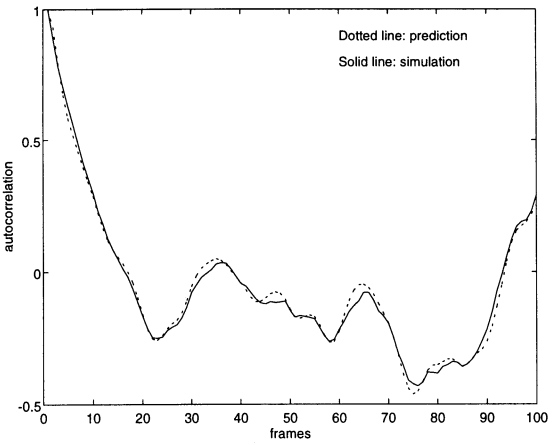

Figure 8: Comparison of the autocorrelation function of the predicted video traffic and the actual one.

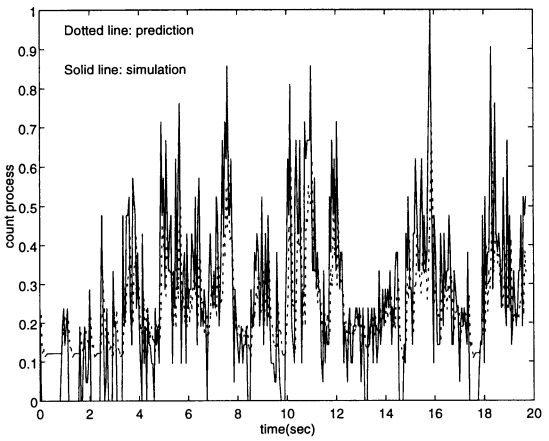

Figure 9: Prediction results for the arrival process of voice sources.

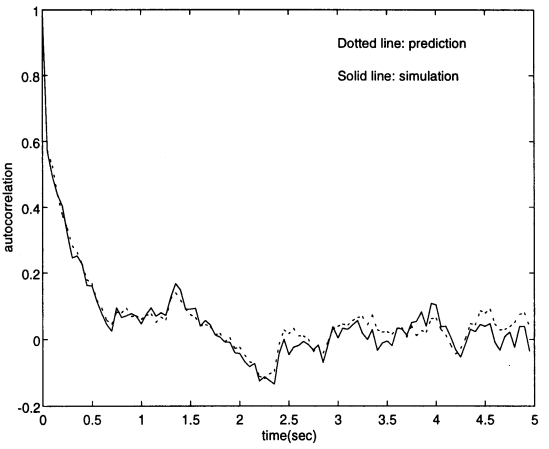

Figure 10: Comparison of the autocorrelation of the number of packet arrivals of the predicted voice traffic and the actual one.

Experiment 2: In this experiment, we use three voice sources. Here the time series $x(n)$ is used to represent the count process $N(0,t)$ which measures the number of packet arrivals in time $(0,t)$. The arrival process is sampled at every sampling period T_s. The choice of the parameter T_s is influenced by the type of the traffic and should guarantee that the used sampled version of the arrival process captures all correlations contained in the actual process. In this application T_s has been found to be 50 ms [Tarraf 94]. Figure 9 and Figure 10 show that the neural network prediction is very close to the actual traffic values.

Experiment 3: In this experiment, one video source and three voice sources are used to generate a heterogeneous superposition arrival process. Prediction results of the count process are shown in Fig. 11 and Fig. 12. It should be pointed out that more training iterations of the neural network are needed in this experiment than in the previous ones, since the heterogeneous traffic is more difficult to characterize.

Experiment 4: Recently, Leland et al. demonstrated that Ethernet local area network traffic is statistically *self-similar* [Leland 93]. To capture this *fractal* behaviour, they proposed to model the traffic using deterministic *chaotic* maps. Chaos is a dynamical system phenomenon in which simple, low order, nonlinear deterministic equations can produce behaviour that mimics random processes. To illustrate the underlying idea, consider a nonlinear map $f(\cdot)$ that descrides the evolution of a state variable $x(n) \in (0,1)$ over discrete time as $x(n+1) = f(x(n))$. The packet generation process for an individual

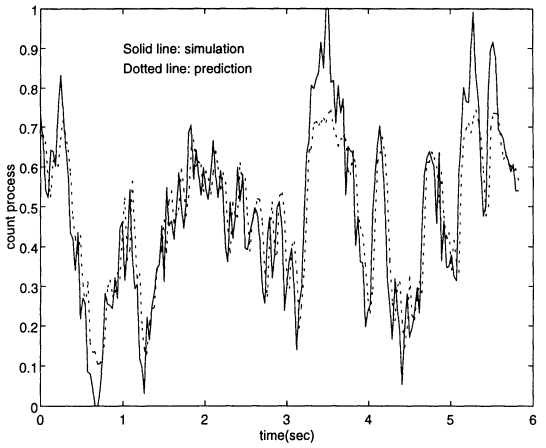

Figure 11: Prediction results for the heterogeneous traffic.

