Web Traffic Forecasting Using Singular Spectrum Analysis and Neural Networks

Bharadwaj Janarthanan | bharadwaj@cs.toronto.edu | 1005068720

Summary

The number of internet users have been growing worldwide, with 48% of total world population (7.4 billion) having access, as of 2017, per the report by International Telecommunications Union. With the growing usage of internet, it becomes critical to be able to accurately forecast web traffic over a considerable long period of time. An accurate forecast would allow business organisations better plan their resources to reduce traffic congestion and avert cost overheads. With big data technologies enabling efficient storage of large datasets, disaggregated data has become more available for analysis. Predictions at disaggregated levels, leveraging spatial and temporal correlations in data provide more precision in forecasts and have economic gain potential from effective resource management. In this project, we explore high dimensional multiple time series forecasting using a subset of Wikipedia web traffic dataset available on Kaggle [10]. We propose a two-stage approach to traffic forecasting. First stage uses a sequential SSA model-free technique to extract trend and seasonal components from multivariate time series data. The second stage models the residual components using a causal convolution network. The details of implementation and results are described in detail in this report.

1. Introduction

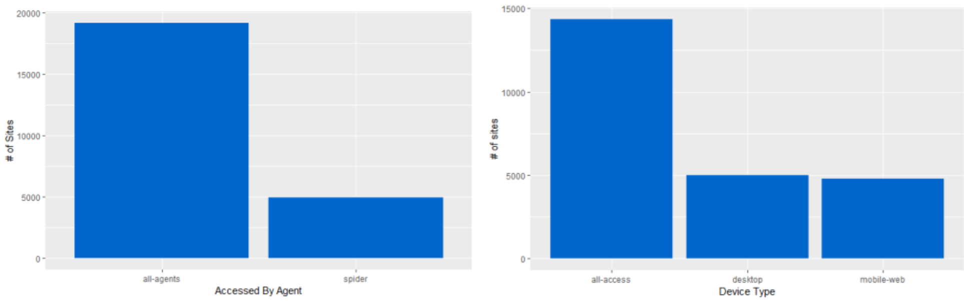
Multivariate time series forecasting has been a subject of interest in several fields such as biology, finance, weather forecasting, audio and video sequence processing. Time series data has been increasingly made available at real-time for analysis, and forecasting for multiple series can now benefit from leveraging real-world temporal and spatial correlations. However, this poses a challenge as the number of features that need to be considered could be extremely large, larger than the number of observations available. A lot of research has been performed in being able to efficiently forecast multiple time series data in high dimensional spaces using techniques such as clustering, principal component analysis [1] and neural networks.

In this project, we explore Singular Spectrum Analysis [2] and Neural networks [3] as an approach for multiple web-traffic time series forecasting. We also explore a combined approach [4] to improve forecast accuracy. The techniques chosen for exploration allow us to leverage correlations both within and across different series and have minimal computational and memory requirements. Additionally, we compare their forecast to those from a traditional time-series model such as ARIMA.

For the purpose of this project, we will use a subset of Wikipedia web-traffic pages in English and evaluate the performance of different approaches. The rest of this project report is separated into different sections. Section 2 describes datasets used for analysis, and the feature extraction process. Section 3 introduces methods used in this project for time-series forecasting. Section 4 briefly explains implementation following which, results are compared in section 5 for different techniques.

2. Datasets and Features

The dataset used for analysis has 24K web-traffic data for pages of Wikipedia English project from July 1st, 2015 to September 10th, 2017. This dataset differentiates spider traffic from the rest and desktop/ mobile traffic from the rest as shown on figure 1. However, as can be seen from figure 2,3 and 4, most traffic series, is low noise, mostly constant, while a few series are fluctuating, causing the average to be higher. Some other things to observe in data are, spider traffic is a series of lows and spikes; desktop traffic is higher in volume than mobile-web and other access types.



(a) # of sites accessed by different agents (b) # of sites accessed by different devices

Figure 1: Illustrates traffic from different web access agents and from different access devices

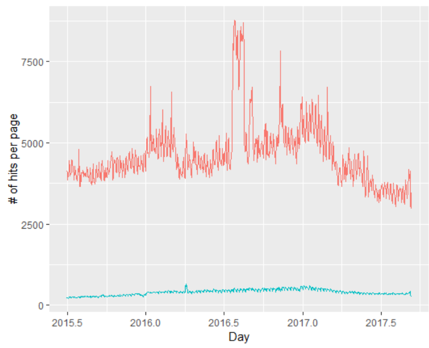


Figure 2: Most traffic is simple, and almost a constant (see median views-blue line) while a few pages experience high volume traffic shifting the average (see avg. views- red line)

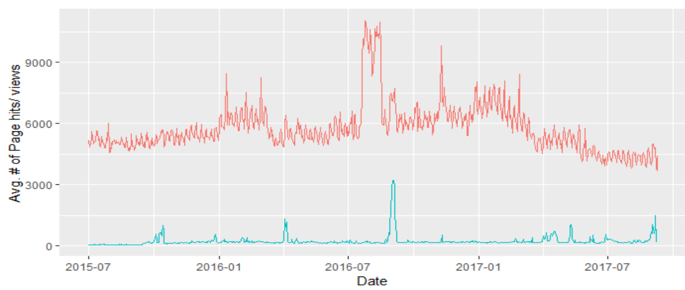


Figure 3: Spider data (blue line) has occasional spikes, it is mostly small while all-other traffic series (red line) shows some strong trend and seasonal patterns in traffic

