Prediction of MPEG-Coded Video Source Traffic Using Recurrent Neural Networks

Aninda Bhattacharya, Alexander G. Parlos, *Senior Member, IEEE*, and Amir F. Atiya, *Senior Member, IEEE*

***Abstract*—Predicting traffic generated by multimedia sources is needed for effective dynamic bandwidth allocation and for multimedia quality-of-service (QoS) control strategies implemented at the network edges. The time-series representing frame or visual object plane (VOP) sizes of an MPEG-coded stream is extremely noisy, and it has very long-range time dependencies. This paper provides an approach for developing MPEG-coded real-time video traffic predictors for use in single-step (SS) and multistep (MS) prediction horizons. The designed SS predictor consists of one recurrent network for I-VOPs and two feedforward networks for P- and B-VOPs, respectively. These are used for single-frame-ahead prediction. A moving average of the frame or VOP sizes time-series is generated from the individual frame sizes and used for both SS and MS prediction. The resulting MS predictor is based on recurrent networks, and it is used to perform two-step-ahead and four-step-ahead prediction, corresponding to multistep prediction horizons of 1 and 2 s, respectively. All of the predictors are designed using a segment of a single MPEG-4 video stream, and they are tested for accuracy on complete video streams with a variety of quantization levels, coded with both MPEG-1 and MPEG-4. Comparisons with *SS* prediction results of MPEG-1 coded video traces from the recent literature are presented. No similar results are available for prediction of MPEG-4 coded video traces and for MS prediction. These are considered unique contributions of this research.**

***Index Terms*—MPEG-coded source traffic, multi-step-ahead prediction, neural networks, neuro-predictors.**

I. Introduction

CURRENTLY, the motivation for predicting video source traffic bit rates arises from at least two important considerations in multimedia networks. These are dynamic bandwidth allocation and quality-of-service (QoS) control of real-time multimedia streams transported over networks that do not offer service guarantees, e.g., best-effort networks, such as internet protocol (IP) networks. It is quite possible that future networked computing needs and applications may bring forward additional circumstances in which source traffic prediction becomes critical.

Efficient and fair utilization of available bandwidth is a topic that has garnered much attention in the networking community. In order to adapt the allocated bandwidth among network end-users dynamically, it is imperative to predict the traffic generated by end-users. Currently, multimedia is responsible for an increasing fraction of traffic over networks and this trend is expected to continue. If algorithms for the prediction of traffic generated by multimedia sources are developed, then they could significantly aid in the design of efficient dynamic bandwidth allocation mechanisms.

There are many problems faced when transporting multimedia streams, such as video or audio, in real-time over networks that offer no service guarantees. The majority of these problems originate from the delay-sensitive nature of multimedia content. Irrespective of the method used to transport media content over an IP network, there is a strict timing sequence that must be used by the decoder during playback. For acceptable playback experience, all relevant packets must be available at the destination for assembly when needed and in the correct sequence. An obvious, and simple, solution to this problem is destination-side buffering and, more recently, edge-caching. The tradeoff of this approach is that the media content is not delivered to the destination in real-time or even in near real-time. Even though many media applications, such as streaming and on-demand video and audio, are tolerant to such large delays in delivery, there are numerous applications that require real-time or near real-time media delivery, such as gaming, conferencing, telephony and teleoperation applications. One of the problems encountered in efforts to design and implement edge-based QoS control in IP network for applications requiring real-time or near real-time content delivery is the need to know the source traffic bit-rate time-series; future values of this time-series are needed.

Most of the reported research in multimedia source traffic prediction has dealt with the development of stochastic source models. These models are tested by demonstrating that histograms and correlation functions constructed from data extracted after utilizing the models match well with the corresponding quantities derived from the raw video stream data. The paper written by Bae and Suda [1] is a good survey of such video models. The authors of [2]–[5] propose different kinds of source video models based on neural networks, Markov chains, and statistical techniques. An insight into different kinds of autoregressive and Markovian models of video source traffic is also provided by [6]–[8]. Several other papers [9]–[11] provide comprehensive knowledge about the various approaches taken by researchers to model video source traffic. Chodorek and Chodorek [12] develop a linear predictor of the MPEG video traffic based on partitioning of the phase space into subregions.

While there is a deluge of research papers that deal with development of statistical and stochastic models, less work has

Manuscript received October 4, 2002; revised April 7, 2003. This work was supported by the State of Texas Advanced Technology Program under Grant 512-0025-2001, the U.S. Department of Energy under Grant DE-FG07-98ID13641, and the National Science Foundation under Grants CMS-0100238 and CMS-0097719. The associate editor coordinating the review of this paper and approving it for publication was Prof. Ilkka Norros.

A. Bhattacharya and A. G. Parlos are with the Department of Mechanical Engineering, Texas A&M University, College Station, TX 77843 USA (e-mail: aninda@tamu.edu; a-parlos@tamu.edu).

A. F. Atiya is with the Department of Computer Engineering, Cairo University, Giza, Egypt (e-mail: amiratiya@link.net).

Digital Object Identifier 10.1109/TSP.2003.814470

been done in the development of predictive models that can capture the inherent nonstationarities and nonlinearities associated with MPEG-coded real-time video streams and that can be used in real time. Chang and Hu [13] investigate the application of a pipelined recurrent neural network (PRNN) for the adaptive traffic prediction of MPEG video signals over ATM networks. Two research papers presented by Doulamis *et al.* [14] and [15] investigate the application of neural networks for nonlinear traffic prediction of VBR MPEG-coded video sources. In a very recent paper by Doulamis *et al.* [16], the authors propose an adaptable neural-network architecture covering online and offline traffic modeling. This paper has results obtained from experiments performed on four sources encoded using MPEG-2.

Adas [17], [18] proposes an adaptive linear predictor for video source traffic prediction. Both a Wiener–Hopf and a normalized least mean square (NLMS) predictor are used and compared for performance. Based on the original work of Adas, Yoo [19] develops an adaptive traffic prediction scheme for VBR MPEG video sources that includes an analysis of the effects of scene changes and traffic variations on the prediction errors. Yoo selects an adaptive time domain prediction technique using the LMS algorithm and considers multiplexed video streams, exhibiting different statistical properties in comparison to single video streams. The paper provides a methodology to predict the sizes of the I- and P-frames. Even though the B-frames are predicted using a linear LMS scheme, prediction errors for the B-frames alone are not reported. Rather, prediction errors for complete video traces are given. Yoo motivates the reported research on the need for a dynamic resource allocation method in multimedia networks.

The works of both Adas and Yoo consider a pth-order, linear single-step predictor (SSP). The coefficients are adapted online. The research papers by Adas [18], Doulamis *et al.* [14]–[16], Chodorek and Chodorek [12], Chang and Hu [13], and Yoo [19] are the most relevant to the current research. The work by Adas [18], Yoo [19], and Chodorek and Chodorek [12] represent some of the most comprehensive information about the achieved SSP errors, while utilizing video traces that can be obtained from a public archive [20]. As a result, SSP comparisons can be presented with respect to any of these publications. Adas also reports SS predictions for time-series comprised of group of pictures (GOPs) or group of video object planes (GOVs) [18]. These schemes are still considered SSPs and not multi-step-ahead predictors (MSP)s because only the next (or one-step-ahead) prediction of a variable, in this case the GOP (or GOV), is estimated. In addition to the lack of MSP results in the literature, no predictors for MPEG-4 video traces have been reported thus far.

In the current paper, a method is presented for designing SSPs and MSPs for estimating video source traffic levels within a finite future horizon. The intended use of these predictors is online and in real time. Past measured or predicted source level information is utilized as inputs in designing the predictors. Such predictors must be equally useful irrespective of the

1) video quality generated, i.e., the quantization level;
2) nature of video content, i.e., low- action versus high-action video;
3) specific encoding scheme used by the encoder.

Toward the end, the present paper makes the following contributions:

- development of SSPs for estimating MPEG-coded video source traffic time-series;
- development of MSPs for recursively estimating MPEG-coded video source traffic time-series within a finite future horizon;
- demonstration of the ability of the developed predictors to perform equally well on video sequences encoded with varying quantization levels and differing MPEG encoding schemes.

The paper is organized into six sections. In Section II, a brief overview of MPEG coding is presented. Section III summarizes the architectural details and the learning algorithms used in developing the neuro-predictors. Section IV presents the implementation results of this study and discusses the relevant outcomes. Comparisons with published results from the literature are also reported in this section. In the final section, a summary and some conclusions from this study are presented.

II. MPEG Video Coding

MPEG standards define the guidelines for the vast array of encoders that generate a compliant bit stream from videos and motion pictures and the methods used by decoders to interpret them. The point to be noted is that MPEG does not specify guidelines on how to create an encoder. This implies that MPEG does not impose any restrictions on technologies that are used in the creation of encoders or decoders (also known as codecs).

In MPEG-1, the term sequence of pictures represents a complete video clip. The next relevant definition is that of GOPs. An MPEG-1 bit stream consists of a repeating GOP structure. The GOP consists of pictures coded in three different ways and arranged in a repetitive structure. The next unit down the hierarchy is *frames*. Frames are the basic building blocks of any MPEG-1 stream. *Macroblocks* are the units that make up each frame. A macroblock contains all the information required for an area of picture representing $16 * 16$ luminance pixels.

There are basically three types of frames in MPEG-1 bitstream.

