# Web-traffic Time Series Forecasting

Ahmad N. , Saud A. , and Bilal M.

***Abstract*— Time Series Analysis deals with finding patterns in a series of equally spaced data points indexed in order of time. With an increase in web connectivity to all users and amount of information available, information has now become a commodity. It is therefore essential for websites to figure out what to host when, to earn maximum revenue. Our paper aims at forecasting popularity of languages among the articles hosted by a web server so that it knows on which language it should dedicate most of its resources in terms of hosting to generate maximum user traffic. Given a training set of only 450 days we are forecasting the number of user hits for the future 50 days. Our work presents a comparative analysis of the two state of the art algorithms used in the domain of Time Series (i) ARIMA (Auto-Regressive Integrated Moving Average) (ii) LSTM (Long Short Term Memory) by forecasting the number of user views/hits corresponding to each language and then generating rankings of all the languages present.**

***Index Terms*— Time Series; Autoregressive (AR); Moving Average (MA); ARIMA; LSTM; Stationarity; ACF; PACF; AIC.**

## I. Introduction

The popularity of Internet has led to availability of huge volumes of user traffic. The idea of traffic being equivalent to a physical commodity has led to a contention among different web hosts to pull the maximum traffic available. After a web host acquires traffic - a certain number of visitors per day, it becomes an interesting space for advertisers. As it understands that the website is a frequently visited resource, which means that by placing advertisements here, it can draw attention to the product, but advertisers can appear here only with the permission of the owner. This opportunity is granted to the advertisers for a certain monetary reward. The web server in our case hosts all sorts of different articles. It is therefore essential for the server to know that hosting articles of which particular language would result in maximum user traffic and hence maximum possible revenue. Given a log file of number of times an article is visited in a day for all articles hosted for a total of 450 days we have to deduce which language would be the most popular in the next 50 days. As the data given is a set of observations of a quantitative variable at equally spaced (per day) time points therefore it is easy to deduce that this is time series data and requires time series modelling. A time series is a sequential set of data points, measured typically over successive times [1,8,9,11].The procedure of fitting a time series to a proper model is termed as Time Series Modelling [16].The main aim of time series modelling is to carefully collect and rigorously study the past observations of a time series to develop an appropriate model which describes the inherent structure of the series. This model is then used to generate future values for the series, i.e. to make forecasts. Time series forecasting thus can be termed as the act of predicting the future by understanding the past [11]. Time Series Modelling can be done through a variety of methods on the basis of underlying assumptions used. One of the most popular technique used is ARIMA (Auto Regressive Integrated Moving Average). ARIMA assumes that the time series is linear and follows a particular statistical distribution. The popularity of ARIMA comes from the fact that it is simple as well as flexible in representing different varieties of time series for optimal model building process. Recently, Neural Networks (NN) have gained a lot of attention in time series modelling [4,6,7]. The best part about Neural Networks is that they have an in built capability of non-linear modelling without any presumption about the statistical distribution followed by the observations [16].

## II. Literature Review

### A. ARIMA(Auto Regressive Moving Average Model)

An ARIMA model is a generalization of the ARMA (Auto Regressive Moving Average) model. An ARMA model is a linear superposition of AR (Auto Regressive) and MA (Moving Average)model.AR modeling techniques enable to predict the current values of a time series, $x_t$ based on the past values $x_{t-1}$,$x_{t-2}$,...$x_{t-p}$ plus a prediction error $e_t$ where $n$ is the model order. Equation 1 describes the autoregressive model with order $p$ and $a_p$ being the AR models coefficients [5,9].

$$x_t = a_1 x_{t-1} + a_2 x_{t-2} + ... + a_p x_{t-p} + e_t \tag{1}$$

An MA model on the other hand is a linear regression of the current value of the series against current and previous (observed) white noise error terms. Hence, an MA(q) model predicts the future value of a variable by computing the linear combination of past and the current error terms,$e_t$ [5,8,9]. Equation 2 describes the moving average model with order $q$ and $b_q$ being the MA models coefficients.

