Web Traffic Time Series Forecasting

Kashish Oberoi
Computer Science Dept.

PES Univeersity
Bangalore, India

Kashishoberoi00@gmail.com

Atharv Verma
Computer Science Dept.

PES Univeersity
Bangalore, India

atharverma@gmail.com

K. Ramakrishnan
Computer Science Dept.

PES Univeersity
Bangalore, India

Ramakrishnanramakrishnan8@gmail.com

Abstract—**With the recent advancements in internet technology, there is an urge to effectively forecast web traffic. The forecast will help the web servers to handle web requests efficiently. Here, we are using the Web Traffic Time Series Forecasting dataset by Google to predict future traffic of Wikipedia articles [1]. Here we approach this problem by using Ensemble Learning to boost the performance of the model. We have observed that different time series model give better results for different time series. We plan to use ensemble learning approach for different classical models, RNN/LSTM based models and Prophet Models. In addition to that we want to reduce the noise/outliers by taking into account the variance and fluctuations in the data. This approach will help us forecast better results and it will be helpful for load management of data servers for web domains.**

I. Introduction

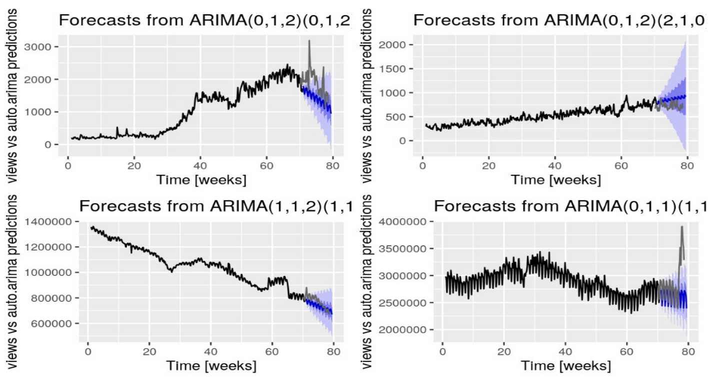
In recent times with the advancement of internet technology and widespread availability of internet connection, forecasting the future web traffic has become a need of the hour. As the traffic within a company's network or website increases, the strain on data center servers grows. Each request to access applications or information from a server adds to the overall processing capacity that it is able to handle. This increase in user access continues to add up until, ultimately, the server cannot handle any more traffic, and crashes. Organizations can avoid this added server strain and potential data center collapse with a responsive server load balancer. Server load balancing is a way for servers to effectively handle high-volume traffic and avoid decreased load times and accessibility problems.
This can be achieved by forecasting the web traffic for a particular website or domain so that, adequate amount of servers are available for load management and high traffic does not affect the performance of the domain.

II. Previous work

A. *The ARIMA (Autoregressive integrated moving average)*[2]

ARIMA, short for 'AutoRegressive Integrated Moving Average', is a forecasting algorithm based on the idea that the information in the past values of the time series can alone be used to predict the future values. ARIMA models are used because they can reduce a non-stationary series to a stationary series using a sequence of differencing steps. On applying the difference operator to a random time series {xt}(a non-stationary series) we are left with white noise {wt}(a stationary series):
∇xt=xt−xt−1=wt

ARIMA essentially performs this function, but does so repeatedly, d times, in order to reduce a non-stationary series to a stationary one. Any 'non-seasonal' time series that exhibits patterns and is not a random white noise can be modelled with ARIMA models.A time series {xt}is an autoregressive integrated moving average model of order p, d, q, ARIMA(p,d,q), if ∇dxt is an autoregressive moving average of order p,q, ARMA(p,q). That is, if the series {xt}is differenced d times, and it then follows an ARMA(p,q) process, then it is an ARIMA(p,d,q) series where auto-regressive / p: we are using past data to compute a regression model for future data.
The parameter p indicates the range of lags; e.g. ARIMA(3,0,0) includes t-1, t-2, and t-3 values in the regression to compute the value at t. Integrated / d: this is a differencing parameter, which gives us the number of times we are subtracting the current and the previous values of a time series. Differencing removes the change in a time series in that it stabilises the mean and removes (seasonal) trends. This is necessary since computing the lags (e.g. difference between time t and time t-1) is most meaningful if large-scale trends are removed. A time series where the variance (or amount of variability) (and the autocovariance) are time-invariant (i.e. don't change from day to day) is called stationary.previous error terms to include in the regression error of the model.
A pure Auto Regressive (AR only) model is one where Yt depends only on its own lags. That is, Yt is a function of the 'lags of Yt'. Likewise, a pure Moving Average (MA only) model is one where Yt depends only on the lagged forecast errors. An ARIMA model is one where the time series was differenced at least once to make it stationary and you combine the AR and the MA terms.
Arima Model:
Predicted Yt = Constant + Linear combination Lags of Y (upto p lags) + Linear Combination of Lagged forecast errors (upto q lags)