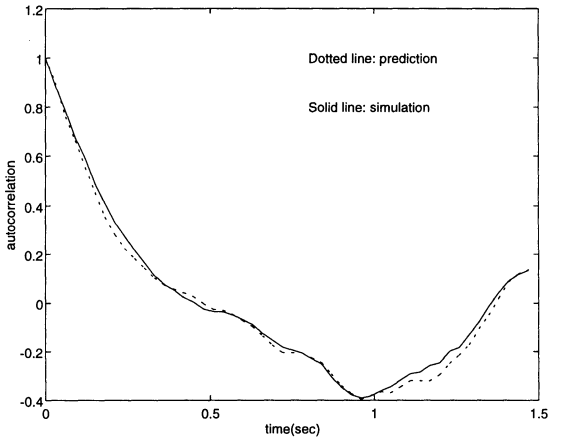

Figure 12: Comparison of the autocorrelation function of the predicted traffic and the actual one in experiment 3.

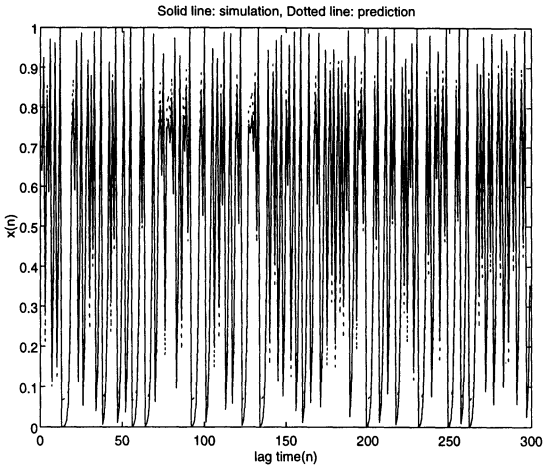

Figure 13: Prediction results for the training set of the chaotic time series.

source can now be modeled by stipulating that the source generates one or no packet at time n depending on whether $x(n)$ is above or below an appropriately chosen threshold. If f is a chaotic map, the resulting packet process can mimic complex packet traffic phenomena. Once an appropriate chaotic map has been derived from a set of traffic measurements, generating a packet stream for an individual source is generally quick and easy. On the other hand, deriving an appropriate nonlinear chaotic map based on a set of actual traffic measurements currently requires considerable guessing and experimenting. Nevertheless, studying arrival streams to queues that are generated by nonlinear chaotic maps may well provide new insight into the performance of queueing systems where the arrival processes exhibit fractal properties.

Here as another experiment, we train the FIR network to perform one-step prediction of a chaotic time series. A chaotic time series generated by the so called logistic map is defined as [Rasband 90]

$$x(n+1) = 4x(n)(1 - x(n)) \tag{24}$$

where the values of $x(n)$ are all in the range $(0, 1)$. The prediction results for the training and test data sets are encouraging, as shown in Figure 13 and Figure 14 respectively.

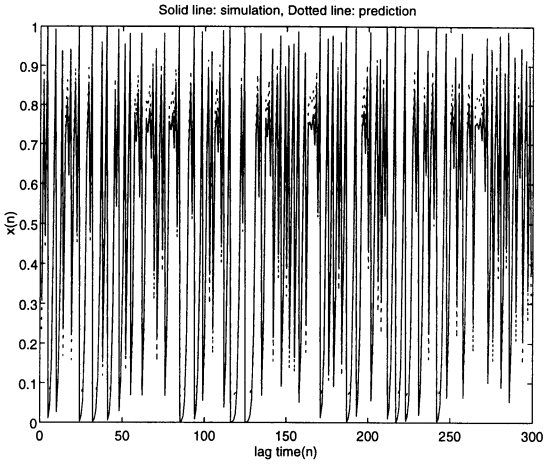

Figure 14: Prediction results for the test set of the chaotic time series.

5 Conclusion

In this paper, we have shown that a FIR network constitutes a powerful tool for use in ATM traffic prediction. The theoretical justification of this approach is that neural networks are capable of approximating any continuous function and perform non-parametric regression. Furthermore, a FIR neural network extends the standard multilayer perceptron to a temporal processing version which is more suitable for modeling of time series. After completing the training phase of the neural network, it can successfully learn the actual pdf of the offered traffic(instead of the approximated simple parameters, such as the peak and mean bit rates). Hence the neural network can be used as an effective traffic descriptor.