$$x_t = b_1 e_{t-1} + b_2 e_{t-2} + ... + b_q e_{t-q} + e_t \tag{2}$$

Autoregressive (AR) and moving average (MA) models can be effectively combined together to form a general and useful class of time series models, known as the ARMA models [5,8,9].

$$x_t = a_1 x_{t-1} + ... + b_1 e_{t-1} + ...a_p x_{t-p} + b_n e_{t-n} + e_t \tag{3}$$

The limitation of ARMA is that it can only be used for time series that are stationary in nature. A time series is

said to be stationary (weakly) if its mean is constant- not a function of time while the covariance is only a function of time difference (lag terms). Some mathematical(hypothesis testing) tests like the one given by Dickey and Fuller [8] are generally used to detect stationarity in a time series data. However in practice as most time series show non-stationary behavior including those time series which contain trend and seasonality patterns. ARMA model in these situations are inadequate to properly describe non-stationary behavior of the time series. ARIMA model being a generalization of the ARMA model relaxes the assumption of stationarity by including a differencing parameter in the model. In ARIMA, a non-stationary time series is made stationary by applying finite differencing of data points [16] The series is differenced with its increasing lagged versions until stationarity is achieved. The degree of differencing is hence the maximum order of lag required to make the series stationary after taking the difference between the actual series and the lagged version. An ARIMA is therefore parameterized by p,q,d where p refers to the order of auto-regressive part, q refers to the order of moving average part while d refers to the order of differencing.

The value of p and q are found by ACF (Auto-Correlation Function) and PACF (Partial Auto-Correlation Function). ACF is defined as the correlation of the time series observations with observations of previous time steps, called lags. PACF on the other hand is the correlation of an observation in a time series with observations at prior time steps with the relationships of intervening observations removed. By observing the autocorrelation function (ACF) and partial autocorrelation (PACF) plots of the differenced series, you can tentatively identify that whether there is an existence of AR and/or MA term or not. If the PACF displays a sharp cutoff while the ACF decays more slowly (i.e., has significant spikes at higher lags), then the model most suited for the stationarized series would have AR order only i.e q is equal to zero but p is not equal to zero.

Conversely, if the ACF displays a sharp cutoff while the PACF decays more slowly (i.e., has significant spikes at higher lags), then the model most suited for the stationarized series would have MA order only i.e p is equal to zero but q is not equal to zero.

For many cases, both ACF and PACF would show similar trends of decaying with increasing lags and this would suggest that a mixed model i.e ARMA model is suited for the time series  both p and q are not equal to zero .Finding the model parameters, in this case, becomes difficult by just looking at the plots of PACF and ACF, so some different techniques such as grid search to find minimum AIC( Akaike information criterion) would help in choosing the right order of ARMA model [12].

### B. LSTM

Neural Networks have been recently used a lot in time series modelling as their inherent features make them quite favorable for such analysis. To begin with Neural Networks are data-driven and self-adaptive in nature [3, 7]. There is no need to specify a particular model form or to make any a priori assumption about the statistical distribution of the data; the desired model is adaptively formed based on the features presented from the data. In addition, they are inherently non-linear, which makes them more practical and accurate in modelling complex data patterns, as opposed to various traditional linear approaches, such as ARIMA methods [3, 4, 7].

However, there are special versions of Neural Networks designed especially to deal with time dependent or sequential data known as RNNs (Recurrent Neural Networks). A RNN can be thought of a collection of regular modules of Neural Networks with the hidden layers connected to each other. As a result of this, the input size to this structure is variant and the output depends upon the previous outputs hence introducing a notion of memory in the system. Even though, RNNs are suited perfectly to capture the time dependencies present in the data, a vanilla RNN does not perform well practically. This problem arises due to phenomenons known as vanishing and exploding gradients. As the gradient in any layer is a product of gradients of the previous layers (back-propagation algorithm) therefore deep in the network if the preceding gradients are less than one then the current gradient would approach to zero resulting in no update in the weights of the model- Vanishing Gradient. However, if the preceding gradients are greater than one then the current gradient would blow up result in very large updates to the weights of the model during training- Exploding Gradients.