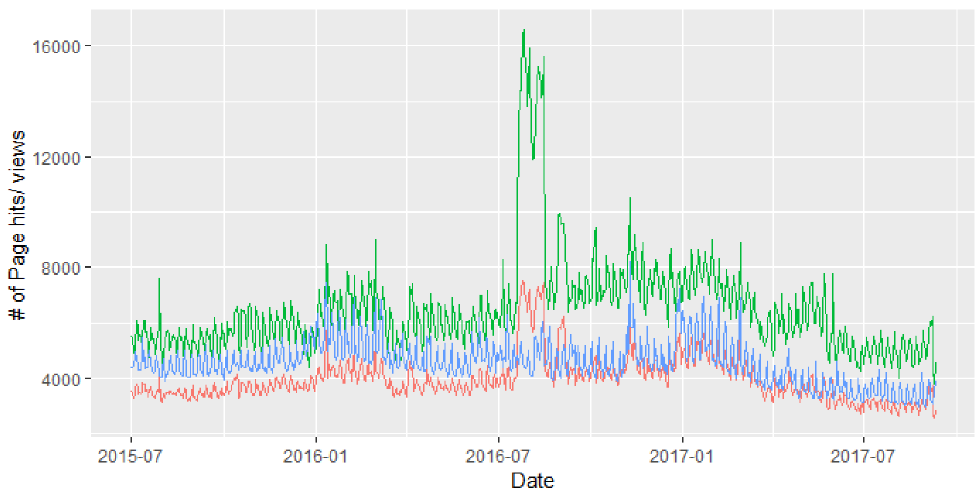


Figure 4: Desktops (Green line) have higher volume of traffic than, Mobile-Web (red line)

Also, the dataset has about 6% of traffic information missing. Since the dataset doesn't provide a way to differentiate between missing days and no traffic days in our analysis, we used a fill forward approach to treat missing data and any further NaN or 0 are treated as no-traffic days.

Based on exploratory analysis, we see that there's merit in building features specific to device types and access types in addition to accounting for trend and seasonal variations in traffic series. We would use median web-traffic series for each access type, device type as additional series into our system while performing SSA decomposition and use a one-hot encoded feature of access, device type when performing residual modelling using neural networks. These features will capture inherent differences in traffic volume from different devices and agents.

We observed that some time points in series have very high volume of traffic as against the usual traffic volume for the system. This could be an anomaly caused by an external event, rather than removing them from our data we transform the traffic volume to a logarithmic scale to treat such outliers. To be more specific, the traffic series is transformed to a log(1+x) scale after missing data treatment.

3. Methods

This section briefly introduces the methods explored in this project.

[1] Singular Spectrum Analysis for Trend and Seasonality Extraction

The key idea in singular spectrum analysis is to decompose time series into trend, oscillatory/ seasonal and noise components with no priori assumption about the parametric form. To do this, we first construct the variance-covariance matrix, C between series X and its lags and then compute eigen values and eigenvectors of C. A generic scheme for SSA methods is shown in figure 5 (taken from [5]).

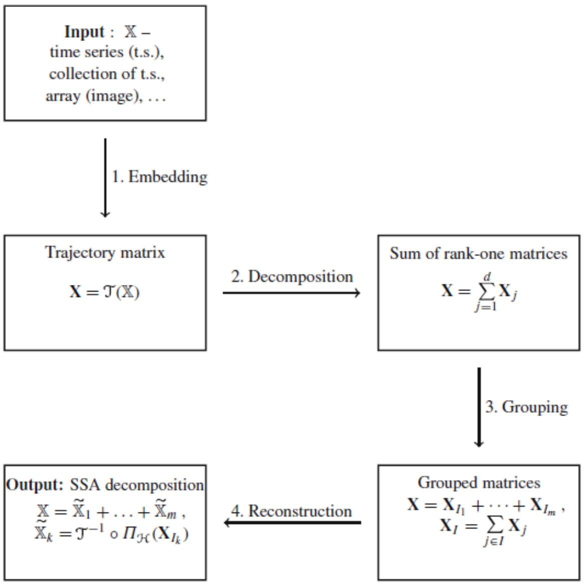


Figure 5: General SSA scheme for time series decomposition

For our data, we used a multichannel singular spectrum analysis, that takes into consideration the combined structure of the multivariate system. A brief description of the algorithm is as follows,

Decomposition:

Input:

For a multivariate web-traffic series defined as

$$\left\{\mathbf{X^p} = \left(x_k^p\right)_{k=1}^{N_p},\ p = 1 \dots s\right\} \text{of length } N_p \forall\ p = 1..s$$

Output:

Decomposition of the trajectory matrix into elementary matrices,

$$\mathbf{X} = \mathbf{X_1} + \mathbf{X_2} + \cdots \mathbf{X_d}\ ; \text{where } \mathbf{X_i} = \sqrt{\lambda_i} U_i V_i^T$$

Algorithm:

1. Construct a trajectory matrix of $\boldsymbol{X}$ in a stacked Hankel structure as,

$$\tau(X) = [\boldsymbol{X^1} \dots . \boldsymbol{X^s}], where\ \boldsymbol{X^p} = \left(\boldsymbol{X_1^p} \dots \boldsymbol{X_K^p}\right)^T = \begin{pmatrix} x_1^p & \dots & x_K^p \\ x_2^p & \dots & x_{K+1}^p \\ x_L^p & \dots & x_N^p \end{pmatrix}$$

