Inter-Data-Center Network Traffic Prediction with Elephant Flows

Yi Li*, Hong Liu†, Wenjun Yang†, Dianming Hu†, Wei Xu*
* Tsinghua University, Beijing, China, 100084 {li-yi13, weixu}@mails.tsinghua.edu.cn
† Baidu Inc., Beijing, China, 100083 {liuhong03, yangwenjun, hudianming}@baidu.com

***Abstract*—With the ever increasing number of large scale Internet applications, inter data center (inter-DC) data transfers are becoming more and more common. Traditional inter-DC transfers suffers from both low-utilization and congestion, and traffic prediction is an important method to optimize these transfers.**

Inter-DC traffic is harder to predict than many other types of network traffic, because it is dominated by a few large applications. We propose a model that significantly reduces the prediction errors. In our model, we combine wavelet transform with artificial neural network (ANN) to improve prediction accuracy. Specifically, we explicitly add information of elephant flows, the least predictable yet dominating traffic in inter-DC network, into our prediction model. To reduce the amount of monitoring overhead for the elephant flow information, we added interpolation to fill in the unknown values in the elephant flows.

We demonstrate that we can reduce prediction errors over existing methods by 5%$\sim$10%. Our prediction is already in production at Baidu, one of the largest Internet companies in China, helping reducing the peak network bandwidth.

I. Introduction

The large scale and geographically distributed applications are on the rise. These applications, such as web search, video streaming and file sharing are commonly distributed to several data centers. Partitioning applications into multiple data centers can help reduce cost and improve service reliability.

All these applications can lead to heavy network traffic among the data centers. We call this type of traffic *inter-data center* (inter-DC) traffic to differentiate it from the traffic from end-users accessing these applications from the Internet (which we call *Internet traffic*). Many large service providers use dedicated fibers to handle inter-DC traffic [4]. Given the cost of inter-DC bandwidth, it is essential to keep the inter-DC links highly utilized.

Many Internet service providers (ISPs) charge for bandwidth by the peak bandwidth that a customer uses. Pure traffic shaping might be useful to reduce the peak bandwidth but it may hurt the performance of some critical applications (esp. when the priority is not configured correctly). Thus scheduling traffic with traffic engineering methods to reduce peak bandwidth of each link is important here to reduce costs. Existing work such as Google's B4 and Microsoft's SWAN uses software defined network (SDN) to accurately monitor and schedule the inter-DC data transfers. However, most conventional data centers do not have the infrastructure to support flow-level monitoring and scheduling, and thus relies on an accurate prediction of the future traffic to perform short-term / long-term traffic scheduling.

978-1-5090-0223-8/16/$31.00 © 2016 IEEE

With the high utilization of inter-DC links, spikes and fluctuations in the traffic can cause congestions, which are especially harmful to interactive applications. Accurate network traffic prediction is an important component for tasks like network resource provisioning, scheduling and traffic engineering [1], and thus traffic prediction has been a hot research topic. However, to our knowledge, there is no work taking the special inter-DC traffic patterns into account.

In this paper, we present our new model for predicting the network traffic on a inter-DC link at Baidu, one of largest Internet company in China. This link serves as Baidu's inter-DC backbone, connecting multiple data centers with tens of thousands of servers. These data centers host hundreds of large scale applications, both interactive and batch. We reduced the prediction errors by 5%$\sim$10%. Using our prediction method, Baidu is able to reduce the peak bandwidth for about 9% on average.

While researchers have proposed many network prediction models under different network environment, these models do not work well for inter-DC traffic prediction. There are several reasons why it is hard to predict inter-DC traffic:

First, inter-DC traffic neither represents linear processes nor has stable statistical properties, thus widely used linear models for time-series prediction, such as Autoregressive models (AR) [17], Autoregressive moving average models (ARMA) [18] and Autoregressive Integrated Moving Average models (ARIMA) [20] do not work well.

Second, inter-DC traffic exhibits different patterns compared to Internet backbone traffic. Studies have shown that data center traffic is bursty and unpredictable at such long time-scales (especially at 100 seconds or longer timescales) [2]. In fact, with the data we collected at Baidu, we can predict the Internet traffic 10 minutes ahead with only about 2% error from the real value. However, the inter-DC traffic prediction error is as high as 8% to 9%.

Third, the recurring patterns in inter-DC traffic are not obvious because this traffic is often generated by a small number of large applications. For example, in our case, the top 5 applications account for about 80% of the inter-DC traffic. The elephant flows generated by these applications usually occupy large portion of traffic [24] and impact more on the total traffic than mice flows. Usually, the number of elephant flows is far smaller than the number of mice flows, which is referred to as "the elephants and mice phenomenon" [8].

There are four key ideas in our prediction method.

First, we apply wavelet transform [22] to decompose the raw time domain traffic to capture both the time and frequency features. We apply Daubechie's 4 (Db4) wavelets with ten levels of decomposition [23] and show that it works well in reducing prediction errors.

Second, we put incoming and outgoing traffic together for training. Thus we can predict incoming and outgoing traffic using the same model.

Third, we recognize the contribution of elephant flows to the inter-DC traffic. We explicitly add information about elephant flows as separate feature dimensions in the prediction. A practical difficulty is that it is quite expensive to capture all elephant flow information frequently enough to help with the short term prediction. We use different interpolation methods to fill in the missing values of elephant flow traffic, which allow us to incorporate elephant flow information without introducing much data collection overhead.

