Power and Performance Analysis of Network Traffic Prediction Techniques

Muhammad Faisal Iqbal and Lizy K. John
University of Texas at Austin
{iqbal, ljohn}@ece.utexas.edu

Abstract

We study power and performance characteristics of different traffic predictors for online one-step-ahead predictions. The goal is to identify a predictor with reasonable accuracy and low power consumption. Our experiments on a large number of real network traces indicate that Double Exponential Smoothing and Auto-Regressive Moving Average are low cost predictors with reasonable accuracy.

1 Introduction

Multicore processors are increasingly being used in networking applications in order to keep up with the growing amount of traffic and complexity of applications. FreeScale's p4080 [2], Intel IXP [5] and Tilera processors [3] are some examples of multicore processors being used in networking applications. These packet processing systems are designed and provisioned with enough resources to meet with the peak traffic load. But network traffic varies with time and reaches the peak value for only a small portion of time. Resources in these packet processing systems can be utilized more efficiently if the future traffic can be predicted accurately [15, 11]. For example, if we can accurately predict future traffic based on its past behavior, idle or low traffic times can be exploited to force the system into a low-power state. Accurate traffic prediction is of interest in many other applications such as congestion control, admission control and network bandwidth allocation. We use a large number of real network traces to study the predictability of network traffic and compare the power and performance characteristics of various on-line one-step-ahead predictors. We study three categories of predictors: Classic Time Series based predictors, Artificial Neural Networks based predictors [9, 7], and Wavelet Transform based predictors [13]. Our results indicate that Double Exponential Smoothing (DES) is a low cost predictor with reasonable accuracy. DES is a well known predictor for financial time series predictions but we are the first ones to use DES for network traffic prediction. Auto-Regressive Moving Average predictor also exhibits good prediction performance.

2 Traffic Prediction Techniques

2.1 Classic Time Series Predictors

Last Value (LV) predictor uses last observed value as prediction for the next interval. In *Windowed Moving Average (MA)* we use average of past n past observations as prediction for the next interval.

Predictor	Equation	Compute	Storage
LV	$X_{t+1} = X_t$	0	1R
MA	$X_{t+1} = \frac{1}{n}\sum_{i=0}^{n-1}(X_{t-i})$	(n-1)A+1S	q(n)
DES	$S_t = \alpha y_t + (1-\alpha)(S_{t-1} + b_{t-1})$, $b_t = \gamma(S_t - S_{t-1}) + (1-\gamma)b_{t-1}$, $X_{t+1} = S_t + b_t$	6M + 4A	4R
AR	$X_{t+1} = \sum_{i=0}^{n-1} C_i * P_{t-i}$	n(M + A)	q(n)+nR
ARMA	$X_{t+1} = \sum_{i=0}^{n-1} C_i * X_{t-i} + \sum_{i=0}^{m-1} \alpha_i * E_{t-i}$	(n+m)(M+A)	q(n)+q(m) (n+m)R

Table 1: compute and storage requirements of classic predictors

Double Exponential Smoothing (DES) gives exponential lower weights to older observations like Exponential smoothing but also caters for trends in the data. In *Auto-Regression (AR)* technique, a signal is regressed with itself to exploit the autocorrelation structure. *Auto-Regression Moving Average (ARMA)* uses a combination of moving average (MA) of previous error terms in addition to Auto-regression (AR) for making predictions. ARMA based traffic predictors have been studied previously by [10, 14].

2.2 Artificial Neural Network (ANN) based Predictors

Neural Networks learn the relationship between input and output by looking at training examples. ANN consists of functions called neurons. These neurons have connections to receive the inputs and they pass the output to other neurons through more connections. Each connection has a weight associated with it. These weights determine behavior of the ANN. These weights are learned during the training phase.