- Intraframes (I-frames): An *intraframe* or *I-frame* is a frame that is encoded using only information from within that frame. It is a frame that is encoded spatially with no information from any other frame.
- Nonintra frames (P-frames and B-frames): Nonintra frames use information from outside the frame, i.e., from the frames that have already been encoded. In nonintra frames, motion-compensated information is used for a macroblock. This results in fewer data than directly coding the macroblock. There are two types of nonintra frames—*predicted frames (P-frames)* and *bidirectional frames (B-frames)*. The I-frame is typically used as a reference for creating the B and P frames. To encode a P-frame, each macroblock in the P-frame will search for a matching macroblock in the encoded I-frame. Residuals of the macroblocks must be considered if they are not identical. Thus, P-frames are predicted from the I-frames and other P-frames. B-frames are encoded by

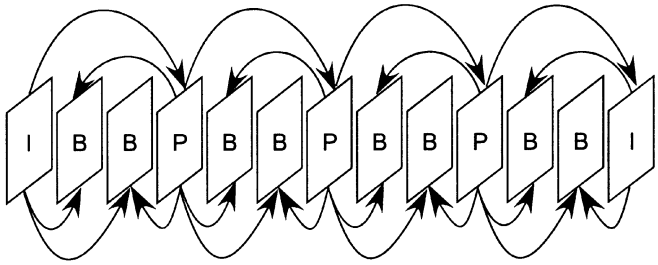

Fig. 1. GOP coding structure.

using the information from I-frames as well as P-frames. This is known as *bidirectional encoding*.

A typical MPEG GOP can be represented by the sequence of frames $IBBPBBPBBPBB$, as well as $IBBPBBPBBPBBPBB$. The entire video clip comprises of this sequence of frames repeating itself. The sequence $IBBPBBPBBPBB$ is called a regular GOP sequence as it can be characterized by two parameters—M and N. M represents the distance between two I-frames, whereas N represents the distance between two P-frames. In the case shown above, $M = 12$, and $N = 3$. Although it is quite easy to construct irregular GOPs, they are not used in real-time multimedia streams. However, they are used in other applications like DVDs. B-frames require both the preceding as well as succeeding anchor files for decoding. This is the reason that P-frames are sent earlier than the B-frames of a GOP. For a playback sequence of $IBBPBBPBBPBB\cdots$, the transmission sequence is $IPBBPBB\cdots$. The arrows of Fig. 1 depict these frame dependencies.

The main objective for the formulation of MPEG-4 is encoding video and audio at very low rates. Another objective is to increase error resilience to packet losses. The MPEG-4 architecture has made it possible for the generation of many new types of applications. Introduction of objects is one of the significant contribution of this standard. Different parts of the final scene can be coded and transmitted separately as *video objects* and *audio objects* to be brought together by the decoder. Separation of objects allows interaction with the objects. This feature is very useful in games and educational software.

Each of the objects in an MPEG-4 image is a visual object plane (VOP). Therefore, each object in a scene is represented by a series of VOPs in time. This is not true for static objects in video images. The static objects can be represented by a single VOP. A VOP contains texture and shape data associated with the respective object. VOPs are analogous to frames in the earlier versions of MPEG standards. VOPs (and frames) can be coded using the discrete cosine transform (DCT) or DCT and motion compensation techniques.

The next level of upper hierarchy in the MPEG-4 standard is GOV. GOVs are similar to GOPs in earlier versions of the MPEG standard. They provide points in the bitstream where the VOPs are coded independently from each other. Video session (VS) is the top video level of the MPEG-4 standard. It comprises of all video objects, irrespective of their nature of origin in the scene. As defined in the earlier versions of the MPEG standard, encoder syntax must support many coding possibilities. The blocks of the images in an MPEG-4 stream can be coded as either I-, P- or B-VOPs.

III. Neural Network Predictors

A. Neural Network Architectures

Two neural network architectures are used in this research: a feedforward multilayer perceptron (FMLP) and a recurrent multilayer perceptron (RMLP). An FMLP is simply the standard feedforward neural network. An RMLP is a multilayer network where each hidden layer possesses crosstalk connections. The crosstalk serves to add memory to the network, making it suitable for modeling dynamic systems. Each of the processing elements of an RMLP is governed by the following equations:

$$z_{[l,i]}(t) = \sum_{j=1}^{N_{[l]}} w_{[l,j][l,i]} x_{[l,j]}(t-1) + \sum_{j=1}^{N_{[l-1]}} w_{[l-1,j][l,i]} x_{[l-1,j]}(t) + b_{[l,i]} \quad (1)$$

and

$$x_{[l,i]}(t) = \sigma_{[l,i]}\left(z_{[l,i]}(t)\right) \quad (2)$$

where $z_{[l,i]}(t)$ represents the internal state variable of the ith node at the lth layer for sample t; $x_{[l,i]}(t)$ is the ith node output of the lth layer for sample t, and $b_{[l,i]}$ is the bias of the node; $w_{[l,j][l',i]}$ is the weight associated with the link between the jth node of the lth layer to the ith node of the l'th layer. Furthermore, t represents the discrete time at which the node and network outputs are computed, with the node index $i = 1, \ldots, N_{[l]}$, and layer index $l = 1, \ldots, \mathcal{L}$, and with the $\sigma_{[l,i]}(\cdot)$ for the input and output layers ($l = 1$ and $l = \mathcal{L}$) being linear. The function $\sigma_{[l,i]}(\cdot)$ for the hidden-layer nodes is a squashing function, and in this study, $\tanh(\cdot)$ is used. The term $b_{[l,i]}$ provides the bias for each node. The processing elements of an FMLP are also governed by equations similar to (1) and (2). However, in (1), the contribution of the first sum is not included.

B. Formulation of the Neuro-Predictors

The two neural networks described above are used to construct two types of predictors.

Single-Step-Ahead Predictors (SSPs): The value of the time-series at the time step $t+1$ is predicted using time-series measurements up time t. In other words

$$\hat{y}(t+1\,|\,t) = f(\mathcal{U}(t)) \quad (3)$$

where $\mathcal{U}(t)$ represents time-lagged values of the time-series, outputs $y(\cdot)$, or transformations of the time-series, inputs $u(\cdot)$, up to time t. That is

$$\mathcal{U}(t) \equiv [y(t), \ldots, y(t-n_y+1), u(t), \ldots, u(t-n_u+1)] \quad (4)$$

where n_y and n_u are the maximum number of lags in the outputs and inputs, respectively.

Multistep-Ahead Predictors (MSPs): The value of the time-series at the time step $t+1$ is predicted recursively, using time-series measurements up time $t-p+1$, where $p > 1$. The *MSP* equation is expressed as

$$\hat{y}(t+1\,|\,t-p+1) = f(\hat{\mathcal{U}}(t)) \quad (5)$$

where

$$\hat{\mathcal{U}}(t) \equiv [\hat{y}(t \mid t-p+1), \ldots, \hat{y}(t-n_y+1 \mid t-p+1) \; u(t), \ldots, u(t-n_u+1)] \quad (6)$$

where the variables are as previously defined. If the inputs $u(t)$ for $t > t-p+1$ are not available, best estimates of these quantities can be used instead.

In both of the aforementioned SSP and MSP formulations, the functional $f(\cdot)$ can be approximated either by an FMLP or an RMLP. This results in four combinations of neuro-predictors.

C. Training Algorithms for Neuro-Predictors

To train an FMLP network used in SSP, the standard back-propagation (BP) algorithm is utilized (see Haykin [21]). For training an RMLP network in the SSP, the algorithm developed by the authors in [22] is used, whereas for training either an FMLP or an RMLP network for MSP, the algorithm developed by the authors in [23] is used. The latter method minimizes an objective function that is presentative of the MSP nature of the problem, rather than an SSP objective function error typically found in many recurrent neural network learning algorithms. This is a dynamic learning algorithm that takes into account the propagation of any weight change throughout the prediction horizon, as well as with the network output update iteration. The error gradients for this algorithm are quite complex, and they are not repeated here due to space limitations. These gradients have been recently published in full (see [23]).

In addition to the specific training algorithms used, selection of the stopping point during learning and network architecture determination are two important aspects of neuro-predictor design. Throughout this study, procedures that are well known in the neural network community are used in stopping the learning and in network architecture selection [21]. Cross-validation is used to determine the stopping point during learning. A segment of the training set is set aside and not used during network parameter updates. Rather, network performance is tested on this cross-validation segment of the training set.

D. Real-Time Implementation of Neuro-Predictors

The predictors proposed in this study are intended to work online and in real time. Predictors based on neural networks, and in fact on many other nonlinear estimation tools, are very time consuming to design, i.e., train and validate. Nevertheless, once training is complete, the predictor execution, which is also called the recall phase, is quite fast, enabling its real-time implementation. The recall phase of a neuro-predictor requires execution of a few multiplication and additions per processing element or node, depending on the size of the preceding network layer, and execution of a $tanh$ function per node, implemented as a look-up table. An additional advantage of such predictors, as opposed to the adaptive linear predictors proposed in [18] and [19], is that online network weight adaptation is not typically needed because of the good generalization achieved during off-line training. The proposed predictors are adaptive, but all parameter adaptation is performed offline, as opposed to LMS-type predictors, which are continuously adapted. This reduces the impact of various convergence problems encountered in the online implementation of adaptive predictors. Nevertheless, in recent studies, it has been demonstrated that online parameter adaptation of neuro-predictors can further improve their predictive accuracy compared to neuro-predictors implemented online with fixed parameters [24], [25].

In a recent study, a comprehensive computational complexity analysis of FMLP and RMLP predictors has been performed, demonstrating that real-time implementation of such predictors is indeed feasible [26]. In addition, recently, such neuro-predictors have been implemented on a fixed-point arithmetic digital signal processor (DSP), and their real-time operation has been experimentally verified at much higher sampling rate requirements than the video source traffic prediction application presented in this study [27]. In fact, real-time neuro-predictor operation has been experimentally verified at a 1000-Hz sampling rate, as opposed to the 25-Hz sampling rate of the current application (25 frames/s).