Last but not least, as the patterns are highly non-linear, we use artificial neural network (ANN) to build the prediction model. ANN not only handles non-linearity well, but it also allows us to combine different features into the same model.

Note that both the features from wavelet transform and elephant flows can be regarded as decompositions. The wavelet transformation is an internal decomposition as we are decomposing the traffic time series using the series itself, while separating out the elephant traffic is an example of external decomposition using additional information. Combining the internal and external decomposition is the key for our prediction accuracy improvements.

We make the following three contributions:

1) We propose a network traffic prediction model for inter-DC traffic, a traffic type that is hard to predict using previous models, by treating the elephant flows explicitly. We show that by combining wavelet transform and artificial neural networks, we can significantly reduce prediction errors.

2) We introduce effective interpolation method to reduce the amount of expensive flow-level observations for the elephant flows.

3) We evaluate our model on a real world, massive scale inter-DC link with tens of thousands of servers and reduce prediction errors by 5%$\sim$10% over existing work.

The rest of this paper is organized as follows. Section II presents the researches on network traffic prediction in recent years. Section III describes our model. Section IV shows the experiment results, including comparisons between different strategies. We conclude in Section V.

II. Related Work

Many studies have been done on network traffic prediction with traditional linear models. Hu et al. [26] used Seasonal Trend Decomposition using Loess (STL) [21] to decompose original series into three components: season component, trend component and irregular component and then used X11-ARIMA for network traffic prediction. Yoo et al. [10] developed a model to support prediction on high-bandwidth network. FARIMA, known as autoregressive fractionally integrated moving average, which captures the characters of long-memory time series, is also widely used in traffic prediction [28]. Zhou et al. [27] combined ARIMA and GARCH, which is a non-linear model, to create a conditional mean and conditional variance model called ARIMA/GARCH, and compared the differences of the performance between ARIMA/GARCH and FARIMA. Periyanayagi et al. [29] proposed a time series model called S-ARMA, using Swarm intelligence and ARMA, for the network traffic prediction in wireless sensor networks. Wavelet transform have been used to preprocess series before the prediction with linear models [7], [32]. However, as inter-DC traffic is bursty and unpredictable at long-time scales, linear models are not suitable for inter-DC traffic prediction, especially for long-time-ahead prediction.

Learning methods are useful in network traffic prediction. Researchers have applied Support Vector Machine (SVM) based classification and regression for time series prediction. For example, Feng et al. [33] applied SVM for one-step-ahead prediction on WLAN and compared the performance for various prediction methods. Qian et al. [34] used Empirical Mode Decomposition (EMD) to reduce the noise in the data before applying SVM for prediction.

Another important and useful learning model for time-series prediction is artificial neural networks (ANNs) [15]. ANNs have the capability to do non-linear modeling and approximate any continuous function to any desired accuracy theoretically [19], thus ANNs can be used to predict complex time series. Some variants of ANNs have been proposed. For example, algorithms such as PSO [6] can be used to optimize the training process. We can also embed new tools such as wavelet transformation into a neural network, like [12] did. G. Peter Zhang [14] proposed a hybrid approach to time series forecast using both linear ARIMA model and the nonlinear ANN to predict complex series data with both linear and nonlinear correlation structures. Wavelet Neural Network (WNN) employs nonlinear wavelet basis functions to solve nonlinear fitting problems and have been used for traffic prediction [37]. Xiao et al. [36] studied fuzzy-neural network prediction models with wavelet decomposition. Alarcon-Aquino et al. combined maximal overlap discrete wavelet transform (MODWT) with ANNs and proposed a multi-resolution finite-impulse-response (FIR) neural-network-based learning algorithm, which would be suitable for capturing low- and high-frequency information as well as the dynamics of time-varying signals [13].

On our inter-DC traffic dataset, we experimented different prediction models, and did not find significant improvements on prediction accuracy. It is not coincidental: the inter-DC traffic is dominated by a combination of elephant flows, which demonstrates less patterns. In this work, instead of keep improving the prediction models, we focus on designing better features to capture the elephant flow information.

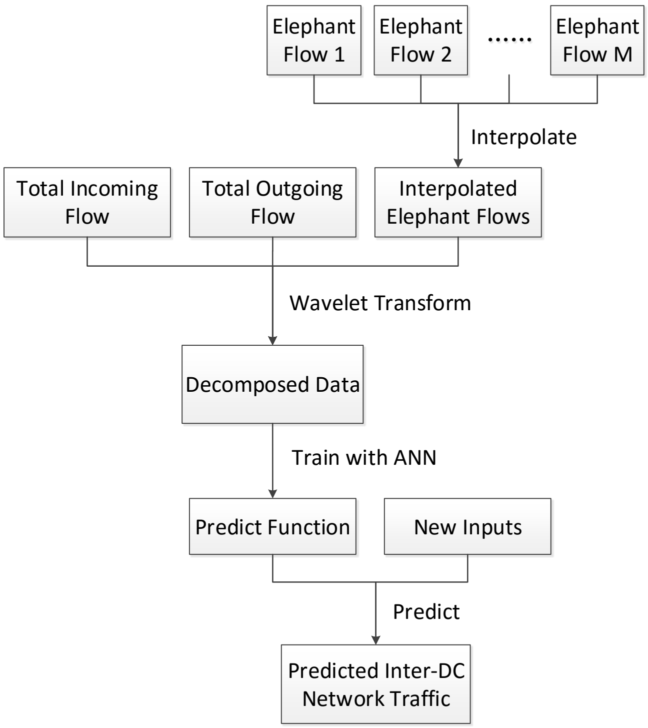

Fig. 1. The process flow of our model. After training, when we get a new data item, which contains the total incoming and outgoing traffic, the sampled or interpolated traffic data of elephant flows, we perform decomposition to it. Then we take the features of previous k steps, including this step, as the input of the predict function, and get the predicted future total traffic.