$\mathrm{L} = \text{Window length}, \mathrm{N} = \text{Total number of observations}, \mathrm{p} = 1..\mathrm{s}$

where, $1 < \mathrm{L} < \mathrm{N}$ and $\mathrm{K} = \mathrm{N} - \mathrm{L} + 1$

Each X^p is the trajectory matrix of the individual time series p which are stacked horizontally to form $\tau(X)$

2. Compute the singular valued decomposition,

$$\boldsymbol{X} = \boldsymbol{X_1} + \cdots + \boldsymbol{X_d}; \boldsymbol{X_i} = \sqrt{\boldsymbol{\lambda_i}} \boldsymbol{U_i} \boldsymbol{V_i^T}$$

Reconstruction:

Input:

Decomposition, $X = X_1 + \cdots + X_d$; where $X_i = \sigma_i U_i V_i^T$ and

$||U_i|| = ||V_i|| = 1$; grouping $\{1 \ldots d\} = U_{j=1}^{m} I_j$

Output:

Decomposition of the time series into identifiable components, $X = X_1 + \cdots + X_m$

Algorithm:

1. Construct the grouped matrix decomposition, $X_{I_1} + \cdots . X_{I_m}; X_{I_m} = \sum_{i \epsilon I} X_i$ and $I = \{1 \ldots r\}, I \subset \{1 \ldots L\}$

2. Perform diagonal averaging/ Hankelization to transform the reconstructed matrix $\boldsymbol{X_{I_m}}$ to the form of a Hankel matrix.

$$X = X_1 + \cdots X_m; X_m = \tau(X)^{-1} * \pi(X)$$

where $\tau(x)$ is the block − trajectory matrix of X and

$\pi(x)$ is the orthogonal projector defined as,

$\pi(\mathbf{X}) = [\pi(\mathbf{X}^1) \ldots . \pi(\mathbf{X}^p)]$; $\pi(X^p)_{i,j} = \sum_{(l,k)\epsilon A_s} x_{lk}/w_s$

$$s = i + j - 1, A_s = \{(l, k): l + k = s + 1, 1 \le l \le L, 1 \le k \le K\}, w_s = |A_s|; \text{the number of elements in } A_s$$

Forecasting:

For our analysis we used a recurrent, column-based forecasting technique, the other types of forecasting are discussed in [5]

Input:

Collection of time series X_p of length N_p, where p = 1, . . ., s,

Window length L,

Orthonormal system of vectors $\{P_i\}_r$, i=1,

Forecast horizon M

Output:

Forecast values, $\left(\hat{x}^p_{N_{p+1}} \ldots \hat{x}^p_{N_{p+M}}\right), p = 1 \ldots s$

Algorithm:

1. Construct Z matrix as the last L-1 values of the reconstructed components.

$$Z = \begin{pmatrix} \tilde{x}_{N_1-L+2}{}^1 & \cdots & \tilde{x}_{N_1}{}^1 \\ \vdots & \vdots & \vdots \\ \tilde{x}_{N_s-L+2}{}^s & \cdots & \tilde{x}_{N_s}{}^s \end{pmatrix}$$

2. If, $v^2 = \sum_{j=1}^{r} \pi(U_j)^2 < 1$ then the column MSSA forecast is defined as,

$$R_n = ZR_L; R_L = \frac{1}{1 - v^2} \sum_{j=1}^{r} \pi(U_j) U_j \; \epsilon \; R^{L-1}$$

A thorough and detailed review of the singular spectrum analysis techniques can be found on textbook [5] from which the above algorithms have been adapted.

[2] Modelling using dilated 1D convolutions

Convolution networks are commonly used in image and speech classification problems to capture spatially invariant patterns. The idea can be extended to solve multivariate time series forecasting problems to capture spatial and temporal correlations in data. CNNs are good at capturing local patterns and extracting similar features across data using kernels/ filters which slide across the data in fixed widths to generate abstract features from the series. For example, for a CNN with kernel of width 7, we slide the kernel across 7-day subspaces in data and look for weekly periodicity in series. A wavenet [6] based approach is used to capture both autocorrelation and cross-correlation in multivariate series. The core structure of a wavenet is causal padding and dilation. The time series data is padded such that the prediction at time step, t doesn't depend on any of the future time steps, (t+1), (t+2) and so on. This can be achieved using the causal padding structure, which shifts the input to a convolution by a few time steps as shown on figure 6.

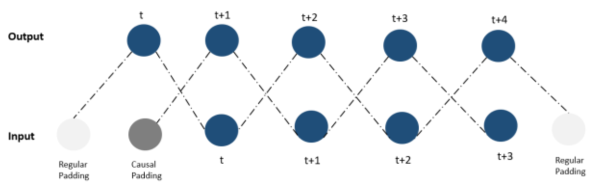


Figure 6: Example of causal padding of input series of 4-time steps to predict 5th

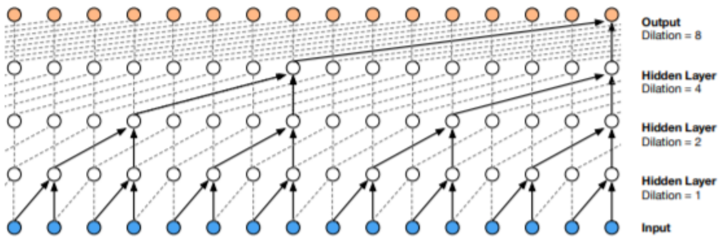


Figure 7: An illustration of a stacked dilated convolution from the Wave Nets paper