TABLE I
PEAK/MEAN AND MEAN BIT RATE (MBR) OF MPEG-4 TRACES

Trace	Peak/Mean($X_{max}/\overline{X}$)	MBR(Mbps)
Aladdin	7.1	0.4
ARD Talk	5.7	0.4
Jurassic Park I	4.4	0.8
Star Wars	6.8	0.3
Die Hard III	6.6	0.2
Lecture Room	11.9	0.1
Silence of the Lambs	21.4	0.1
Skiing	9.8	0.2

IV. NEURO-PREDICTOR IMPLEMENTATION RESULTS

A. Generation of Video Sequences Used

The video traces used are obtained from [28]. This Internet site, which is maintained by the Telecommunication Networks Group of the Technical University of Berlin, Berlin, Germany, is an excellent repository of downloadable MPEG-4 and $H.263$ video traces. A technical report available at the site provides the details of the procedure used in generating the video traffic traces [29]. The video was played from VHS tapes using an ordinary video cassette recorder. Uncompressed YUV information of each video was grabbed using the tool `bttvgrab` (version 0.15.10) [30] at a frame rate of 25 frames/s in the QCIF format. The luminance resolution was 176×144 picture elements (pels) and $4:1:1$ chrominance subsampling at a color depth of 8 bits. This information was stored on a disk. The stored YUV frame sequences were used as input for both the MPEG-4 encoder and the $H.263$ encoder. The encoding was not performed in real-time. Mean bit rate (MBR) provides information about the picture quality of the traces used. Table I provides MBRs of the video data traces used in the current research.

Each of the considered video sequences has been coded using three different quantization levels. Depending on the level of quantization, encoded sequences have been categorized into high, medium, and low quality. The following sequences

were chosen for the current research work: Aladdin, ARD Talk, Jurassic Park I, Star Wars, Die Hard III, Lecture Room, Silence of the Lambs, and Skiing.

B. Video Sequence Used for Training

For all predictors designed in this study, only a segment of the Aladdin video sequence is used for training and cross-validation, i.e., out-of-sample testing, to determine the stopping point for the training process. After fixing all of the network parameters, the developed predictors are tested on the remaining segment of the Aladdin sequence and on all other complete video sequences mentioned above. All time-series are scaled by a single scaling factor so that they lie mostly in the range from -0.5 to 0.5, making them suitable for processing by the neural network. Some of the details of the video data traces used in the current research, along with the segments used in training and cross-validation, are as follows: The total number of VOPs in each data trace is 89 998, the number of I-VOPs in each data trace is 7500, the number of P-VOPs in each data trace is 22 500, and the number of B-VOPs in each data trace is 59 998. The fraction of I-VOPs used for training (1500) and cross-validation (500) is $2000/7500 = 0.27$, and the fraction of P-VOPs used for training (1500) and cross-validation (500) is $2000/22\,500 = 0.09$, whereas the fraction of B-VOPs used for training (20 000) and cross-validation (10 000) is $30\,000/59\,998 = 0.5$.

Looking at the final prediction results, it appears that the training sets are sufficient to capture the structure underlying the video data traces used in this research and some others reported in the literature employing a different coding scheme. It is difficult to ascertain whether or not the self-similar behavior of the data, if any, has been captured. In some sense, this is somewhat less important to the problem at hand. The important metric is the ability to predict video data traces not used in any manner during the design of the predictors. Increasing the number of data points used in training did not provide any significant improvement. Reducing the number of data points used in training deteriorated the final prediction results.

As seen from Table I, the MBR of the video data traces considered varies from 0.1 Mb/s (Silence of the lambs) to 0.8 Mb/s (Jurassic Park I). The ratio of peak frame size to mean frame size varies from 4.4 (Jurassic Park I) to 21.4 (Silence of the lambs). Additional statistical parameters of the VOPs of the video traces, such as the mean, auto covariance, and autocorrelation, indicate similar, typical range variations. No single video appears to consistently reside at the extremes of these ranges. None of these statistical parameters indicate that the trace of the Aladdin video is a representative sample of all video traces. However, the values of the MBR and the peak/mean ratio of the Aladdin trace are close to the median of the corresponding parameters for all the video traces considered. Apart for this observation, there is no evidence to suggest that the Aladdin trace had any spatial or temporal characteristics that are representative of all the video traces considered. One should point out that the use of aggregate statistical parameters to characterize a highly nonstationary signal is ill-advised, although these are some of the metrics in widespread use by the multimedia networking and communication communities.

The thrust of this research is to show that a generic predictor can be obtained for use in the prediction of video source traffic irrespective of the encoding parameters of the video data traces. Attempting to identify very complex parameters that would indicate whether a video trace is representative enough to be used in training might eliminate the intent of this study. As witnessed by the results of the sections to follow, the developed predictors are generic to some extent, and they can be used to predict the video source traffic for a wide range of MPEG-coded video streams without any tuning.

C. Performance Metrics

Three types of errors were used as performance metric for the prediction schemes developed in the current work. Let $S(j)$ and $\hat{S}(j)$ for $j = 1, \ldots, N$ denote the actual time-series and the predicted time-series. The first error measure is the ratio between the sum of the square of the prediction error and the sum of the square of the actual time-series values. This performance metric is called relative mean square error (RMSE) and is represented by the following equation:

$$\text{RMSE} = \frac{\sum_{j=1}^{N} (S(j) - \hat{S}(j))^2}{\sum_{j=1}^{N} S(j)^2} \times 100. \tag{7}$$

The second performance metric measures the maximum error and is termed the maximum absolute error (MAE), as follows:

$$\text{ME} = \max_{1 \le j \le N} |S(j) - \hat{S}(j)|. \tag{8}$$

The third and the final metric is maximum relative error (MRE). MRE represents a measure of how large the maximum error is relative to the actual time series value, and it is written as

$$\text{MRE} = \max_{1 \le j \le N_S} \frac{|S(j) - \hat{S}(j)|}{|S(j)|}. \tag{9}$$

D. Single-Step-Ahead Prediction of I-VOPs

1) Training and Predictor Structure: In the first experiment, prediction of simply the next frame size for the I-VOPs is considered.[1]

The time-series comprising of the VOP sizes is split into four different time-series. The first time-series consists of I-VOP sizes. The next three time-series consist of the three P-VOP sizes. The time-series are extracted on the basis of a common index. Four different indices are defined as follows:

$$\begin{aligned} k(i) &= M \times (i-1) + 1, \qquad m_1(i) = k(i) + N \\ m_2(i) &= m_1(i) + N, \qquad m_3(i) = m_2(i) + N \end{aligned} \tag{10}$$

where $k(i)$, $m_1(i)$, $m_2(i)$, and $m_3(i)$ are the four new indexes, and $i = 1, 2, \ldots, L$, where L is the number of GOVs in the

[1]In [19], this prediction is considered multistep because of the presence of multiple frames in between two I-VOPs. In the current research, such a prediction would be considered single-step, whereas its extension to multiple-step would include prediction of several I-VOPs.

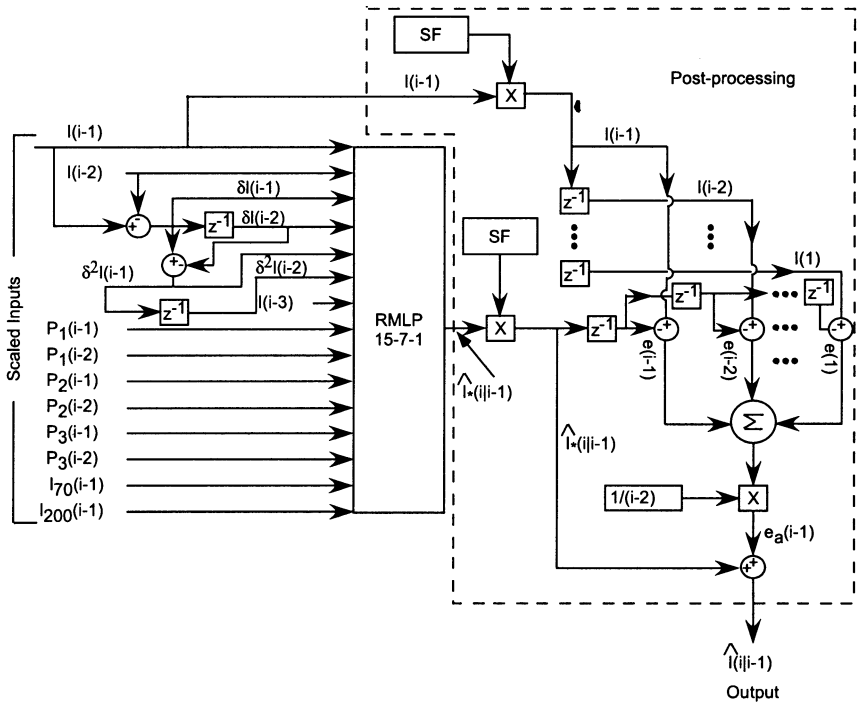

Fig. 2. Schematic representation of the neural network structure for SSP of I-VOPs.

video data trace. M is the distance between two I-VOPs. Similarly, N is the distance between two P-VOPs. For the data traces used in the current research, the values of M and N are 12 and 3, respectively. The indexes defined above can be associated with real-time as follows:

$$t = t_0 + (\text{Index} - 1) \times \Delta \tag{11}$$

where t_0 is the time the video stream starts, and Index represents any of the $k(i)$, $m_1(i)$, $m_2(i)$, or $m_3(i)$. For convenience, t_0 can be assumed to be 0. The time difference between two consecutive VOPs is represented by Δ such that $\Delta = 1/f$, where f is the number of VOPs per second. In the case of video data traces used in the current research, f is equal to 25. For the remainder of this paper, only the indexes will be defined with their association to real-time being similar to (11).