III. Model Overview

In our model, we collect the total incoming/outgoing traffic data and traffic data of elephant flows. As the traffic data of elephant flows is sampled less frequently than the total traffic, we use interpolation methods to construct the missing values so that we can align total traffic data samples with that of the elephant flows. Then we decompose the collected data with wavelet transform to reveal additional frequency information for training. After decomposition, we normalize the data and train it with ANN to get a prediction function. With the prediction function and new inputs, we can predict the total traffic data in near future. Figure 1 shows the process flow of our model.

A. Data Collection

We collect two types of data from each inter-DC link: the total traffic for both incoming and outgoing directions and a sample of elephant flows. Given a time series $(t_1, t_2, \ldots, t_n)$, We denote the total incoming/outgoing traffic at time t_i as in^i and out^i. To reduce useless information and improve the efficiency of computation, we only use information from the top M applications which account a great proportion of total traffic. We use a $2M$-dimentional vector to represent the raw elephant flow information at each sample time: $(ein_1, eout_1, ein_2, eout_2, \ldots, ein_M, eout_M)$, where ein_k and $eout_k$(where $k = 1, 2, \ldots, M$) denotes the number of incoming/outgoing traffic of the k-th largest application.

As the traffic data of elephant flows is sampled less frequently, we use interpolation methods to construct the missing values to roughly align the data sample of the total traffic and the elephant flow samples. Thus for each timestamp t_i, we get a $(2 + 2M)$-dimentional vector as our raw data: $(in^i, out^i, ein_1^i, eout_1^i, ein_2^i, eout_2^i, \ldots, ein_M^i, eout_M^i)$.

The goal of the prediction is that given all the history, we want to predict the traffic at different time points in the near future. Formally, we want to predict the next k-step total traffic tuple (in^{i+k}, out^{i+k}), where $k = 1, 2, \ldots$.

Note that we have an alternative approach to model the incoming and outgoing traffic separately, using two $(M+1)$-dimentional vectors for each. Intuitively, the incoming and outgoing traffic of a data center are highly correlated. Using a combined model can help us saving model training cost by about 40% while not affecting the prediction accuracy much. We compare these two models in Section IV.

B. Interpolation

The elephant flow data are sampled less frequently due to resource cost concerns. We construct the missing values using interpolation, a common method in numerical analysis. There are many interpolation methods. In this paper, we compare the following four methods.

One of the simplest methods is *zero interpolation*, which fills zeros for all unknown points. Surprisingly, even with this simple method, we can still significantly reduce the prediction errors compared to methods without using elephant flow information.

We call the second method *scale interpolation*. As the elephant flows occupy large part of the total traffic, we construct the missing values by filling in a number that is proportional to the total traffic. Given the total incoming traffic in^i and in^{i+s} at time t_i and t_{i+s} respectively, assume that the traffic data of elephant flows is sampled at that two time points but not sampled at t_{i+1}, t_{i+2}, $\ldots$, t_{i+s-1}. Then the unsampled incoming traffic $ein_k^{i+s'}$ $(0 < s' < s)$ of application k, the interpolation value, namely the incoming traffic of the elephant flow, is

$$ein_k^{i+s'} = ein_k^i \times \frac{in^{i+s'}}{in^i}.$$

Intuitively, this method may reduce the effectiveness of adding elephant flow information, as we “pollute” the elephant flow data with numbers that is highly correlated with the total traffic, and our experiments confirm the intuition.

The third method is *linear interpolation*. Using the same notations as above, the interpolation value is

$$ein_k^{i+s'} = ein_k^i + (t_{i+s'} - t_i) \times \frac{ein_k^{i+s} - ein_k^i}{t_{i+s} - t_i}$$

which means $(t_{i+s'}, ein_k^{i+s'})$ is a point in a line segment linking (t_i, ein_k^i) and (t_{i+1}, ein_k^{i+s}).

The last interpolation method we use is *spline interpolation*. With spline interpolation, we can get a smooth curve linking points. To make the interpolation error small and make the computation simple, we decide to use third order polynomials as interpolation functions (also known as *cubic spline interpolation*) [25].

Using interpolations allows us to use the elephant flow information while keeping the monitoring cost low. We evaluated all four kinds of interpolations and show the results in Section IV.

C. Decomposition

As we use learning algorithms to predict the traffic, we need "features" (in machine learning terminology) to capture the predictable information at each time point. We use decomposition to provide better features.

We decompose the raw data into new series using wavelet transform, which extract deeper information from the raw data. Wavelet transform is a powerful technique to analysis time series. Comparing to Fourier transform, wavelet transform has advantages in processing time-domain series data as it can reserve both time and frequency information while Fourier transform can only reserve frequency information. Wavelet transform uses wavelet functions to decompose time series. A wavelet is a function Ψ that is used to decompose the time series to a low-frequency part and a high-frequency part:

$$X(a,b) = \frac{1}{\sqrt{b}} \int_{-\infty}^{+\infty} x(t)\Psi(\frac{t-a}{b})dt$$

where a is the scaling parameter and b is the translation parameter. In practice, we use discrete wavelet (DWT) instead of continuous wavelet transform (CWT) as CWT computation is much more expensive than DWT. We recursively decompose the low-frequency part, adding one new series per recursive run.