Additionally, to increase the receptive field of the convolution network, we use dilation that increases exponentially at each layer, thereby allowing to look at an input receptive field larger than the kernel width of the network. A dilated convolution skips certain parts of the input, allowing capture of long-term dependencies in series. Stacking dilated convolution layers one on top of another allows to have a large receptive field as illustrated in figure 7. Do note, that the time required for model training and convergence is much lesser than the time needed for a recurrent neural network to converge, a network that is more commonly used in time series forecasting problems. Wave-Net structure has a form similar to an AR(p) model,

$$\hat{x}(t+1) = \sum_{i=1}^{p} \alpha_i x_{t-i} + \epsilon(t); where\ \epsilon(t)\ is\ WN\ \ and\ p\ is\ kernel\ width$$

The forecasts at time t+1, x_{t+1} is conditioned on traffic at, $x_{1,2\ldots t}$ and the traffic on other pages $y^{p}_{1,2\ldots t}\ \forall\ p = 1..s; where\ s = \#\ of\ webpages$ using a soft-ReLU (rectified linear unit) activation function. Reference [3] discuss in detail this approach for conditional time series forecasting of future.

We explored the two techniques as stand-alone approaches for multivariate forecasting and evaluated performance of a combined approach as well. Wikipedia English project web-traffic data described in section 3 was used for this study. Section 4 describes in detail our implementation and 5 compares the results across different models against a traditional time series model such as ARIMA.

4. Implementation

This section explains how the methods discussed in section 3, were used in forecasting web-traffic of 24k webpages of Wikipedia.

The forecasting approach involves three layers,

1. **First Layer:** Trend Extraction
2. **Second Layer:** Seasonality Extraction
3. **Third Layer:** Local Correlations in Series

First Layer: Trend Extraction using SSA

The log(1+x) transformed data is used for modelling. This transformation is done to ensure that no one page with large views would dominate while performing singular spectrum analysis (SSA). The data is arranged in format of [Date X Web-Page].

Since, primary intent in this step is to extract Trend, a small window-length say 30, was chosen to construct the trajectory matrix for each webpage views. The constructed matrices were then stacked horizontally and a singular valued decomposition of the same was performed to extract eigen components. Do note that, a window length of 30 was chosen based on experiments of different window lengths on a subset of 4000 webpages and evaluating their performances on a holdout period of the last 64 days.

To identify which component/ eigen vector represents trend, we plot them and look at increasing or decreasing components in them, which would constitute the trend. As can be seen on figure 8, there's a sharp elbow after the first component, suggesting that it explains the most variance in the series and from figure 9, the first component suggest a simple constant, which is what most webpage traffic are through time. Also, figure 10, the W-correlation matrix suggests that components 2 to 30 are mixed or have eigen components of similar weights. Separability of components is one important criterion while reconstructing the series.

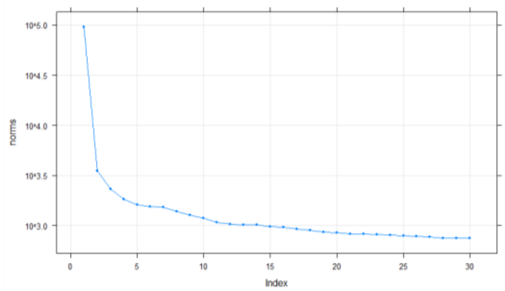


Figure 8: Scree Plot for MSSA with window length, L=30

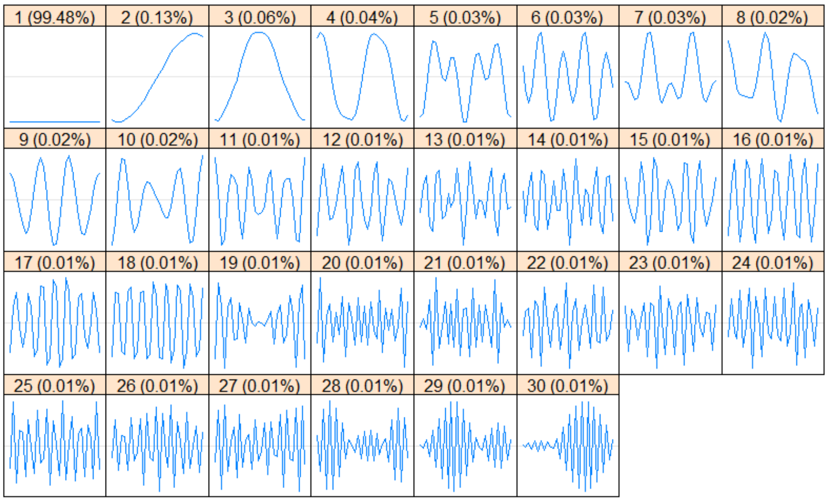


Figure 9: Eigen Vector Plot of MSSA with window length, L=30

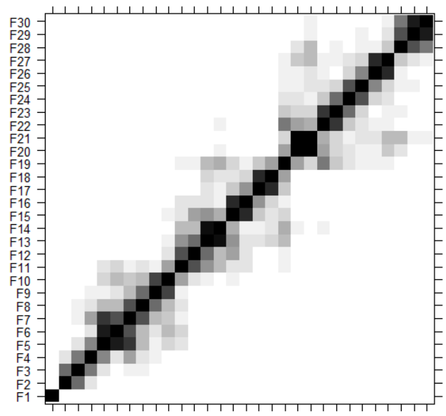


Figure 10: W- Correlation Matrix of Eigen Components for MSSA with L=30

This can be handled using a sequential SSA technique, the trend is first extracted using a MSSA of window length 30 and the residuals are modelled using a MSSA of larger window length to capture seasonal/ cyclic components of the series. This leads us to the second layer of modelling.