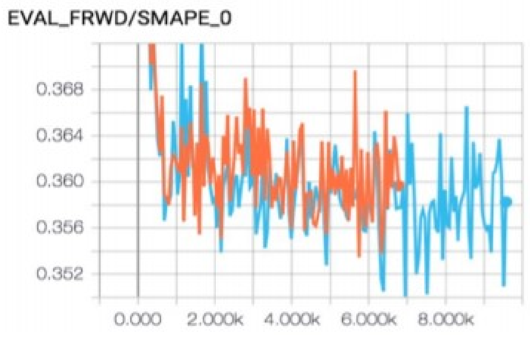


The results are not too bad, actually. Especially the lower left plot. Results show a downturn in the upper left plot. The upper right plot is a challenging problem, because the levelling of the viewer numbers at the end of the time range was not predictable from the previous behaviour. The same is true for the large spike in the lower right plot.
Other Drawbacks of this approach are –

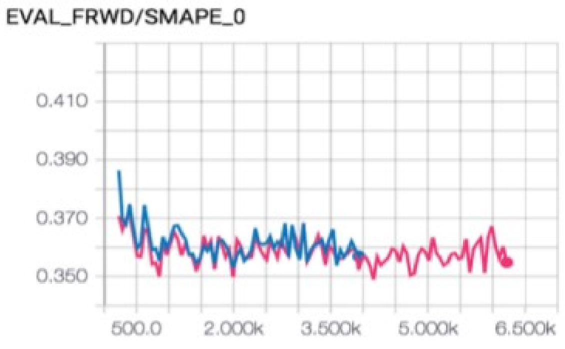
- The model trained has inevitably high variance, due to very noisy input data. This could be rectified by providing a better dataset that accounts for the reason of high variance.
- Same model trained on different training datasets can have varying performance due to irregularity of dataset.
- The model might diverge (instead of converging) from the actual trend due to the noisiness of the training dataset.
- Wild fluctuations in performance going from step to step were recorded while training the model.
- RMSE might provide misleading error. It is better to use other measures for estimating accuracy.

B. *RNN seq2seq model*[3]

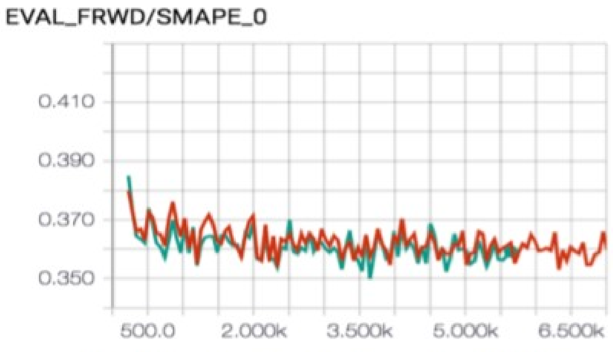
This paper presents a forecasting model to predict the web traffic. The prediction of web traffic can help the site owner in many ways, including better understanding of the user behaviour and determine an efficient strategy for load-balancing for future loads. This has made forecasting an active area of research. The model in this paper focuses on the temporal observations that emerge from analysis and forecast.
The model assumes that the medians considered over a window of exponentially varying sizes, and their median may also affect the forecast and includes them in the features of the input to the model. The model aims at bringing out the relationship between the magnitude of the features used for prediction model, its overall performance and the optimal number of features required while maintaining high accuracy prediction results.
The data, Google's Web Traffic Time series forecasting, was obtained from Kaggle. It was then cleaned since the missing values were replaced with 0. The model was rebuilt from the RNN seq2seq model, using the encoder decoder architecture where cuDNN GRU was used to encode and TensorFlow GRUBlockCell as decoder with the output of the decoder passed on to the next step until the end of the sequence.
Rather than considering the entire time period, the model considers divided windows of time. The median in each time window is added as the new feature. A rolling window was used to provide more weightage to more recent data. The window started off with a smaller window size of 6 and followed the Fibonacci sequence to increase the size of the window. This rolling window was used to find the seasonality and trend.