Assuming we use wavelet transform with w levels of decomposition, we decompose each series in the raw data into $w+1$ new series. Assume the raw time series data of total incoming traffic is $(in^1, in^2, \ldots, in^n)$. Given a time point t and length $l(l \ll n)$, we decompose the time series data $s = (in^{t-l+1}, in^{t-l+2}, \ldots, in^t)$ using Db4 into $w+1$ series

$$s_1 = (in_1^{t-l+1}, in_1^{t-l+2}, \ldots, in_1^t)$$
$$s_2 = (in_2^{t-l+1}, in_2^{t-l+2}, \ldots, in_2^t)$$
$$\vdots$$
$$s_w = (in_w^{t-l+1}, in_w^{t-l+2}, \ldots, in_w^t)$$

Then we choose $(in_1^t, in_2^t, \ldots, in_w^t)$ as new features of time point t. Note that the relationship between in^t and the new features is

$$in^t = \sum_{i=1}^{w} in_i^t.$$

The new features can be regarded as decomposition of the raw value and contain the relationship information between the value and the old values. As we can see, each raw dimension is decomposed into w dimensions. Now we have $2+2M$ time series data, where M is the number of applications generating elephant flows. With decomposition, we get $(2+2M)\times(w+1)$ - dimentional features for each time point. We denote the new features by f_i. We can choose the parameter l heuristically. In our experiment, we find that $l = 60$, or using 30 minutes of data for decomposition, provides good results.

We then normalize the data before training. The goal of normalization is to scale the data to a given bound. Data normalization can help the learning algorithms avoid computational problems and facilitate network learning [19]. We use z-score [40] to standardize the series data. The z-score is defined as

$$z = \frac{x-\mu}{\sigma}$$

where x is the raw data to be scaled, μ is the mean of dataset and σ is the standard deviation of the dataset.

D. Prediction

We train the normalized data with Artificial Neural Networks (ANNs). ANNs are inspired by biological neural networks. Generally, an ANN consists of multiple layers, including an input layer, a number of hidden layers and an output layer. ANNs can capture non-linear characters and find complex relationships between inputs and outputs. ANNs are widely used in function approximation, classification, data processing and robotics [38]. The architecture (e.g. the number of layers, the number of nodes in each layer and so on) of an ANN and optimization algorithms used can affect the final training results.

As usual, we need to specify features and labels for training. Without loss of generality, a data item can be represented as $d_i = (\boldsymbol{f}_i, \boldsymbol{l}_i)$, where $\boldsymbol{f}_i$ stands for the feature vector while $\boldsymbol{l}_i$ stands for the label vector. Usually, we first train a dataset to get a predict function. Then we can predict the labels ($\boldsymbol{l}_i$) with the function and the features ($\boldsymbol{f}_i$). As mentioned above, by decomposing the total traffic data and the traffic data of elephant flows, we get $(2+2M)\times w$ new features, denoted by f_i, for each time point. Obviously, we should use previous data to predict in^i and out^i. Assume we use the data of k previous steps for one-step-ahead, then we have

$$\boldsymbol{f}_i = [f_{i-k+1}, f_{i-k+2}, \ldots, f_i]$$

$$\boldsymbol{l}_i = [in^{i+1}, out^{i+1}]$$

As to multiple-step-ahead prediction, we just need to replace each element of $\boldsymbol{l}_i$ with the corresponding one (e.g. $[in^{i+2}, out^{i+2}]$ for two-step-ahead prediction and $[in^{i+s}, out^{i+s}]$ for s-step-ahead prediction). Thus $\boldsymbol{f}_i$ is a vector of length $(2+2M)\times w \times k$. This means that when we get a predict function F, we pass the $(2+2M)\times w\times k$ features derived from the k previous steps as input to F and get the predicted in^i and out^i.

E. Measure Prediction Errors

We use Relative Root-Mean-Squared Error (RRMSE) to measure prediction errors. It is calculated as follows:

$$RRMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\frac{\hat{\theta}_i - \theta_i}{\theta_i})^2},$$

where $\hat{\theta}_i$ is the predicted value and θ_i is the raw value. We can see that it is unitless and can reflect variance and bias at the same time [39].

IV. Experimental Results

We first describe the dataset we use in the evaluation. Then we show that we can achieve significant prediction error reduction over existing methods. Finally we provide details on the effects of different methods and parameters in our prediction model.

A. Experiment Setup

We collect the inter-DC network traffic data from a production data center with tens of thousands of servers from Baidu for six weeks. The total incoming/outgoing traffic data is direct snapshots of the counters on the data center edge routers using SNMP, and we collect a number for both directions every 30 seconds. We use the data of the last day for testing and the rest for training. Figure 2 shows the total incoming and outgoing traffic of the data center. Due to confidentiality concerns, we normalize the Y-axis of all figures so we do not reveal the actual amount of data transfers. The normalization does not affect the results of this paper.

We use tags, such as source and destination IPs, port, protocol ids, type of service and input/output interface, to identify a flow. We collect the number of packets each flow contributes during a certain period of time and then calculate the average traffic. We sample the flow statistics every five minutes (comparing to the 30 seconds sampling rate for the total traffic) due to the limit of computation and storage resource. We observe the distribution of the traffic and see that the elephant flows from the top-5 applications dominate the traffic, which account for about 80% of the total traffic, as Figure 3 shows. From Figure 3, we can see that the traffic of the chosen elephant flows displays a substantial, but not perfect correlation with the traffic of the total flows.