Each I-VOP is linked to the index $k(i)$. Similarly, the P-VOP that appears first in a GOV is linked to $m_1(i)$. The other two P-VOPs in a GOV are linked to the remaining two indices $m_2(i)$ and $m_3(i)$, respectively. Therefore, four different time-series can now be defined from the original time-series of $x(j)$, where j is the VOP index of a video trace, as follows:

$$\begin{aligned} I(i) &= x(k(i)), \qquad P_1(i) = x(m_1(i)) \\ P_2(i) &= x(m_2(i)), \qquad P_3(i) = x(m_3(i)) \end{aligned} \tag{12}$$

where all the indexes are as previously defined.

An RMLP network with 15 input nodes, seven hidden nodes, and one output node (a $15-7-1$ network) is used. The predictor architecture is determined by following the procedure outlined in Section III-C. The SSP is trained using the first 1500 I-VOP sizes of the data trace Aladdin, while cross-validated on the following 500 I-VOP sizes of the same data trace. The designed predictor is tested on the remaining, unused segment of the entire Aladdin video stream, as well as on the entire video traces of all the other video streams considered. Off-line training of this RMLP network from randomly initialized weights is continued for 175 000 training cycles or epochs, consuming approximately 100 hours of real-time on a personal computer that is about three years old. As indicated in later sections on MSP, incremental off-line training of a network is significantly faster, reducing total time needed for learning to a couple of hours.

The inputs to the neuro-predictor for prediction of the ith I-VOP size are sizes of the previous three I-VOPs, sizes of the previous P-VOPs, difference between the sizes of two consecutive I-VOPs, second derivative of the I-VOP sizes, and moving averages of I-VOP sizes. The difference between the sizes of two consecutive I-VOPs is calculated as shown in the following:

$$\delta I(i) = I(i) - I(i-1). \tag{13}$$

The second derivative of the I-VOP sizes can be calculated by

$$\delta^2 I(i) = \delta I(i) - \delta I(i-1) = I(i) - 2I(i-1) + I(i-2) \tag{14}$$

where the second derivative takes values starting at $i = 3$. The simple moving average is defined as

$$I_w(i) = \frac{1}{w} \sum_{j=i-w+1}^{i} I(j). \tag{15}$$

Two different moving averages are included as inputs $I_{70}(i)$ and $I_{200}(i)$.

2) Post-Processing and Testing of Predictor Outputs: Once the prediction is performed, the predicted output is rescaled back in its original range. Although the predicted data managed to capture the trends of the time series to a great extent, yet it had an almost constant offset from the original time-series. Therefore, a postprocessing step was included, whereby an estimate of this offset is added to the predicted output; see Fig. 2 for a display of this corrective term. The added offset is estimated by

TABLE II
PERFORMANCE METRICS OF THE SSP FOR THE I-VOP SIZES

Trace	RMSE (%)	MAE (in bytes)	MRE
Aladdin	2.6	9379.3	13.2
ARD Talk	0.9	6746.1	2.5
Jurassic Park I	0.8	9079.8	7.7
Star Wars	1.5	4784.3	6.2
Die Hard III	2.9	4636.9	10.5
Lecture Room	0.2	2687.1	0.9
Silence of the Lambs	3.6	10802.0	10.4
Skiing	2.0	4831.7	2.4

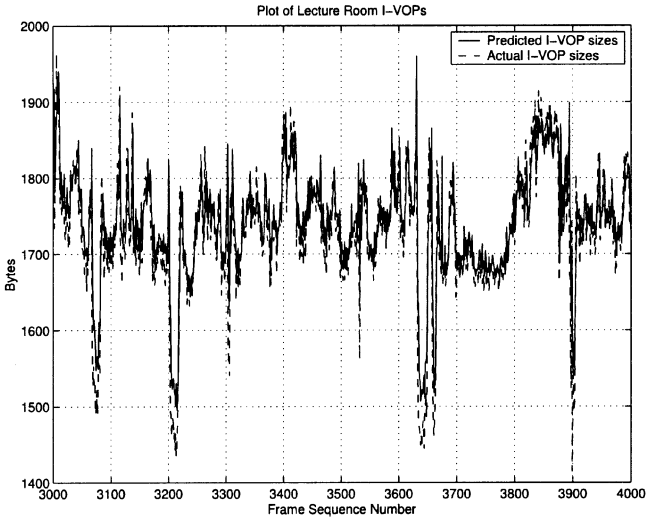

Fig. 3. Single-step-ahead predicted sizes of I-VOPs of Lecture Room.

observing the history of differences between the actual and predicted values. The new prediction becomes

$$\hat{I}(i\,|\,i-1) = \hat{I}_*(i\,|\,i-1) + \frac{1}{(i-2)}\sum_{j=2}^{i-1}(I(j) - \hat{I}_*(j\,|\,j-1)) \tag{16}$$

where $\hat{I}(i\,|\,i-1)$ is the predicted size of I-VOPs at step i after the offset error correction, $\hat{I}_*(i\,|\,i-1)$ is the actual output from the neural network, and $I(j)$ is the actual size of I-VOPs at step j.

Fig. 2 depicts the architectural details and inputs used in this SSP. The term SF in the figure is the scaling factor used to rescale the output, whereas $e_a(i)$ represents the error between the predicted I-VOP size, $\hat{I}_*(i\,|\,i-1)$, and the actual I-VOP size $I(i)$ for $i = 1, 2, \ldots, L$, where L is the number of GOVs in the video data trace.

Following the inclusion of this correction term, the I-VOP SSP is tested using all of the complete video traces considered. Table II summarizes the performance metrics of the proposed I-VOP predictor for all the complete video traces tested. Fig. 3 shows a portion of the time-series depicting the best prediction range.

E. Single-Step-Ahead Prediction of P-VOPs

In the next experiment, the design of a neural network for predicting the sizes of P-VOPs in single-step-ahead is considered. VOPs from the original stream are preprocessed and separated into several substreams (time-series). Elements of these time-series are components of an input vector to the predictor. Unlike the inputs used in the I-VOP case, the P-VOPs in this predictor are not divided into three time-series. Instead, a composite time-series consisting of all P-VOPs is considered.

The index $m(j)$ of this composite time-series is defined as follows:

$$m(j) = \begin{cases} 3\times(j+1)+1, & \text{if } j \neq 1 \text{ and } \bmod(j-1,\,3)=0 \\ 3\times j+1, & \text{otherwise} \end{cases} \tag{17}$$

where $j = 1, 2, \ldots, \mathcal{P}$, where $\mathcal{P}$ is $L \times N$, and where N is the distance between two P-VOPs. In the context of the current research, the value of $N = 3$. Based on the above index, the following time-series is defined in terms of the original time-series $x(j)$ as

$$P(j) = x(m(j)) \tag{18}$$

where all indexes are as previously defined. As in the case of SSP of the I-VOPs, the $I(i)$ time series is also used in this experiment.

While experimenting with various inputs for SSP of I-VOP sizes, it is observed that involving the first difference between the sizes of VOPs in the prediction scheme provided the neural network with a sense to judge the direction of the impending change of the size of VOP in the next time step. This led to considerable improvement in prediction accuracy, but in the case of P-VOPs, it is not sufficient to include the difference between consecutive P-VOP sizes. Including differences between consecutive P-VOPs and I-VOPs led to even more improved performance. A composite time-series comprising of only I- and P-VOP sizes is created for each video data trace. In constructing this time-series, B-VOP sizes are neglected. The following index $l(p)$ is defined as

$$l(p) = 3\times(p-1)+1 \tag{19}$$

where $p = 1, 2, \ldots, \mathcal{M}$, and where $\mathcal{M}$ is $L \times (N+1)$. Based on the above index, the following time-series is defined in terms of the original time-series $x(j)$ as

$$\delta P(p) = \delta x(l(p)) = x(l(p)) - x(l(p-1)) \tag{20}$$

where all indexes are as previously defined. The parameters $\delta P(p-1)$, $\delta P(p-2)$, $\delta P(p-3)$, and $\delta P(p-4)$ represent the difference between the sizes of consecutive I- and P-VOPs of the new series extracted from the original VOP size series. $\delta P(p)$ can be represented visually by the following series:

$$\underbrace{I\cdots P}_{\delta P(p-4)}\cdots\underbrace{P\cdots I}_{\delta P(p-1)}\ \underbrace{P}_{P(j\,|\,j-1)}\ .$$

An FMLP of size $11-10-1$ is used for this predictor, although an RMLP of similar size produced almost similar results. The similarity in the performance of the two architectures prompted the use of the simpler predictor, which is the

TABLE III
PERFORMANCE METRICS OF THE SSP FOR THE P-VOP SIZES

Trace	RMSE (%)	MAE (in bytes)	MRE
Aladdin	10.3	10104.0	12.6
ARD Talk	5.9	11696.0	6.4
Jurassic Park I	4.0	13418.0	5.9
Star Wars	9.2	9313.3	12.9
Die Hard III	9.0	5742.3	6.7
Lecture Room	6.9	2713.4	5.7
Silence of the Lambs	11.0	8044.7	18.6
Skiing	6.2	4671.2	21.3

FMLP. Obtaining similar predictive results from both FMLP and RMLP networks appears to be coincidental, as it is not widely encountered in practice. Therefore, throughout this research, both classes of network architectures are tested when designing a predictor. The specific predictor architecture is determined by following the procedure outlined in Section III-C. Similar to the I-VOP case, the training set consists of the first 1500 P-VOP sizes of the data trace Aladdin, whereas the following 500 P-VOP sizes of the same data trace are used for cross-validation. The designed predictor is tested on the remaining, unused segment of the entire Aladdin video stream, as well as on the entire video traces of all the other video streams considered. Off-line training of this FMLP network from randomly initialized weights is continued for 912 000 training cycles or epochs, consuming approximately 225 h of real-time on a personal computer that is about three years old.