(a) Orange-Existing Model; Blue-New Model



(b) Blue – Existing Model; Pink – New Model



(c) Red-Existing Model; Green-New Model

The new model performed better than the previous model which is evident from the SMAPE: improved 0.351 to 0.349 for 804 days and 0.354 to 0.351 for 740 days in the training set. The model however, considers only 4 pages and does not incorporate the relations among the pages. In our point of view, the model may be sensitive external factors like spikes in traffic which could have occured during a DDoS attack.

Model and its features	**Training set(days)**	**SMAPE**
Existing model	804	0.351
	740	0.354
Model with only 7 day median and page popularity	804	0.350
Model with 7, 30, 90, 180 days median as independent features	804	0.349
Model with median of medians with Fibonacci sequence of variable window length	804	0.349
Model with all the newly added features	740	0.351

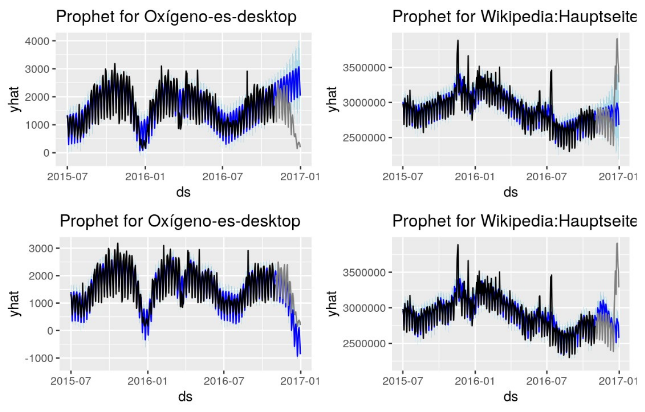
C. *Prophet(open source software released by Facebook's Core Data Science team.)*[4]

Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well.

In this approach emphasis is given to yearly seasonality which suggested an overall decline in views towards the middle of the year. The parameter changepoint.prior.scale adjusts the trend flexibility. This is a hyperparameter and was taken to be 0.5 to ensure optimum results as with an increase in trend flexibility the model was seen to overfit to noise which hindered the results.

Enabling prophet to recognise long-term seasonal variations in the data is crucial for a successful forecasting of our time series data. To demonstrate this, the following two sample curves: the German main page and the entry for Oxygen in the Spanish Wikipedia. The upper row of plot shows forecasts without a seasonal component and the presence of this component in the lower row. We can clearly see that the seasonal forecasts predict the real time series evolution much better than the others.

Here, it was observed that it gave better results with less fluctuating and seasonal time series data.



Concluding by the fact that, Prophet's default prediction interval is 80% too less. This is indicative of the fact that forecasting time series with high confidence being very difficult, or just impossible in some cases using Prophet.

III. PROPOSED MODEL

As our first step we plan to remove outliers/noise from our dataset which diverge our forecasting from the actual trend. We observed that the algorithms performed better on particular type of time series data. Like for time series with more seasonal component prophet approach would give better results. Classical methods like ETS and ARIMA give better results with short term dependencies in time series whereas complex models like RNN/LSTM gave better results when there was long term correlation in time series.

Thus we plan to do Ensemble learning [5] for web traffic prediction. Ensemble learning combines multiple predictions (forecasts) from one or multiple methods to overcome accuracy of simple prediction and to avoid possible overfit. The models that we would be working with are as follows –

1) Ensembles of classical models-

- Autoregressive (AR),
- Moving Average (MA),
- Autoregressive Moving Average (ARMA),
- Autoregressive Integrated Moving Average (ARIMA), and
- Seasonal Autoregressive Integrated Moving Average (SARIMA) models.

2) Ensembles of LSTM models
3) Ensembles of Prophet

CITATIONS

[1] https://www.kaggle.com/c/web-traffic-time-series-forecasting/data

[2] https://www.kaggle.com/headsortails/wiki-traffic-forecast-exploration-wtf-eda

[3] Petluri, N., & Al-Masri, E. (2018). Web Traffic Prediction of Wikipedia Pages. 2018 IEEE International Conference on Big Data (Big Data). doi:10.1109/bigdata.2018.8622207

[4] https://facebook.github.io/prophet/

[5] https://www.researchgate.net/publication/330484523_Machine-Learning_Models_for_Sales_Time_Series_Forecasting