Second Layer: Seasonality Extraction using SSA

The residuals from trend SSA are then decomposed using SVD, into eigen components each representing a certain component of the series. In order to reconstruct the series, we find the top eigenvectors which best approximate the overall series. From the scree plot, figure 11 it is evident that components 1,2; 3,4;5,6;7,8 and so on have similar slopes and represent approximately the same component of the series. Observing the paired plots of the eigen vectors, figure 12 it is evident that eigen vectors 1 and 2 represent weekly seasonal component of the series forming a regular shape with about 7 vertices. We can validate the same by estimating the parameters of the two wave components using the ESPRIT technique [7] and it suggest that the waves have a periodicity of 7, equivalent to saying weekly seasonal behaviour, fragment 1. However, no other paired plots suggest a sine-cosine pair and suggest that the other components don't explain a seasonal behaviour. Do note that, though the authors of SSA [2] suggest a window length of N/2, for our data a lower window length of 60 works well.

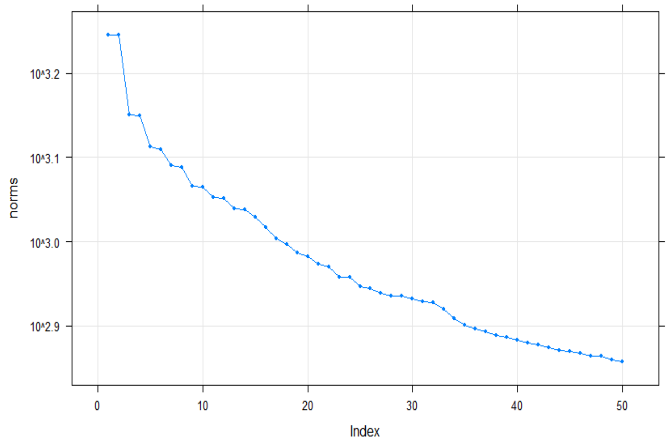


Figure 11: Scree plot for Seasonality MSSA, L=60

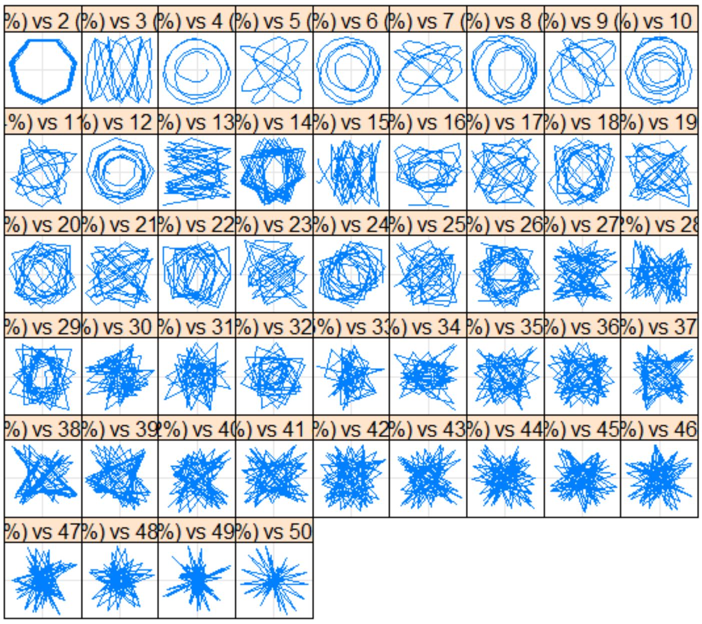


Figure 12: Paired Eigen Vector Plot for Seasonality MSSA, L=60

```
> parestimate(ssa_remaining, groups = list(1:2), method = "esprit")
   period       rate   |      Mod       Arg  |       Re        Im
    7.036  -0.000752   |  0.99925      0.89  |  0.62664   0.77835
   -7.036  -0.000752   |  0.99925     -0.89  |  0.62664  -0.77835
```

Fragment 1: Parameter Estimation: first two eigen values of Seasonal MSSA, L=60

Third Layer: Local Correlation in Series using CNN

The remainder of the series components was then fed into a stacked 1D convolutional network with dilation. The web-traffic data was reshaped into a tensor of dimension, Web page X Date X Features. The architecture of CNN is as follows,

- Input series: causal padded to ensure that forecasting at, t+1 doesn't depend on data in t+2…. N series values. This is an asymmetric padding structure unlike regular CNN.
- 9 layers of 1D CNN stacked with dilation growing exponentially:1, 2,4, 8…256
- Batch normalization layers to ensure data is batch normalized

- A kernel of size 7 to capture weekly seasonality in web traffic
- A soft ReLU activation function on the final layer and a dropout layer with a dropout rate of 0.2 is used on the final layer
- Data is split into 70% for training and 30% for validation during model training
- During prediction, the forecast at t is fed as recurrent input to predict traffic at t+1
- The model was trained using a batch size of 256 for a maximum of 2000 epochs with an early stopping rule, wherein we would stop training if the validation loss doesn't drop for 10 consecutive epochs.