Sizes of previous three P-VOPs $P(j-1)$, $P(j-2)$, and $P(j-3)$, sizes of the previous two I-VOPs $I(i)$ and $I(i-1)$, and I-VOP size differences $\delta I(i)$ and $\delta I(i-1)$ are the inputs used in the P-VOP SSP. Difference in sizes between the VOPs of the composite series defined by the index $l(p)$, $\delta P(p-1)$, $\delta P(p-2)$, $\delta P(p-3)$, and $\delta P(p-4)$ are also used as inputs to the predictor.

The post-processing component of the designed predictor for predicting the sizes of P-VOPs is similar to that of the post-processing component of I-VOPs. A summary of the results for SSP prediction of P-VOPs is shown in Table III. Fig. 4 depicts portions of the best performing segment of the time-series, respectively.

F. Single-Step-Ahead Prediction of B-VOPs

In the third experiment an SSP is implemented for predicting the B-VOP sizes. An FMLP of size $11-15-1$ is used for this purpose. The predictor architecture is determined by following the procedure outlined in Section III-C. The predictor is trained and cross-validated using 20 000 and 10 000 B-VOP sizes of Aladdin, respectively. The designed predictor is tested on the remaining, unused segment of the entire Aladdin video stream, as well as on the entire video traces of all the other video streams considered. Offline training of this FMLP network from randomly initialized weights is continued for 65 000 training cycles or epochs, consuming approximately 16 h of real-time on a personal computer that is about three years old.

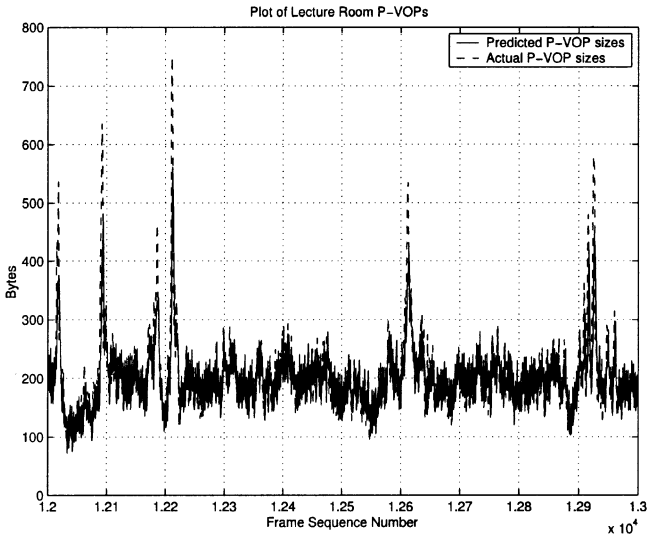

Fig. 4. Single-step-ahead predicted sizes of P-VOPs of Lecture Room.

The majority of inputs used are more or less similar to those used in the previous I-VOP and P-VOP predictors. However, the following index $q(r)$ is defined that can be associated with the time-series consisting of the B-VOPs:

$$q(r) = \begin{cases} \dfrac{3r+1}{2}, & \text{if } r \text{ odd} \\ \dfrac{3r}{2}, & \text{if } r \text{ even} \end{cases} \tag{21}$$

where $r = 1, 2, \ldots, \mathcal{Q}$, where $\mathcal{Q}$ is $L \times Z$, with Z the number of B-VOPs between two I-VOPs. In this study, Z is 8. Using the index $q(r)$, the following time-series is defined:

$$B(r) = x(q(r)) \tag{22}$$

where all indexes are as previously defined. As before, $B(r-1)$ represents past B-VOP sizes.

Differences between the sizes of B-VOPs, not only from each other but also from the I- and P-VOPs, provide information to the neural network about the gradient of B-VOP sizes. The original time-series $x(j)$ can be used to defined the following time series denoting the gradient of the B-VOPs as

$$\delta B(j) = \delta x(j) = x(j) - x(j-1) \tag{23}$$

where the index j takes values starting from 2 up to the number of total VOPs.

A composite time-series consisting of I- and P-VOPs is developed as an input to the neural network. This series can be represented as $IPPPI\cdots$. Encoding of B-VOPs is heavily dependent on the I- and P-VOPs. Thus, it makes logical sense to include a series combining I- and P-VOP sizes to predict the size of B-VOPs. From now on, VOPs belonging to this special series will be referred as IP-VOPs. The index $l(p)$, which was defined earlier in the subsection dealing with SSP of sizes of P-VOPs can be used to define this time-series as follows:

$$\text{IP}(p) = x(l(p)). \tag{24}$$

The difference in the sizes between IP-VOPs gives an indication of the gradient series. The consecutive differences

TABLE IV
PERFORMANCE METRICS OF THE SSP FOR THE B-VOP SIZES

Trace	RMSE (%)	MAE (in bytes)	MRE
Aladdin	8.2	8333.8	15.0
ARD Talk	3.2	4688.2	33.0
Jurassic Park I	2.2	8440.1	67.5
Star Wars	3.5	4138.4	25.9
Die Hard III	4.0	3692.1	10.7
Lecture Room	28.3	821.3	4.7
Silence of the Lambs	27.8	2741.8	18.8
Skiing	14.7	1686.1	29.1

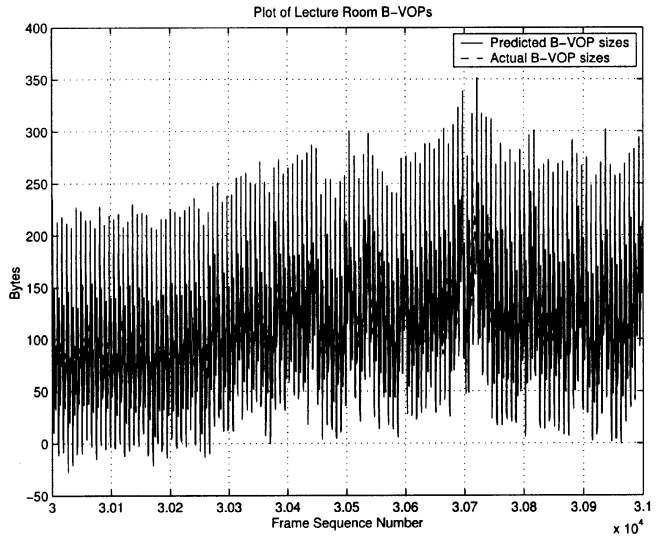

Fig. 5. Single-step-ahead predicted sizes of B-VOPs of Lecture Room.

$\delta IP(l(p))$ are also used as inputs that are computed from the $\delta P(p)$ series.

The inputs used in the B-VOP SSP are sizes of the previous three B-VOPs ($B(r-1)$, $B(r-2)$, and $B(r-3)$), B-VOP size differences ($\delta B(j-1)$, $\delta B(j-2)$, and $\delta B(j-3)$), sizes of the I-VOPs and P-VOP of the composite IP-VOP series ($IP(p)$, $IP(p-1)$, $IP(p-2)$), and difference in sizes between the I-VOPs and P-VOP of the composite IP-VOP series ($\delta P(p)$, $\delta P(p-1)$).

The post-processing of B-VOPs follows along the same lines as the post-processing schemes of I- and P-VOPs. Table IV tabulates the performance of the B-VOP prediction scheme for the tested video traces. Like I- and P-VOP prediction results, the best performing time-series is shown in Fig. 5.

G. Single-Step-Ahead Prediction of the Moving Average Time-Series of VOP Sizes

In this experiment, instead of the original time-series, a time-series that is a smoothed and down-sampled version of the original VOP size series is considered. There are a number of reasons for considering this experiment more useful from a practical point-of-view. The VOP size time-series is extremely noisy. Smoothing it removes the unwanted noise and focuses the analysis on the fundamental dynamics of the traffic and its long-term dependencies. The other reason is that whether our goal is dynamic bandwidth allocation or control of the media sending rate, for both cases, a long-term horizon is needed. By predicting the moving average over some horizon, essentially, an estimate of the average frame sizes over the specified averaging horizon will be available. Having estimates of this averaged time-series over a longer horizon will enable better control and planning. In fact, for a very short horizon, the control effort will require too much computational overhead. As it will be demonstrated in the sequel, the fortunate fact is that the predictor performance does not degrade much with the length of the averaging horizon, which is not surprising in view of the well-known multiscale properties of network traffic time-series.

The moving average time-series $X(k)$ (in units of Bytes) is given in terms of the original time-series $x(j)$ by the following equation:

$$X(k) = \frac{1}{w} \sum_{j=kp-w+1}^{kp} x(j) \tag{25}$$

where w is the window size, p is the amount by which the window is moved ($p \leq w$), and $x(j)$ is the VOP size of the jth VOP in the video trace. The values of w and p are selected to be 25 and 12, respectively. The value of $w = 25$ corresponds to a video stream segment of 1 s.

An FMLP of size $11 - 22 - 1$ is used for the prediction of the moving average time-series of VOP sizes. The predictor architecture is determined by following the procedure outlined in Section III-C. The predictor is trained and cross-validated using 1500 and 500 data points from the moving average time-series of VOP sizes of Aladdin, respectively. The designed predictor is tested on the remaining, unused segment of the entire Aladdin video stream, as well as on the entire video traces of all the other video streams considered. Offline training of this FMLP network from randomly initialized weights is continued for 2000 training cycles or epochs, consuming approximately 1 h of real-time on a personal computer that is about three years old. The significantly reduced training time for both the SSP and MSP of the moving average series can be attributed to the smoother nature of the signals involved, as compared with the frame-by-frame video signals. Fig. 6 depicts the neural network structure used.