In our experiment, we use one day data as test data. As we take a sample every 30 seconds, there are 2880 values we are predicting for the day. We then perform 30-second-ahead, 1-minute-ahead, 5-minute-ahead, 10-minute-ahead, 15-minute-ahead and 20-minute-ahead prediction and compare the differences between different strategies in each case.

B. Overall Prediction Measurement

We compare our model with two well-known models: one is a representative traditional linear model ARIMA [20], the other is ANN without wavelet transform and interpolation.

In the following evaluation, we use one input layer, one hidden layer and one output layer for the artificial neural network. We have also evaluated the prediction accuracy using more than one hidden layers and did not find much difference. We use Stochastic Gradient Descent (SGD) [11] as the optimization algorithm for model training. To include elephant flow data, we use zero interpolation method. We will evaluate other interpolation methods in the next section.

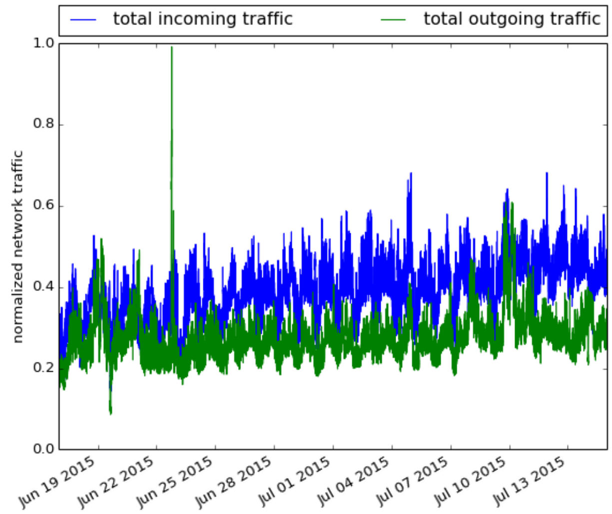

Fig. 2. The total incoming/outgoing traffic.

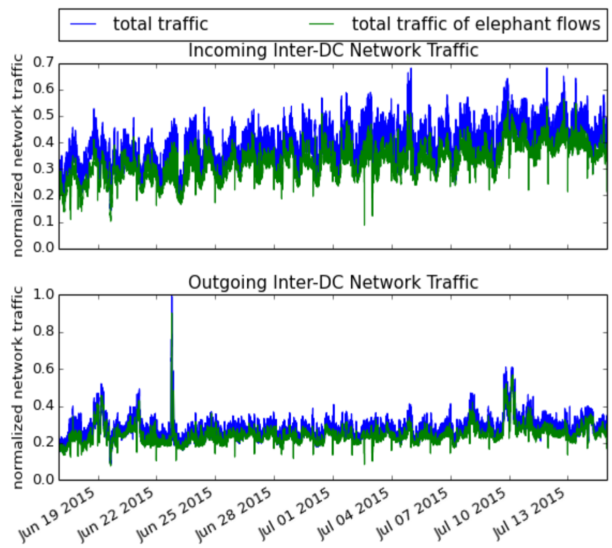

Fig. 3. Correlations between total traffic and the total traffic of the elephant flows. The elephant flows occupy a large portion of the total traffic.

Table I, II and Figure 4 show the comparison results. We can see that the linear model ARIMA performs the best for very short prediction, such as 30-second-ahead and 1-minute. That is because during very short time period, the long term patterns play a less important role than the short-term patterns, which are best captured by the linear models.

However, for 5-minute-ahead or longer time predictions, the non-linear and longer-term patterns prevail. We show that our model reduces the prediction errors by about 8.5% for

TABLE I
PREDICTION ERRORS (RRMSE) FOR INCOMING TRAFFIC

	30s	1min	5min	10min	15min	20min
ANN	0.0439	0.0525	0.080	0.096	0.105	0.113
ARIMA	0.0398	0.0496	0.0793	0.0971	0.111	0.119
Ours	0.0415	0.0496	0.0749	0.0900	0.0993	0.106

TABLE II
PREDICTION ERRORS (RRMSE) FOR OUTGOING TRAFFIC

	30s	1min	5min	10min	15min	20min
ANN	0.0439	0.0522	0.0808	0.0967	0.1089	0.117
ARIMA	0.0396	0.0492	0.0795	0.0980	0.112	0.122
Ours	0.0434	0.0517	0.0765	0.0913	0.102	0.110

incoming traffic and 6.9% for outgoing traffic, compared to linear models. Also, comparing to the conventional ANN, our model reduces prediction errors by 5.8% and 4% in average for incoming and outgoing traffic, respectively.

The accuracy improvement is essential for production: using the improved prediction results as guidance for traffic scheduling, Baidu is able to reduce the peak inter-DC link utilization (the ISP's billed utilization) by about 9%. The actual implementation of the prediction-based traffic scheduling system is out of the scope of the paper and thus omitted here.

C. Effect of Different Factors in Our Model

The prediction error reduction is the result of a combination of different methods and parameters. We evaluate the effects of the key components in our model.

1) Length of Training Set: Intuitively, using longer history as training set can help reduce the data noise and thus reduce prediction errors, to a certain point. A large training set may be of little use while bringing in extra and unnecessary training cost. Our evaluation confirms this intuition.

Fig. 4. Prediction error reduction over ARIMA and ANN. Positive numbers mean that we reduce the prediction errors actually while negative numbers mean the opposite. We can see that our model reduces prediction errors significantly for long-time-ahead prediction.