2.3 Wavelet based Predictors

The prediction using wavelets usually involves three steps namely wavelet decomposition, signal extension and signal reconstruction. Wavelet decomposition divides the signal into a low pass output called *Approximation* and a high pass output called *Detail*. The wavelet decomposition function can be applied recursively to the approximations to get further levels of approximations and details. At any level the original signal is sum of the approximation at that level plus details at all lower levels. i.e., for a level 3 decomposition of signal x, $x = a3 + d3 + d2 + d1$. A model (e.g., AR) is fitted on approximation and details which are extended by predicting the next values using this model. Finally the extended approximations and details are combined to get the predictions for the original time series.

3 Experimental Methodology

We use real network traces from Caida [6], University of Auckland [4] and Bellcore Research [1]. The original traces contain arrival times of every packet. The trace is divided into two parts. The initial 25% of trace constitutes training set and the remaining is used to test the prediction accuracy. The details of the set of traces used in this study are listed in Table 2. We use Normalized Mean Square Error to compare the performance of predictors. $NMSE = \frac{1}{\sigma^2}\frac{1}{M}\sum_{t=1}^{M}(X_t - \hat{X}_t)^2$ where X_t is the actual value, $\hat{X}_t$ is the predicted value and M is number of predictions. σ^2 is the variance of X_t during prediction. In case of a trivial predictor (a predictor which always predicts mean) NMSE=1. If NMSE$>$ 1, this means that the predictor is worse than the trivial. NMSE=0 in case of a perfect predictor.

Source	Traces Avail.	Traces Used	duration/trace
Caida	20	16	1 hour
University of Auckland	84	20	24 hours
Bellcore Research	4	4	1 million packets

Table 2: Summary of Network traffic traces used in this study

4 Results and Discussion

4.1 Accuracy of Predictors

We evaluated accuracy of all the prediction techniques described in Section 2. For each predictor, we present the results of best

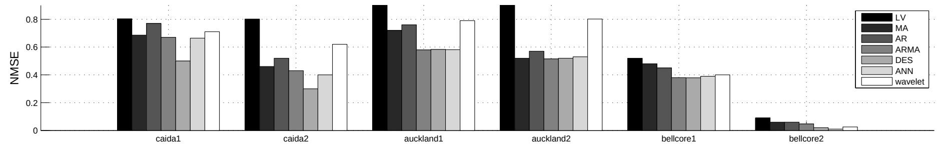

Figure 1: Normalized Mean Square Error of different predictors for Caida traces. Prediction interval used is 100 milli-Seconds

performing configurations of the predictors. In our simulations MA(8), AR(8) and ARMA(9,8) predictors performed the best. Also, we present the results of 3 layer ANN. The number of neurons in input layer is 8 and middle layer is 4. Results of wavelet based predictors using 2 level decomposition and db3 as mother wavelet are presented. We ran experiments on all the traces described in Table 2. Figure 1 shows accuracy of predictors on two sample traces from each source in Table 2. For all the predictors the NMSE value is less than 1, which means these traces are generally predictable. For Caida traces DES is a clear winner in terms of accuracy. For Auckland and Bellcore traces, ARMA is the best performing predictor. DES and ANN also perform comparably. For bellcore 2 trace, all of the predictors perform exceptionally well. This trace captures only external traffic and contains long periods of inactivity. So most of the predictors exhibit good behavior for this trace. It is also interesting to note that wavelet predictor does not perform well in most of the situations despite its high cost. This unsatisfactory performance of wavelet based prediction may be due to the effect of boundary conditions when applying wavelet transform to a finite length time series [9].