The inputs used in this predictor are the previous three moving averages ($X(k-1)$, $X(k-2)$, and $X(k-3)$), three lags of difference between the sizes of two consecutive data points of the moving average time-series $\delta X(k) = X(k) - X(k-1)$, three lags of second derivative of the moving average time-series $\delta^2 X(k) = X(k) - 2X(k-1) + X(k-2)$, and two lags of the I-VOP size of the original time-series $I(k-1)$ and $I(k-2)$.

The designed predictor employs an equivalent post-processing as described earlier in the section on I-VOP size prediction. Fig. 7 shows the best prediction result. The values of the three performance metrics used in this study are listed in Table V.

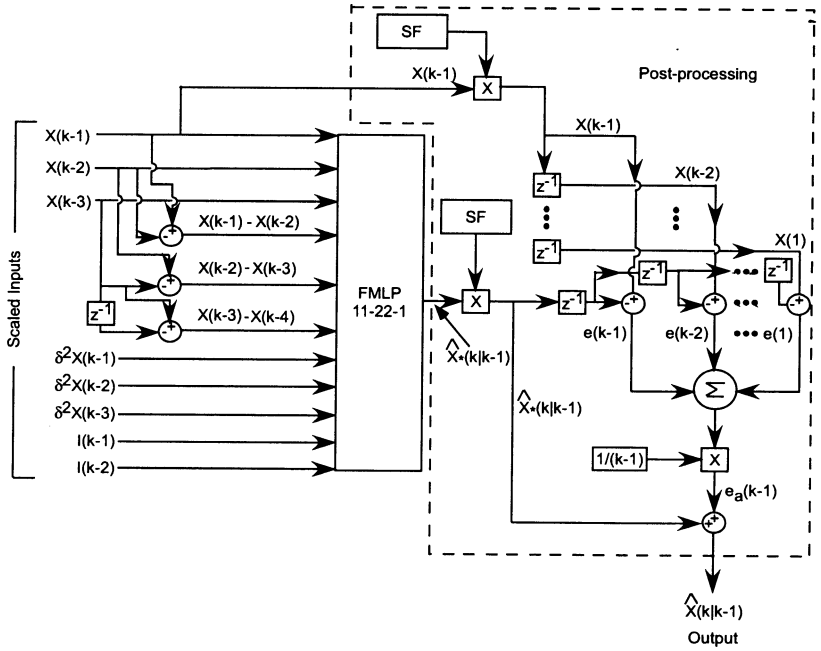

Fig. 6. Schematic representation of the neural network structure for SSP of moving average time series of VOP sizes.

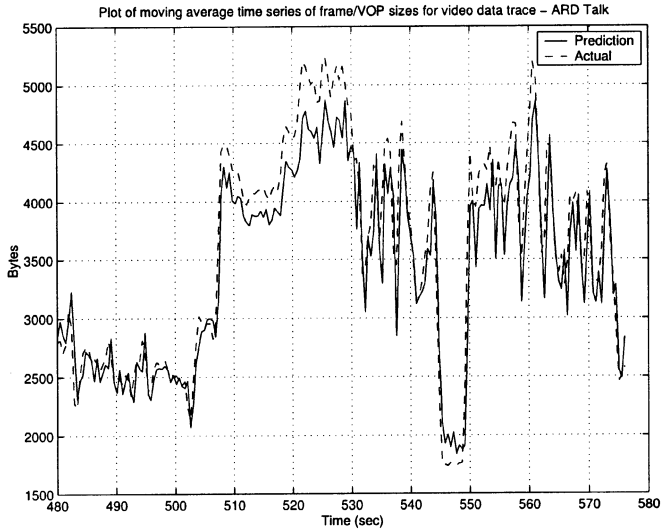

Fig. 7. Single-step-ahead prediction of moving average time series of VOP sizes for ARD Talk.

TABLE V
PERFORMANCE METRICS OF THE SSP OF SMOOTHED TIME-SERIES

Trace	RMSE (%)	MAE (in bytes)	MRE
Aladdin	2.5	7843.9	18.9
ARD Talk	0.8	6333.5	3.3
Jurassic Park I	1.0	7658.0	12.7
Star Wars	1.7	7799.8	17.0
Die Hard III	2.4	7841.1	18.7
Lecture Room	14.8	7517.2	10.1
Silence of the Lambs	3.4	6084.0	2.8
Skiing	2.7	7888.8	21.3

H. Multi-Step-Ahead Neuro-Predictors

As mentioned in the previous subsection, time-series prediction over longer horizons is highly desirable. To assess how far in the future prediction is feasible, experiments are performed for MSP. The moving average time-series is considered, and the following experiments are performed: In the first experiment, two-steps-ahead are predicted, whereas in the second experiment, four-steps-ahead are predicted. These two predictions correspond to 1 and 2 s horizons, respectively. The algorithm developed by the authors in [23] is used to train neural networks for MSP; see Section III-C.

An RMLP network having 11 inputs, 22 hidden nodes, and one output node is used to predict both two-step-ahead and four-step-ahead experiments. All inputs are the same as described in the SSP approach of the previous section. The only difference is that in the second prediction step and beyond, the network uses network forecasts rather than actual values of the time-series for some of the inputs. This is because the actual time-series values are not yet available. The predictor architecture is determined by following the procedure outlined in Section III-C. The predictor is trained and cross-validated using 1500 and 500 data points from the moving average time series of VOP sizes of Aladdin, respectively. The designed predictor is tested on the remaining, unused segment of the entire Aladdin video stream, as well as on the entire video traces of all the other video streams considered. Offline training of this RMLP network for the two-step-ahead predictor from randomly initialized weights is continued for 2500 training cycles or epochs, consuming approximately 2 h of real-time on a personal computer that is about three years old. The same neuro-predictor is used for four-step-ahead prediction. Incremental tuning of the two-step-ahead predictor with the time-series comprising of the moving average of VOP sizes for four-step-ahead prediction is accomplished in 2000 training cycles, consuming approximately 1.5 h of real-time on a personal computer that is about three years old. A schematic of this predictor is shown in Fig. 8.

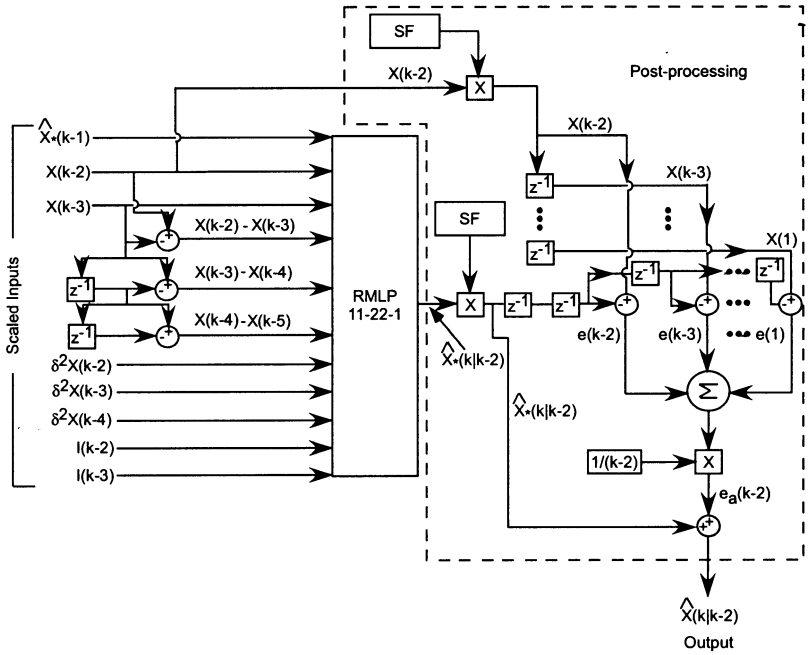

Fig. 8. Schematic representation of the neural network structure for two-step-ahead prediction of moving average time series of VOP sizes.

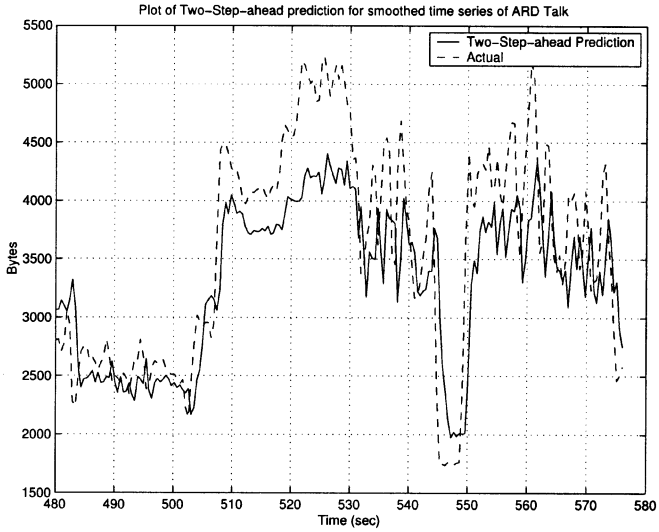

Fig. 9. Two-step-ahead prediction errors for smoothed time series of ARD Talk.