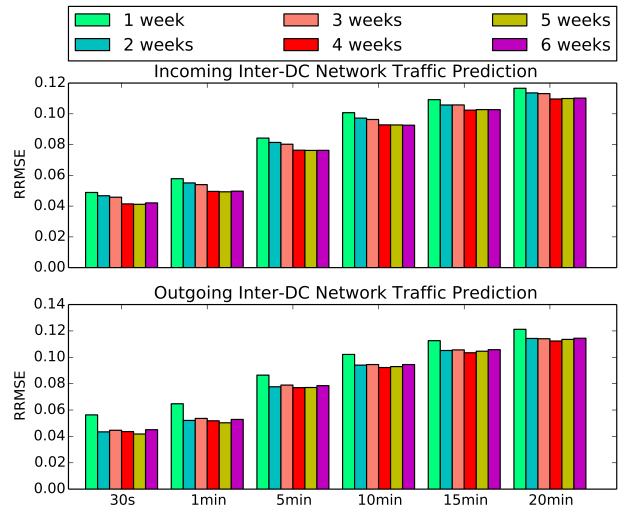

Fig. 5. Prediction errors of the models with different training set sizes.

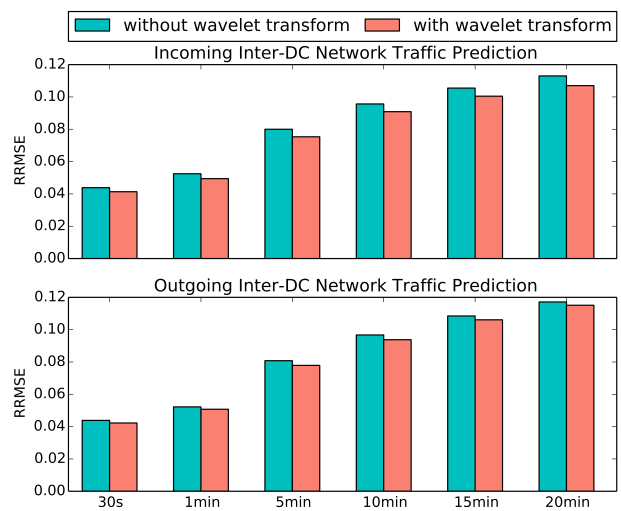

Fig. 6. The reduction in prediction errors using wavelet transform.

Thus we need to balance the advantages with the disadvantages of increasing the training set size. We compare the performance of different training set sizes. Figure 5 shows that using a training histroy of longer than 4 weeks, we can obtain a good enough model. We are still evaluating if it is related to a regular monthly pattern, collecting data for a much longer term, which is an important future work for us.

2) Effectiveness of Wavelet Transform: We use Daubechie's 4 (Db4) wavelets with ten levels of decomposition, as [23] did. For each time point, we decompose the subseries consisting of 60 values (including the current one) to get 11 new feature as Section III-C describes. We use a 4-week history for training. Figure 6 compares the prediction errors with and without wavelet transform.

Wavelet transform is an essential preprocessing step: for different steps prediction, the wavelet transform reduces the average prediction errors by 5.4% and 2.9% for incoming and

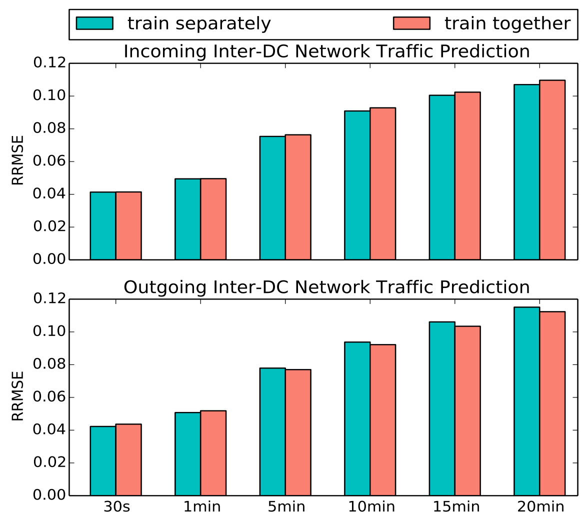

Fig. 7. Prediction errors: combining incoming flows and outgoing flows vs. training separate model for them. We can see that they provide similar results.

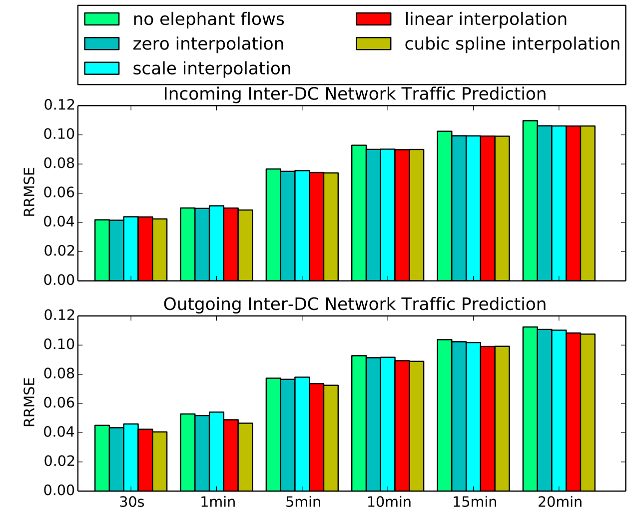

Fig. 8. Prediction errors of the models using different kinds of interpolations.

outgoing traffic. Intuitively, learning methods like ANNs work because they capture the (non-linear) correlations among multiple dimensions of data. Wavelet transform adds dimensions representing the reoccurring patterns of the data and reveals another level of important correlations. Thus the combination of wavelet transform and ANN brings a significant prediction error reduction.