Forecasting of series x_{t+1} is conditioned on traffic through $x_{1,2\ldots t}$ and the traffic on other pages y_{t+1}^{p} $\forall$ p=1...s, where s is the number of webpages, through an activation function. Prediction of the three layers are then combined and transformed back to its original scale. We also, explored using Seq. SSA and CNN as independent approaches to forecast web-traffic, in which case the web-traffic data was directly used for modelling. Figure 13 provides a view of the combined prediction architecture.

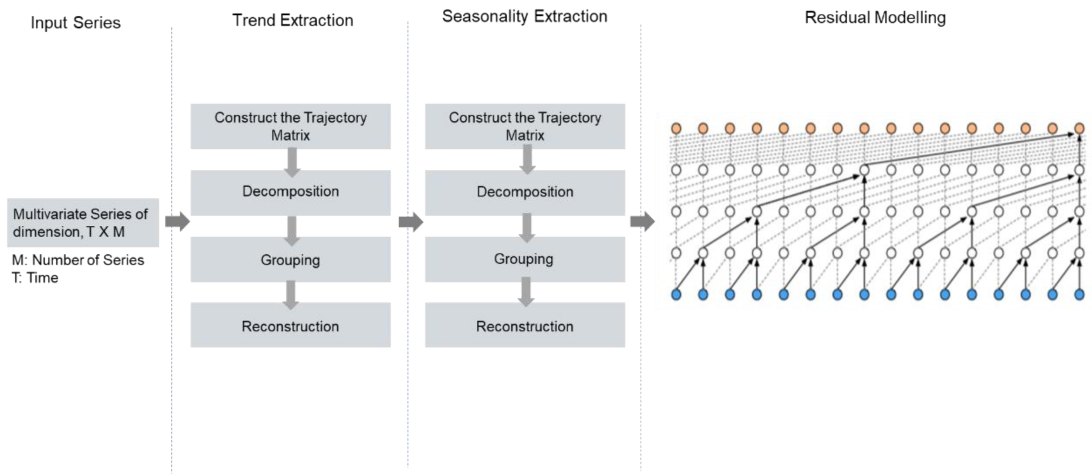


Figure 13: Proposed Modelling Approach using Seq. SSA and CNN

5. Results and Evaluation

In this section, we compare and evaluate performance of different modelling approaches. We use sMAPE as the metric to measure accuracy of prediction. sMAPE is robust to outliers in data, however it is not symmetric in percentage scale unlike MAPE. sMAPE penalizes under-prediction more than over-prediction. We don't want to under-predict traffic volume and cause congestion on network, so our focus in measuring model performance would be to measure how well we predict traffic volume and how often and to what extent we under-predict, therefore sMAPE would be an appropriate choice of metric for the same.

The definition of sMAPE is as follows,

sMAPE= 2*(|Actual-Forecast|)/(|Actual|+|Forecast|)

Do note that sMAPE ranges from (0,2) and sMAPE% can range from 0 % to 200%.

We train on one year of data and forecast the next 64 days. Since, we have two years of data from 2015-07-17 to 2017-07-19 we do a 6-fold sliding window cross validation to evaluate how our model performs at different periods of time. We compared results for different forecast horizons and saw how error accumulates over time period for different validation folds. The results from the same are shown on table 1, 2 and 3. Figure 14, compares results from the three approaches for May 2017 to July 2017 forecast and it is evident that Seq. SSA does much better in predictions than a CNN approach and there's a marginal gain in accuracy by combining the two approaches. We then compared the results from a univariate ARIMA model with the results from our approach for the most recent validation period and found that, our approach performs better than ARIMA models, see figure 15. Figure 16 shows forecast for the top 5 web-traffic series and the forecast series closely model the accuracy except for when the series experience anomalous shift in volume during the validation period.

Validation Start Date	**Validation End Date**	**Seq. SSA**									
		1	**8**	**15**	**22**	**29**	**36**	**43**	**50**	**57**	**64**
01-07-2016	02-09-2016	30.61	34.38	36.34	37.96	39.48	40.74	42.23	43.55	44.28	44.74
03-09-2016	05-11-2016	29.46	33.24	35.62	37.46	38.96	40.19	41.28	42.11	42.90	43.58
06-11-2016	08-01-2017	31.66	36.88	37.66	38.23	38.64	39.29	39.90	40.75	41.77	42.32
09-01-2017	13-03-2017	33.30	34.29	36.17	37.65	38.88	39.78	40.55	41.11	41.72	42.13
14-03-2017	16-05-2017	25.81	28.75	31.32	32.37	34.17	35.29	35.82	36.11	36.52	36.96
17-05-2017	19-07-2017	25.58	29.06	30.77	31.65	32.70	33.90	34.78	35.60	36.15	36.44

Table 1: sMAPE for different validation periods, and forecast horizons using seq. SSA

Validation Start Date	**Validation End Date**	**CNN**									
		1	**8**	**15**	**22**	**29**	**36**	**43**	**50**	**57**	**64**
01-07-2016	02-09-2016	65.18	65.78	65.72	65.59	65.72	65.69	66.04	66.33	66.32	66.21
03-09-2016	05-11-2016	63.74	62.49	62.40	62.23	62.32	62.37	62.62	62.73	62.78	62.87
06-11-2016	08-01-2017	61.02	63.43	62.85	62.27	62.02	61.94	61.82	61.92	62.13	62.07
09-01-2017	13-03-2017	61.60	61.37	61.70	61.99	62.27	62.37	62.41	62.39	62.48	62.42
14-03-2017	16-05-2017	56.73	57.52	57.82	58.00	59.25	60.04	60.09	59.82	59.71	59.77
17-05-2017	19-07-2017	52.90	53.28	52.90	53.16	53.62	53.55	53.58	53.73	53.71	53.65