TABLE VI
PERFORMANCE METRICS OF THE TWO-STEP-AHEAD PREDICTION

Trace	RMSE (%)	MAE (in bytes)	MRE
Aladdin	8.2	6891.5	18.3
ARD Talk	2.4	5229.3	2.7
Jurassic Park I	3.8	8081.8	10.7
Star Wars	4.9	6700.4	15.5
Die Hard III	7.8	6685.3	15.9
Lecture Room	22.6	6463.1	9.0
Silence of the Lambs	9.3	4995.8	4.6
Skiing	7.9	6806.6	18.0

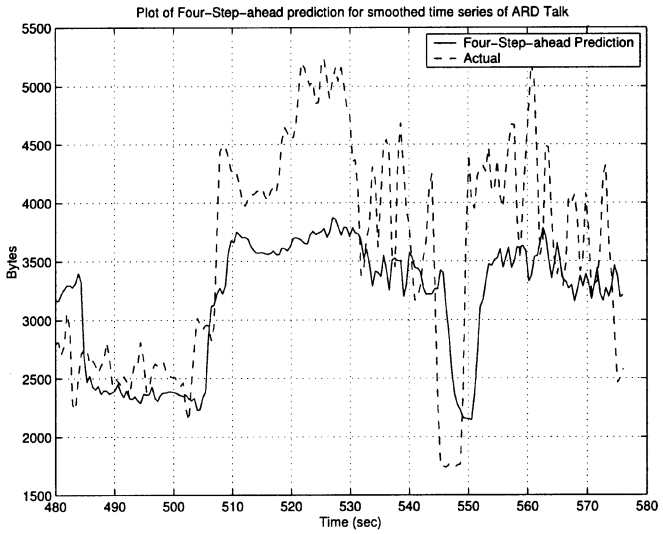

Fig. 10. Four-step-ahead prediction for smoothed time series of ARD Talk.

Fig. 9 shows the result of best-case scenario for two-step-ahead prediction in comparison to the actual values of the moving average time-series. The performance metrics shown in Table VI indicate deterioration in the performance of the two-step-ahead predictor compared with that of the one-step-ahead predictor. Even though a deterioration is expected, it is not clear how significant the deterioration is compared with what is expected. Furthermore, it is not entirely obvious how significant this deterioration is for the applications intended to use the MSP.

The best case of four-step-ahead prediction is shown by Fig. 10. Performance metrics for all the video traces are also given in Table VII. RMSE of each smoothed time-series increases by about two times, indicating further deterioration.

It should be noted that by increasing the averaging or smoothing window and performing an SSP on the resulting time-series would have resulted in much better predictions than the MSP case, even if the horizons for both cases are taken to

TABLE VII
PERFORMANCE METRICS OF THE FOUR-STEP-AHEAD PREDICTION

Trace	RMSE (%)	MAE (in bytes)	MRE
Aladdin	13.0	6186.9	15.1
ARD Talk	4.2	5541.1	2.3
Jurassic Park I	8.0	8728.3	14.1
Star Wars	8.6	6032.3	33.5
Die Hard III	14.9	5980.7	14.2
Lecture Room	33.3	5758.5	8.6
Silence of the Lambs	19.0	4291.2	8.1
Skiing	15.1	6102.2	16.1

be the same. However, the intermediate predictions resulting from a recursive MSP are needed in the case of a flow control algorithm. This is particular true when using flow control algorithms based on model predictive control (MPC).

I. Comparison of Predictor Performance With the Published Literature

Following a literature survey, the work by Adas [18], Yoo [19], and Chodorek and Chodorek [12] appear to have some of the most comprehensive results for SSP of I-, P- and B-VOPs for MPEG-1, and MPEG-2 streams, in the case of the third paper. The first two papers propose the use of adaptive linear predictors with some variations on how to compute the predictor parameters. The former paper uses mostly the Wiener–Hopf formulation and the NLMS algorithms, whereas the latter utilizes some form of an adaptive step-size in the LMS predictor based on the detection of scene change. The last paper proposed a predictor based on phase space analysis. The data traces used by Adas and Yoo in their research are coded using the MPEG-1 standard, whereas Chodorek and Chodorek utilized video traces coded with both MPEG-1 and MPEG-2. The MPEG-1 data traces were obtained from [20]. To judge the comparative performance of the predictors presented in this research, the designed SSPs are applied to the same video traces used by Yoo and comparisons presented. Furthermore, a relatively small sample of complete video traces that are common to the work by Adas [18], Yoo [19], and Chodorek and Chodorek [12] are compiled and compared with the current work. No published literature utilizes MPEG-4 video traces for prediction or performs true MSP. Therefore, the comparison is limited to the SSP case with MPEG-1 coding.

Table VIII compares the RMSE of SSP developed in the current research using MPEG-4 traces, with the predictors from Yoo's research paper. The table deals with the I-frames, P-frames, and with all combined frames for five different MPEG-1 coded data traces lasting almost 30 min. Since Yoo reports only the RMSE for prediction of different data traces, the comparison of the prediction schemes has been limited to RMSE as the only performance metric. Additionally, Table IX presents a comparison of a few complete video traces that are common to the aforementioned three papers. Predictions for the I-, P-, and B-frames, as well as for the total traces, are given.

Although the predictors designed in this research are trained using MPEG-4 data traces, they performed comparatively well

TABLE VIII
COMPARATIVE RMSE RESULTS OF SINGLE-STEP-AHEAD PREDICTION

Trace		This Work	Yoo [19]
Silence of the Lambs	I	2.3	3.1
	P	24.5	22.4
	Total	6.8	6.9
Star Wars	I	1.4	2.4
	P	23.9	16.7
	Total	5.1	7.1
Terminator	I	2.2	3.1
	P	13.3	15.0
	Total	5.2	8.4
Mr. Bean	I	0.9	1.2
	P	10.9	9.8
	Total	3.0	3.2
Simpsons	I	1.8	2.6
	P	22.5	26.0
	Total	7.3	10.1

TABLE IX
COMPARATIVE RMSE RESULTS OF SINGLE-STEP-AHEAD PREDICTION FROM VARIOUS PAPERS

Trace		This Work	Yoo [19]	Adas [18]	Chodorek and Chodorek [12]
Jurassic Park I	I	1.1	-	1.2	-
	P	10.9	-	12.0	-
	B	4.0	-	4.8	-
	Total	3.0	-	-	-
Star Wars	I	1.4	2.4	1.4	1.3
	P	23.9	16.8	26.0	15.3
	B	7.1	-	7.7	7.1
	Total	5.1	7.1	-	4.6
Terminator	I	2.2	6.7	2.5	2.1
	P	13.3	15.7	13.6	8.4
	B	9.2	-	10.6	9.4
	Total	5.2	8.4	-	5.1

while predicting the MPEG-1 data traces. In fact, for the prediction of I-VOPs, our method outperformed Yoo's method for every single trace by a significant amount. The best improvement is approximately 50%. For the P-VOP case, the results are comparable with Yoo's method somewhat outperforming our method. The same can be said regarding the comparison of the predictors presented in this study with those by Adas, as seen in Table IX. The results by Chodorek and Chodorek are comparable with the ones presented in this study, with the exception of the P-VOP predictions, where better results are reported.

J. Comments on the Results

The presented results indicate the success of the proposed approach in predicting VOP sizes (frame-by-frame) and averaged VOP size for SSP and MSP. By looking at the time-series involved, one can certainly observe the noise level in these time-

series. Despite this noise level, RMSE errors of a few percentage points are obtained for I-VOPs. For P-VOPs and B-VOPs, the performance is somewhat worse but, in most cases, is close to or below 10%. Comparing the autocorrelation functions for MPEG-4 video traces with similar figures reported in the literature by Adas and Yoo for MPEG-1 video traces indicates that the former traces have much more pronounced long-term dependencies (LTDs). In fact for 20 lags, the MPEG-4 trace autocorrelation is above 0.9 and in many instances above 0.95, whereas the corresponding values for MPEG-1 traces are significantly lower, in the range of 0.4 to 0.6. time-series with significant LTDs are typically more difficult to predict.

The reported results compare favorably with Yoo's method and especially for I-VOPs, even though the developed predictors were trained on MPEG-4 traces, and then tested on Yoo's MPEG-1 traces. The same can be said regarding the comparison of the predictors presented in this study with those by Adas.

Another interesting result obtained is that the proposed predictor performs considerably well on the smoothed time-series. It obtains errors that are approximately similar to the case of SSP of the I-VOPs but much better than those of the P-VOPs and B-VOPs. This is despite the fact that the smoothed time-series has smoothing horizon of 1 s and includes all I-, P-, and B-VOPs frame sizes in that horizon. The smoothed series is, in fact, more desirable because it allows more effective control strategies to be implemented in real-time. As expected, the MSP results exhibited a deterioration as compared with the SSP results. The MSP is a fairly difficult problem, and novel methods must be developed. Nevertheless, experience with MSP algorithms used in control strategies indicates that even relatively inaccurate predictions could be effective in controller implementation because of the forgiving nature of feedback control [31].

V. Summary and Conclusions

In this study a neural network system is developed for predicting MPEG-coded video source traffic. This is an important and widely researched topic because it can lead to more efficient dynamic bandwidth management and, more recently, to better control of real-time multimedia streams resulting in improved QoS. In the first experiment, SSPs are implemented, and they are shown to achieve comparable and sometimes even better results than the results reported in the literature. This indicates that the problem appears to have nonlinearities, and future research should perhaps deal with alternate nonlinear prediction methods. In the second experiment, a smoothed and down-sampled form of the video sequence time-series is used, and SSP results appear equally good. Thus, a longer horizon forecast can be obtained with very little degradation in performance. In the final experiment, an MSP method is developed for the smoothed or averaged frame size series, but the results show some deterioration. More fundamental analysis and more novel methods are needed for the MSP problem of time-series with significant LTDs, as this has not been a very well researched topic.

References

[1] J. J. Bae and T. Suda, "Survey of traffic control schemes and protocols in ATM networks," *Proc. IEEE*, vol. 79, pp. 170–189, Feb. 1991.

[2] V. S. Frost and B. Melamed, "Traffic modeling for telecommunications networks," *IEEE Commun. Mag.*, vol. 32, pp. 70–81, Mar. 1994.

[3] M. Krunz and H. Hughes, "A traffic model for MPEG-coded VBR streams," in *Proc. Joint Int. Conf. Meas. Modeling Comput. Syst.*, May 1995, pp. 47–55.

[4] D. P. Heyman and T. V. Lakshman, "Source models for VBR broadcast-video traffic," *TIEEE/ACM Trans. Networking*, vol. 4, pp. 40–48, Feb. 1996.