3) Combing Incoming/Outgoing Traffic in The Same Model: As we discussed in Section III-A, we can either train separate models for incoming and outgoing traffic, or we can combine both traffic numbers into the same model. This is a key benefit of using learning methods like ANN – we have the flexibility to combine prediction models without changing to the model itself. Here we compare the result of these two alternatives.

Figure 7 shows the comparison results. There is no significant difference in prediction accuracy. This is as expected because the incoming and outgoing traffic are highly correlated.

It is beneficial to use the combined model. The single model is not only easier to implement and maintain, but also it needs less time to train comparing to the two separate models. In our experiments, using the combined model approach reduces the training time by about 40% comparing with the separate models.

4) Elephant Flows: The elephant flows play an important role in our model. Figure 8 shows the results of adding elephant flow information using different interpolation methods. We have the following observations from the figure.

First, elephant flow information reduces the prediction errors. For both incoming and outgoing traffic, adding elephant flows information reduces prediction errors, especially for the 5-minute or longer time ahead prediction.

Second, different interpolation methods have similar effects, except for the scale interpolation. As we have discussed in Section III-B, as ANN works on the correlations among different dimensions, the assumed correlation between the traffic of elephant flows and total traffic actually negatively affects the power of ANN. Interpolation methods that consider the neighbor values (e.g. the linear or cubic interpolation) perform slightly better than zero interpolation, which is as expected.

Given the good balance between simplicity and performance of zero interpolation, we choose it as our interpolation method in production.

Third, we observe that the more accurate number of elephant flow is, whether the measurement comes from interpolation or actual measurements from flow sampling, the better the overall prediction accuracy is.

Intuitively, the wavelet transform and ANN capture all the reoccurring patterns of the total traffic, but the elephant flows contribute to the overall traffic in a much more random way. We use the traffic data of elephant flows to “calibrate” the total traffic prediction, and thus the accuracy of elephant flows plays an important role. As an on-going future work, we are improving our elephant flow monitoring system to provide more frequent measurements.

V. Conclusion and Future Work

We propose a new model for inter-DC network traffic prediction. Different from normal network traffic, inter-DC traffic are dominated by a few large applications producing elephant flows. We can view the traffic as a combination of reoccurring patterns and some large noise.

The key for the traffic prediction is decomposing the various components from the combined traffic pattern. We decompose the traffic in two ways: first we use Db4 wavelet transform to decompose the time domain traffic data. Then we also add explicit elephant flow information. The elephant flow information provides multiple calibration points that significantly reduce the prediction errors, especially for 5-minute or longer time ahead prediction.

We emphasize on practical issues in the prediction model design, especially the cost of measurements. We show that we can significantly reduce the flow sampling overhead using interpolation methods. We also evaluate the possibility of reducing the training overhead by combining both incoming and outgoing traffic into the same model, reducing the training overhead by 40%. Our prediction method can help Baidu reduce the peak bandwidth for about 9% on average. The monetary cost reduction is significant for large scale inter-DC network. Thus the accuracy improvement is necessary and worthwhile.

As future work, we will extend the prediction to a longer time periods (weeks to months) to support tasks like resource provisioning. We will also explore models to predict traffic on multiple inter-DC links, as well as the traffic on core switches within a data center. On the engineering side, we are improving the technique to elephant flows traffic at a higher frequency.

ACKNOWLEDGEMENT

This research is supported in part by the National Natural Science Foundation of China grants 61361136003, 1000 Talent Plan grant, Tsinghua Initiative Research Program grants 20151080475 and a Google Faculty Research Award.