4.2 Power and Performance Overhead of Traffic Predictors

We implemented these predictors in software and measured the performance and energy overhead of these software predictors on a simple 2-issue processor. Table 3 shows instructions executed and energy consumed per prediction for different predictors. We focus only on power and performance overhead during the prediction phase. A predictor needs to be trained only once and that overhead can be ignored. In other situations, where traffic behavior changes over time, we may need to re-train the predictors. But this training is required very rarely as previous research has shown that traffic behavior remains steady over time [13, 14]. We used a one hour long trace and measured the

	LV	*MA*	*AR*	*ARMA*	*DES*	*ANN*	*Wave*
Instructions	225	258	269	307	230	417	590
u-Joules	21.6	24.7	25.9	29.5	22.8	39.4	93.1

Table 3: Per Prediction processing and energy of predictors

performance and power using GEMS [12] full system simulator integrated with Wattch [8] for power measurement. Table 3 shows instructions per prediction for each type of predictor when the predictors are implemented in software and corresponding energy per prediction. We see that ANN and wavelet based predictor requires considerably more instructions than other predictors. It is interesting to note that DES predictor has very small overhead yet it provides very good performance. We have seen from the performance results in Section 4 that ANN and ARMA also give very good performance results for most of the traces. But when comparing energy consumption, we can see that DES is the lowest power consuming predictor. It is also comparable in performance to the high cost predictors like ANN which makes this very useful for applications like one-step-ahead traffic prediction for power management. Energy consumption by ARMA predictor is also fairly low as compared to ANN and wavelet. Although ANN performs well in most situations, the power and performance cost associated with it make it suitable only for off-line applications like network design and capacity planning.

5 Conclusions

We have provided a performance and power comparison of three different classes of predictors using real network traces. Our results indicate that network traffic is generally predictable. Furthermore, the choice of predictor is dependent on the characteristics of the network. We found different predictors suitable for traces from different sources. Also, in power critical online applications DES and ARMA show promising accuracy with minimal energy overhead. ANN based predictor performed consistently well but has high power and computation overhead and thus maybe used in off-line studies.

6 Acknowledgements

This work was sponsored in part by Semiconductor Research Consortium under Task ID 2155.001 and National Science Foundation under grant NSF grant 1117895. The opinions and views expressed in this paper are those of the authors and not those of NSF or SRC.

References

[1] Bellcore traces. http://ita.ee.lbl.gov/html/contrib/BC.html.

[2] The p4080 processor. http://www.freescale.com/.

[3] The tilera processor family. http://www.tilera.com/products/processors.

[4] The univresity of auckland traces. http://wand.net.nz/wits/auck/2/.

[5] Intel ixp hardware reference manual, January 2003.

[6] K. Claffy, D. Andersen, and P. Hick. The caida anonymized 2011 internet traces.

[7] A. D. Doulamis, N. D. Doulamis, and S. D. Kollias. An adaptable neural-network model for recursive nonlinear traffic prediction and modeling of mpeg video sources. *IEEE Transactions on Neural Networks*, 14:150–166, 2003.

[8] D. B. et al. Wattch: A framework for architectural-level power analysis and optimizations. ISCA, 2000.

[9] H. Feng and Y. Shu. Study on network traffic prediction techniques. In *WiCom*, pages 1041–1044, 2005.

[10] J. Jiang and S. Papavassiliou. Enhancing network traffic prediction and anomaly detection via statistical network traffic separation and combination strategies. *Comput. Commun.*, 29, June 2006.

[11] R. Kokku, T. L. Riché, A. Kunze, J. Mudigonda, J. Jason, and H. M. Vin. A case for run-time adaptation in packet processing systems. *SIGCOMM Comput. Commun. Rev.*, 34:107–112, January 2004.

[12] M. M. K. Martin, D. J. Sorin, B. M. Beckmann, M. R. Marty, M. Xu, A. R. Alameldeen, K. E. Moore, M. D. Hill, and D. A. Wood. Multifacet's general execution-driven multiprocessor simulator (gems) toolset. CAN, 2005.

[13] Y. Qiao, J. Skicewicz, and P. Dinda. An empirical study of the multiscale predictability of network traffic. In *IEEE HPDC, 2003*.

[14] A. Sang and S. qi Li. A predictability analysis of network traffic. In *IEEE INFOCOM 2000*.

[15] Q. Wu and T. Wolf. On runtime management in multi-core packet processing systems. ANCS '08, pages 69–78.