[5] P. Bocheck and S. F. Chang, "A content based video traffic model using camera operations," in *Proc. IEEE Int. Conf. Image Process.*, Lausanne, Switzerland, Sept. 1996, pp. 817–820.

[6] M. Krunz and S. K. Tripathi, "On the characterization of VBR MPEG streams," *ACM SIGMETRICS Performance Eval. Rev.*, vol. 25, pp. 192–202, June 1997.

[7] M. Krunz and A. M. Makowski, "Modeling video traffic using M/G/∞ input processes: A compromise between Markovian and LRD models," *IEEE J. Select. Areas Communications*, vol. 16, pp. 733–747, June 1998.

[8] M. D. D. Amorim and O. C. M. B. Duarte, "A novel deterministic traffic model based on a two-level analysis of MPEG video sources," in *Proc. IEEE Global Telecommun. Conf.*, Sydney, Australia, Nov. 1998, pp. 696–701.

[9] P. Manzoni, P. Cremonesi, and G. Serazzi, "Efficient modeling of VBR MPEG-1 coded video sources," *IEEE/ACM Trans. Networking*, vol. 7, pp. 387–397, June 1999.

[10] N. D. Doulamis, A. D. Doulamis, G. E. Konstantoulakis, and G. I. Stassinopoulos, "Efficient modeling of VBR MPEG-1 coded video sources," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, pp. 93–112, Feb. 2000.

[11] A. M. Dawood and M. Ghanbari, "Content-based MPEG video traffic modeling," *IEEE/ACM Trans. Multimedia*, vol. 1, pp. 77–87, Mar. 1999.

[12] A. Chodorek and R. R. Chodorek, "An MPEG-2 video traffic prediction based on phase space analysis and its application to on-line dynamic bandwidth allocation," in *Proc. 2nd Eur. Conf. Universal Multiservice Networks*, Apr. 8–10, 2002, pp. 44–55.

[13] P. R. Chang and J. T. Hu, "Optimal nonlinear adaptive prediction and modeling of MPEG video in ATM networks using pipelined recurrent neural networks," *IEEE J. Selected Areas Commun.*, vol. 15, pp. 1087–1100, Aug. 1997.

[14] A. D. Doulamis, N. D. Doulamis, and S. D. Kollias, "Nonlinear traffic modeling of VBR MPEG-2 video sources," in *Proc. IEEE Int. Conf. Multimedia Expo.*, New York, July–Aug. 2000, pp. 1318–1321.

[15] ——, "Recursive nonlinear models for on line traffic prediction of VBR MPEG coded video sources," in *Proc. IEEE-INNS-ENNS Int. Joint Conf. Neural Networks*, Como, Italy, July 2000, pp. 114–119.

[16] ——, "An adaptable neural network model for recursive nonlinear traffic prediction and modeling of MPEG video sources," *IEEE Trans. Neural Networks*, vol. 14, pp. 150–166, Jan. 2003.

[17] A. M. Adas, "Supporting real time vbr using dynamic reservation based on linear prediction," Georgia Inst. Technol., Atlanta, GA, Report GIT-CC-95/26, Aug. 1995.

[18] ——, "Using adaptive linear prediction to support real-time VBR video under RCBR network service model," *IEEE/ACM Trans. Networking*, vol. 6, pp. 635–644, Oct. 1998.

[19] S. J. Yoo, "Efficient traffic prediction scheme for real-time VBR MPEG video transmission over high-speed networks," *IEEE Trans. Broadcasting*, vol. 48, pp. 10–18, Mar. 2002.

[20] O. Rose. (1995, Mar.) Index of /MPEG. Univ. Würzburg, Würzburg, Germany. [Online]. Available: http://www3.informatik.uni-wuerzburg.de/MPEG/.

[21] S. Haykin, *Neural Networks—A Comprehensive Foundation*, second ed. Upper Saddle River, NJ: Prentice-Hall, 1999.

[22] A. G. Parlos, K. T. Chong, and A. F. Atiya, "Application of the recurrent multilayer perceptron in modeling complex process dynamics," *IEEE Trans. Neural Networks, Special Issue on Dynamic Recurrent Neural Networks: Theory and Applications*, vol. 5, pp. 255–266, Mar. 1994.

[23] A. G. Parlos, O. T. Rais, and A. F. Atiya, "Multi-step-ahead prediction using dynamic recurrent neural networks," *Neural Networks*, vol. 13, pp. 765–786, Sept. 2000.

[24] A. G. Parlos, S. K. Menon, and A. F. Atiya, "An algorithmic approach to adaptive state filtering using recurrent neural networks," *IEEE Trans. Neural Networks*, vol. 12, pp. 1411–1432, Nov. 2001.

[25] ——, "An adaptive state filtering algorithm for systems with partially known dynamics," *J. Dyn. Syst., Meas. Contr.*, vol. 124, no. 3, pp. 364–374, Sept. 2002.

[26] R. M. Bharadwaj and A. G. Parlos, "Neural state filtering for adaptive induction motor speed estimation," *Mechanical Syst. Signal Process.*, Mar. 2003, to be published.

[27] P. Harihara, "Real-time implementation of a neural networks-based motor speed filter using a digital signal processor," M.S. thesis, Texas A&M Univ., College Station, TX, Dec. 2002.
[28] F. H. P. Fitzek and M. Reisslein. (2001, Jan.) MPEG-4 and H.263 video traces for network performance evaluation. [Online]. Available: http://www-tkn.ee.tu-berlin.de/research/trace/trace.html.
[29] ——, "MPEG-4 and H.263 video traces for network performance evaluation," Telecommunications Networks Group, Technical University of Berlin, Berlin, Germany, TKN Tech. Rep., Oct. 2000.
[30] J. Walter. (2001, Oct.) bttvgrab. [Online]. Available: http://www.garni.ch/bttvgrab/.
[31] A. G. Parlos, S. Parthasarathy, and A. F. Atiya, "Neuro-predictive process control using on-line controller adaptation," *IEEE Trans. Contr. Syst. Technol.*, vol. 9, pp. 741–755, Sept. 2001.

Aninda Bhattacharya received the B.E. degree in mechanical engineering from South Gujarat University, Surat, India, in 1997 and the M.S. degree in mechanical engineering from Texas A&M University, College Station, in 2002. He is currently pursuing the Ph.D. degree with the Mechanical Engineering Department at Texas A&M University.

He is a graduate research assistant with the Networked and Intelligent Machines Laboratory at Texas A&M. His research interests include control systems design, system identification, neural networks, communication networks, and adaptive multimedia networks.

Alexander G. Parlos (S'81–M'87–SM'92) received the S.M. degree in mechanical engineering, the S.M. degree in nuclear engineering, and the Sc.D. degree in automatic control and systems engineering, all from the Massachusetts Institute of Technology (MIT), Cambridge, in 1985, 1985, and 1986, respectively. He received the B.S. degree in nuclear engineering from Texas A&M University, College Station, in 1983.

He has been on the faculty at Texas A&M University since 1987, where he is currently an Associate Professor of mechanical engineering, with joint appointments in the Department of Nuclear Engineering and Department of Electrical Engineering. His applied research interests include the development of methods and algorithms for life-cycle health and performance management of various dynamic systems, with special emphasis to system condition assessment (or diagnosis), end-of-life prediction (or prognosis), and reconfigurable control. He has been involved with the particular application of these concepts to electro-mechanical systems and, more recently, to computer networks. His theoretical research interests involve the development of learning algorithms for recurrent neural networks and their use for nonlinear estimation and control. He has been involved with research and teaching in neural networks, multivariable control, and system identification, and he has conducted extensive funded research in these areas. His research has resulted in one U.S. patent, three pending U.S. patents, and 18 invention disclosures. He has co-founded a high-tech start-up company commercializing technology developed at Texas A&M. He has over 135 publications in journals and conferences.

Dr. Parlos has been serving as an associate editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS since 1994, and of the *Journal of Control, Automation, and Systems* since 1999. He has served as a technical reviewer to numerous professional journal and government organizations, and he has participated in technical, organizing, and program committees of various conferences. Dr. Parlos is a Senior Member of AIAA, a member of ASME, ANS, and INNS, and is a registered professional engineer in the State of Texas.

Amir F. Atiya (S'86–M'90–SM'97) was born in Cairo, Egypt, in 1960. He received the B.S. degree in 1982 from Cairo University and the M.S. and Ph.D. degrees in 1986 and 1991 from the California Institute of Technology (Caltech), Pasadena, all in electrical engineering.

From 1997 to 2001, he was a Visiting Associate at Caltech. Currently, he is an Associate Professor with the Department of Computer Engineering, Cairo University. He also held other positions in industry as well as in academia, such as in Texas A&M University, College Station, TX; QANTXX Corporation, Houston, TX; Tradelink, Chicago, IL; Simplex Risk Management, Hong Kong; and Countrywide, Los Angeles, CA. His research interests are in the areas of neural networks, learning theory, pattern recognition, Monte Carlo methods, time series analysis, and optimization theory. His most recent interests are the application of learning theory and computational methods to finance.

Dr. Atiya received the highly regarded Egyptian State Prize for Best Research in Science and Engineering, in 1994. He also received the Young Investigator Award from the International Neural Network Society in 1996. He has been an Associate Editor for IEEE TRANSACTIONS ON NEURAL NETWORKS since 1998. He is a Guest Co-editor of the special issue of IEEE TRANSACTIONS ON NEURAL NETWORKS on "Neural Networks in Financial Engineering." He served on the organizing and program committees of several conferences, most recent of which was the Computational Finance CF2000, New York, and the IEEE Conference on Computational Intelligence in Financial Engineering (CIFER-2003), Hong Kong, for which he was program co-chair.