Table 2: sMAPE for different validation periods, and forecast horizons using CNN

Validation Start Date	Validation End Date	Seq. SSA + CNN									
		1	8	15	22	29	36	43	50	57	64
01-07-2016	02-09-2016	30.58	34.12	35.97	37.57	39.16	40.45	41.93	43.25	44.03	44.53
03-09-2016	05-11-2016	29.19	32.95	35.35	37.23	38.74	39.97	41.10	41.95	42.73	43.41
06-11-2016	08-01-2017	32.00	36.84	37.67	38.22	38.64	39.31	39.94	40.82	41.85	42.41
09-01-2017	13-03-2017	33.21	34.05	35.89	37.37	38.60	39.54	40.36	40.99	41.64	42.08
14-03-2017	16-05-2017	25.70	28.62	31.13	32.10	33.88	35.02	35.57	35.86	36.27	36.73
17-05-2017	19-07-2017	25.36	28.84	30.48	31.33	32.38	33.55	34.42	35.22	35.78	36.12

Table 3: sMAPE for different validation periods, and forecast horizons using seq. SSA + CNN

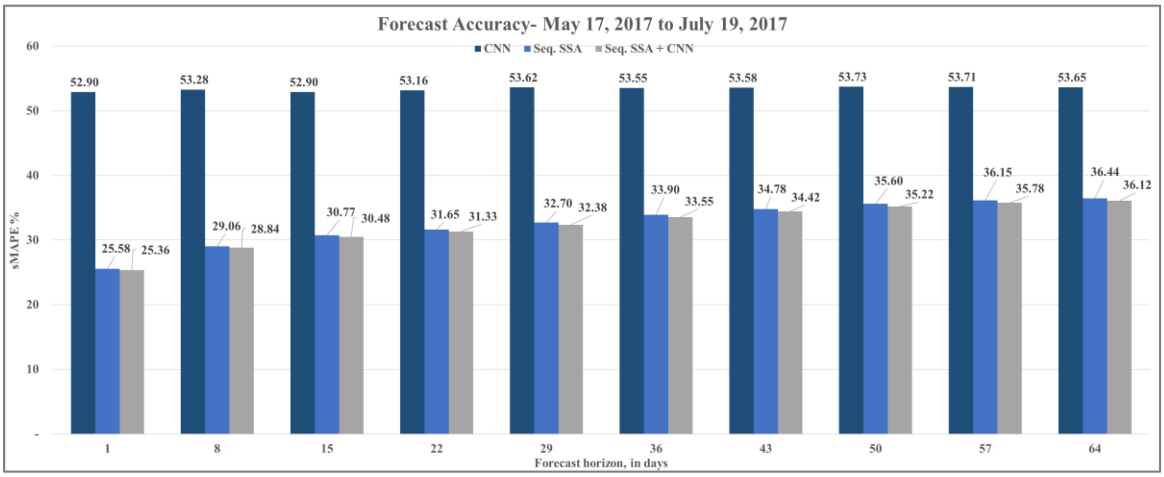


Figure 14: Comparing the three approaches for a recent forecast period, May 2017 to July 2017

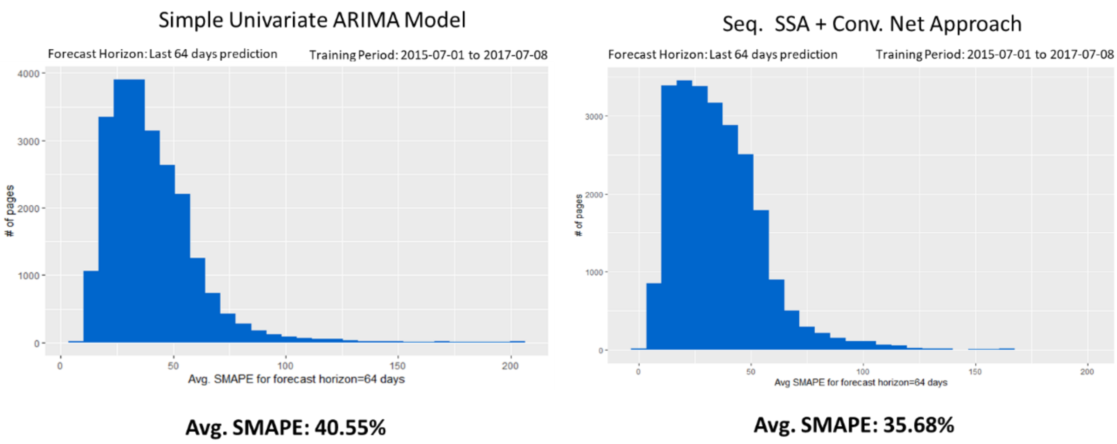


Figure 15: Comparing Seq. SSA + CNN approach with traditional ARIMA approach for validation period, July 9, 2017 to October 10, 2017