In ATM networks, traffic management techniques require traffic parameters that can capture the various traffic characteristics and adapt to the changing network environment. The method based on FIR neural networks can adaptively predict the traffic by learning the relationship between the past and the future traffic variations. Therefore it can be incorporated into traffic control functions in order to achieve better network performance. In [Fan 96], we propose a feedback flow control mechanism based on traffic prediction by FIR neural networks. The predicted traffic patterns in conjunction with the current queue information of the buffer can be used as a measure of congestion. When the congestion level is reached, a feedback signal is sent to sources to reduce their bit rates. Simulation results show that our scheme leads to a much lower cell loss rate than the conventional

feedback control method and hence provides a simple and efficient traffic management for ATM networks.

References

[Amenyo 91] Amenyo, J. T., Lazar, A. A. and Pacifici, G. (1991) Cooperative distributed scheduling for ATS-based broadband networks. CTR Technical Report, Columbia University, New York.

[Daigle 86] Daigle, J. N. and Langford, J. D. (1986) Models for analysis of packet voice communications systems. *IEEE J. Selected Areas in Comm.,* **SAC-4**, 847-855.

[Fan 96] Fan, Z. and Mars, P. (1996) Access flow control for ATM networks using a neural network traffic predictor. to appear in *Proc. 13th IEE Teletraffic Symposium*, Glasgow, UK.

[Funahashi 89] Funahashi, K. (1989) On the approximate realization of continuous mappings by neural networks. *Neural Networks,* **2**, 183-192.

[Habib 92] Habib, I. W. and Saadawi, T. N. (1992) Multimedia traffic characteristics in broadband networks. *IEEE Communications Mag.,* 48-54.

[Haykin 94] Haykin, S. (1994) *Neural Networks.* Macmillan, New York.

[Heffes 86] Heffes, H. and Lucantoni, D. M. (1986) Markov modulated characterization of packetized voice and data traffic and related statistical multiplexer performance. *IEEE J. Selected Areas in Comm.,* **SAC-4**, 856-868.

[Hiramatsu 90] Hiramatsu, A. (1990) ATM communications network control by neural networks. *IEEE Trans. Neural Networks,* **1**, 122-130.

[Hornik 89] Hornik, K., Stinchcombe, M and White, H. (1989) Multilayer feedforward networks are universal approximators. *Neural Networks,* **2**, 359-366.

[Lang 88] Lang, K. J. and Hinton, G. E. (1988) The development of the time-delay neural network architecture for speech recognition. Technical Report CMU-CS-88-152, Carnegie-Mellon University, PA.

[Lapedes 87] Lapedes, A. and Farber, R. (1987) Nonlinear signal processing using neural networks: Prediction and system modeling. Technical Report LA-UR-87-2662, Los Alamos National Laboratory, NM.

[Leland 93] Leland, W. E., Taqqu, M. S., Willinger, W. and Wilson, D. V. (1993) On the self-similar nature of Ethernet traffic. Bellcore Technical Report, NJ.

[Maglaris 88] Maglaris, B., Anastassiou, D., Sen, P., Karlsson G. and Robbins, J. D. (1988) Performance models of statistical multiplexing in packet video communications. *IEEE Trans. Commun.,* **36**, 834-843.

[Morris 94] Morris, R. J. T. and Samadi, B. (1994) Neural network control of communications systems. *IEEE Trans. Neural Networks,* **5**, 639-650.

[Onvural 94] Onvural R. O. (1994) *Asynchronous Transfer Mode Networks,* Artech House, Boston.

[Rasband 90] Rasband, S. N. (1990) *Chaotic Dynamics of Nonlinear Systems,* Wiley, New York.

[Sriram 86] Sriram, K. and Whitt, W. (1986) Characterizing superposition arrival processes in packet multiplexers for voice and data. *IEEE J. Selected Areas in Comm.,* **SAC-4**, 833-846.

[Tarraf 94] Tarraf, A. A., Habib, I. W. and Saadawi, T. N. (1994) A novel neural network traffic enforcement mechanism for ATM networks. *IEEE J. Selected Areas in Comm.,* **SAC-12**, 1088-1095.

[Wan 94] Wan, E. A. (1994) Time series prediction by using a connectionist network with internal delay lines. in *Time Series Prediction*(eds. A. S. Weigend and N. A. Gershenfeld), 195-217, Addison-Wesley.

Biography

Zhong Fan received the BSc and MPhil degrees in electronic engineering from Tsinghua University, Beijing, in 1992 and 1994, respectively. He is now working toward his PhD at University of Durham, UK. His research interests include neural networks, traffic modeling and control of ATM networks.

Philip Mars is Professor of Electronics and Director of the Center for Telecommunication Networks at the University of Durham, UK. He is the coauthor of two research monographs and over 120 published papers. His research interests are in the application of nonsymbolic AI to telecommunications and in network performance modelling and simulation.