REFERENCES

[1] Wang, Hao, et al. "COPE: traffic engineering in dynamic networks." ACM SIGCOMM Computer Communication Review 36.4 (2006): 99-110.
[2] Benson, Theophilus, et al. "MicroTE: Fine grained traffic engineering for data centers." Proceedings of the Seventh Conference on emerging Networking Experiments and Technologies. ACM, 2011.
[3] Hong, Chi-Yao, et al. "Achieving high utilization with software-driven WAN." ACM SIGCOMM Computer Communication Review. Vol. 43. No. 4. ACM, 2013.
[4] Jain, Sushant, et al. "B4: Experience with a globally-deployed software defined WAN." ACM SIGCOMM Computer Communication Review. Vol. 43. No. 4. ACM, 2013.
[5] Ho, S. L., M. Xie, and T. N. Goh. "A comparative study of neural network and Box-Jenkins ARIMA modeling in time series prediction." Computers & Industrial Engineering 42.2 (2002): 371-375.
[6] Cheng, Wei, and Peng Feng. "Network traffic prediction algorithm research based on PSO-BP neural network." 2015 International Conference on Intelligent Systems Research and Mechatronics Engineering. Atlantis Press, 2015.
[7] Feng, Huifang, and Yantai Shu. "Study on network traffic prediction techniques." Wireless Communications, Networking and Mobile Computing, 2005. Proceedings. 2005 International Conference on. 2005.
[8] Papagiannaki, Konstantina, et al. "A pragmatic definition of elephants in internet backbone traffic." Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment. ACM, 2002.
[9] Hinton, Geoffrey E., Simon Osindero, and Yee-Whye Teh. "A fast learning algorithm for deep belief nets." Neural computation 18.7 (2006): 1527-1554.
[10] Yoo, Wuchert, and Alex Sim. "Network bandwidth utilization forecast model on high bandwidth networks." Computing, Networking and Communications (ICNC), 2015 International Conference on. IEEE, 2015.
[11] Bottou, Lon. "Large-scale machine learning with stochastic gradient descent." Proceedings of COMPSTAT'2010. Physica-Verlag HD, 2010. 177-186.
[12] Zhang, Run, Yinping Chai, and Xing-an Fu. "A network traffic prediction model based on recurrent wavelet neural network." Computer Science and Network Technology (ICCSNT), 2012 2nd International Conference on. IEEE, 2012.
[13] Alarcon-Aquino, Vicente, and Javier Barria. "Multiresolution FIR neural-network-based learning algorithm applied to network traffic prediction." Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on 36.2 (2006): 208-220.
[14] Zhang, G. Peter. "Time series forecasting using a hybrid ARIMA and neural network model." Neurocomputing 50 (2003): 159-175.
[15] Yegnanarayana, B. Artificial neural networks. PHI Learning Pvt. Ltd., 2009.
[16] Liu, Jie, and Ya-Lou Huang. "Nonlinear network traffic prediction based on BP neural network." Jisuanji Yingyong Journal of Computer Applications 27.7 (2007): 1770-1772.
[17] Akaike, Hirotugu. "Fitting autoregressive models for prediction." Annals of the institute of Statistical Mathematics 21.1 (1969): 243-247.
[18] Torres, Jose Luis, et al. "Forecast of hourly average wind speed with ARMA models in Navarre (Spain)." Solar Energy 79.1 (2005): 65-77.
[19] Zhang, Guoqiang, B. Eddy Patuwo, and Michael Y. Hu. "Forecasting with artificial neural networks:: The state of the art." International journal of forecasting 14.1 (1998): 35-62.
[20] Contreras, Javier, et al. "ARIMA models to predict next-day electricity prices." Power Systems, IEEE Transactions on 18.3 (2003): 1014-1020.
[21] Cleveland, Robert B., et al. "STL: A seasonal-trend decomposition procedure based on loess." Journal of Official Statistics 6.1 (1990): 3-73.
[22] Burrus, C. Sidney, Ramesh A. Gopinath, and Haitao Guo. "Introduction to wavelets and wavelet transforms: a primer." (1997).
[23] He, Haibo, and Janusz Starzyk. "A self-organizing learning array system for power quality classification based on wavelet transform." Power Delivery, IEEE Transactions on 21.1 (2006): 286-295.
[24] KAWAHARA, Ryoichi, and Shozo NAITO. "On the characteristics of internet traffic variability: Spikes and Elephants." IEICE TRANSACTIONS on Information and Systems 87.12 (2004): 2644-2653.
[25] De Boor, Carl. "A practical guide to splines." Mathematics of Computation (1978).
[26] Hu, Kejia, et al. "Estimating and forecasting network traffic performance based on statistical patterns observed in SNMP data." Machine Learning and Data Mining in Pattern Recognition. Springer Berlin Heidelberg, 2013. 601-615.
[27] Zhou, Bo, et al. "Network traffic modeling and prediction with ARIMA/GARCH." Proc. of HET-NETs Conference. 2005.
[28] Granger, Clive WJ, and Roselyne Joyeux. "An introduction to long-memory time series models and fractional differencing." Journal of time series analysis 1.1 (1980): 15-29.
[29] Periyanayagi, S., and V. Sumathy. "S-ARMA model for network traffic prediction in wireless sensor networks." J. Theor. Appl. Inf. Technol 60 (2014): 524-530.
[30] Riedi, Rudolf H., et al. "A multifractal wavelet model with application to network traffic." Information Theory, IEEE Transactions on 45.3 (1999): 992-1018.
[31] Zhao, Hong, and Nirwan Ansari. "Wavelet Transform-based Network Traffic Prediction: A Fast On-line Approach." CIT. Journal of Computing and Information Technology 20.1 (2012): 15-25.
[32] Tan, Xiaoling, Weijian Fang, and Yong Qu. "Network Traffic Prediction Algorithm based on Wavelet Transform." International Journal of Advancements in Computing Technology 5.5 (2013).
[33] Feng, Huifang, et al. "SVM-based models for predicting WLAN traffic." Communications, 2006. ICC'06. IEEE International Conference on. Vol. 2. IEEE, 2006.
[34] Qian, Yuan, et al. "Network traffic forecasting by support vector machines based on empirical mode decomposition denoising." Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on. IEEE, 2012.
[35] Wu, Zhaohua, and Norden E. Huang. "Ensemble empirical mode decomposition: a noise-assisted data analysis method." Advances in adaptive data analysis 1.01 (2009): 1-41. APA
[36] Xiao, Heng, et al. "Fuzzy-neural network traffic prediction framework with wavelet decomposition." Transportation Research Record: Journal of the Transportation Research Board 1836 (2003): 16-20.
[37] Zhang, Run, Yinping Chai, and Xing-an Fu. "A network traffic prediction model based on recurrent wavelet neural network." Computer Science and Network Technology (ICCSNT), 2012 2nd International Conference on. IEEE, 2012.
[38] https://en.wikipedia.org/wiki/Artificial_neural_network
[39] Mouillot, David, and Alain Lepretre. "A comparison of species diversity estimators." Researches on Population Ecology 41.2 (1999): 203-215.
[40] Cheadle, Chris, et al. "Analysis of microarray data using Z score transformation." The Journal of molecular diagnostics 5.2 (2003): 73-81.