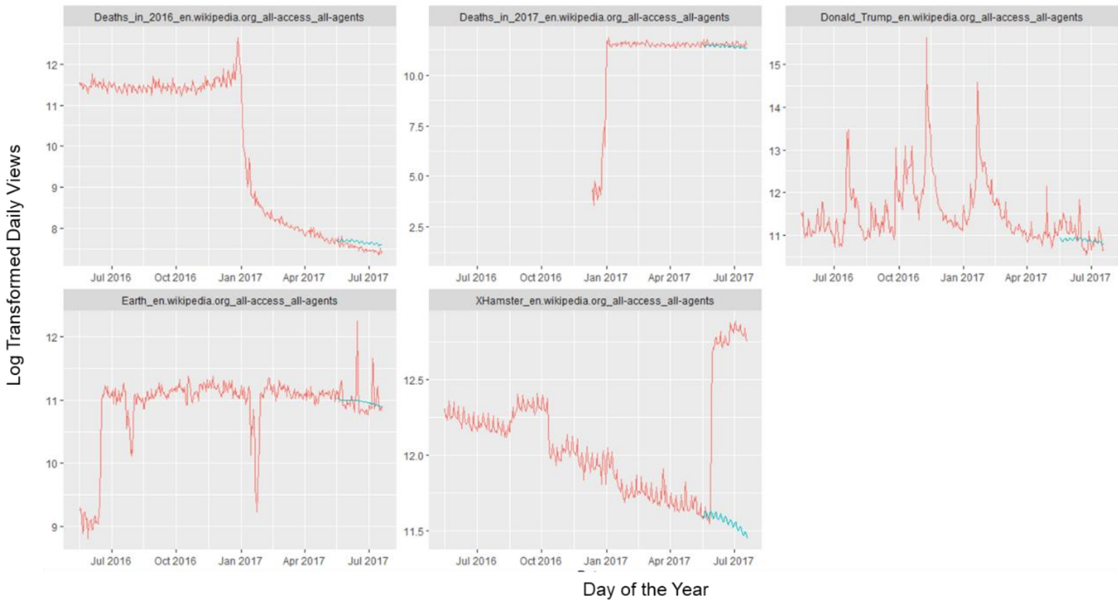


Figure 16: Actual Vs Forecast for July 9, 2017 to October 10, 2017 for top 5 series in traffic volume

6. Conclusion and Future Work

In this project we explored Singular Spectrum Analysis as a technique for forecasting in multivariate system, and Convolutional Networks as a method to perform conditional time series forecasting. We observed from our results that a combined approach of both is better than an ARIMA model for each series. The SSA and CNN techniques capture spatial and temporal correlations across the multivariate system which a univariate ARIMA wouldn't. Also, the number of parameters for model identification is much lesser in an SSA+CNN approach than in ARIMA which would require to at least 3 parameters for every series. Also, ARIMA models developed need to be statistically validated using Box-Jenkins approach as against SSA approach which allows us to use diagnostic eigen and scree plots to identify trend and seasonal components of the series.

However, SSA and CNN have a few parameters that need to be tuned to data to achieve best performance. For SSA, the choice of window length, L needs to be done through cross validation and the grouping of eigen-vectors is done by inspection of their frequency and eigen pair plots. It is also important to ensure the different series are of similar scale to ensure no one series dominates during decomposition. For CNN, the weights of units and the number of layers and dropout can be tuned to yield better performance. The batch size could also be further experimented to yield optimal performance. In our case, we tried 128, 256, 512,1024 and observed that performance didn't really improve much after 256. The data fed into the layers need to be normalized to ensure stationarity in series.

Also, we observed that our approach doesn't model extreme events quite well such as sudden spikes in series and when the series has a sudden shift in volume during validation period which was not observed during training. In such scenarios, we could explore using approach discussed on [9] using LSTMs for anomaly detection and extreme event forecasting.

7. References

[1] Dezhi Hong, Quanquan Gu, Kamin Whitehouse. High-dimensional Time Series Clustering via Cross-Predictability.
http://www.cs.virginia.edu/~dh5gm/pdf/aistats17-paper.pdf

[2] Nina Golyandina, Anton Korobeynikov, Alex Shlemov, Konstantin Usevich. Multivariate and 2D Extensions of Singular Spectrum Analysis with the Rssa Package.
https://arxiv.org/pdf/1309.5050.pdf

[3] Anastasia Borovykh, Sander Bohte, Cornelis W. Oosterlee. Conditional time series forecasting with convolutional neural networks.
https://arxiv.org/pdf/1703.04691.pdf

[4] Sokchhay Heng, Tadashi Suetsugi. Coupling Singular Spectrum Analysis with Artificial Neural Network to Improve Accuracy of Sediment Load Prediction
https://file.scirp.org/pdf/JWARP_2013042214553428.pdf

[5] Nina Golyandina, Anton Korobeynikov, Anatoly Zhigljavsky. A textbook on Singular Spectrum analysis with R.

[6] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, Koray Kavukcuoglu. WaveNet: A Generative Model for Audio.
https://arxiv.org/pdf/1609.03499.pdf

[7] Richard Roy and Thomas Kailath. ESPRIT- Estimation of Signal Parameters Via Rotational Invariance Techniques.
https://pdfs.semanticscholar.org/831e/d2a5f40861866b4ebfe60257b997701e38e2.pdf?_ga=2.257537419.1833705404.1544169506-218396998.1539304439

[8] Rob J Hyndman, Errors on percentage errors
https://robjhyndman.com/hyndsight/smape/

[9] Nikolay Laptev, Jason Yosinski, Li Erran Li, Slawek Smyl. Time-series Extreme Event Forecasting with Neural Networks at Uber
http://roseyu.com/time-series-workshop/submissions/TSW2017_paper_3.pdf

[10] Google, Web Traffic Forecasting-Forecast future traffic to Wikipedia pages.
https://www.kaggle.com/c/web-traffic-time-series-forecasting