Gated Versions of RNNs such as LSTMs are a solution to these problems with gradients. A LSTM has three gates, forget gate, update gate and the output gate.

The forget gate decides which information from the past should be kept and what part should be discarded by passing the information from the previous hidden state and current input to the sigmoid function.

$$f_t = \alpha(x_t U^f + h_{t-1} W^f) \tag{4}$$

The input gate decides which values will be updated by passing the hidden state and current input into a sigmoid function.

$$i_t = \alpha(x_t U^i + h_{t-1} W^i) \tag{5}$$

The candidate cell state is found by passing the previous hidden state and the current input into a tanh function. Moving forward, current cell state is found by multiplying(element wise) the forget gate with the previous cell state and adding it to the product (element wise) of the input gate and the candidate cell state. In doing so the forget gate suppresses some of the past values while the input gate modifies the current values.

$$\bar{C}_t = tanh(x_t U^g + h_{t-1} W^g) \tag{6}$$

$$C_t = \alpha(f_t * C_{t-1} + i_t * \bar{C}_t) \tag{7}$$

The output gate simply decides what the next hidden state would be by multiplying with the current cell state.

$$o_t = \alpha(x_t U^o + h_{t-1} W^o) \tag{8}$$

$$h_t = tanh(C_t) * o_t \tag{9}$$

As the gates ensures that only selected information from the past passes through, they result in solving the vanishing and exploding gradient problem [13]. Fig. 1 shows LSTM block containing showing all the gates (input,forget,ouput) and their workflow.

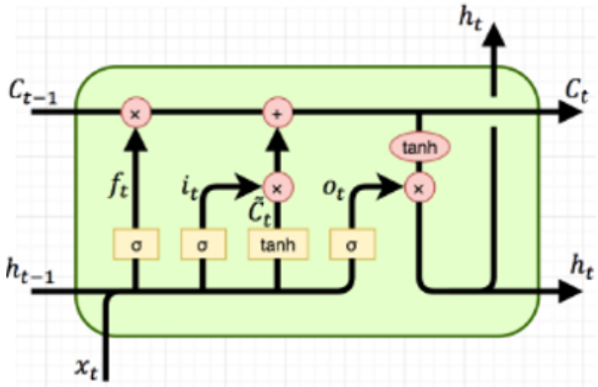


Fig. 1. LSTM block containing showing all the gates (input,forget,ouput) and their workflow

## III. Approach

We are given a data set of 145k different articles along with the number of hits they received over a period of 500 days. The first step involves doing preprocessing.

### A. Preprocessing

In the preprocessing phase, invalid values , NULL values and outliers are removed. Firstly, we remove NULL values from the data. Moving forward we remove the outliers by computing the z-values and using thresholding. Following, this our task is to categorize the articles based on the language in which they are written. This is done by the use of regular expressions. However, there are some articles which either belong to more than two languages or their language cannot be identified. We categorize these articles as invalid and simply omitted them from our data set. Now that we have data in the form of languages and corresponding number of hits/views, we plot correlations of a particular language against all languages for the entire language set.

For each language, its correlation with other languages is very low (below 0.50) therefore we can easily conclude that while predicting hits for one language, the other languages can be ignored. Given that now we have data across all the languages. We pick one language at random to do the analysis. In this paper, we have picked Japanese as an example. The analysis however remains same for all the languages present (Fig. 2 ).

### B. Applying ARIMA

The main task at hand is to find the values of auto-regressive part (p), moving-average part (q) and differencing term (d).

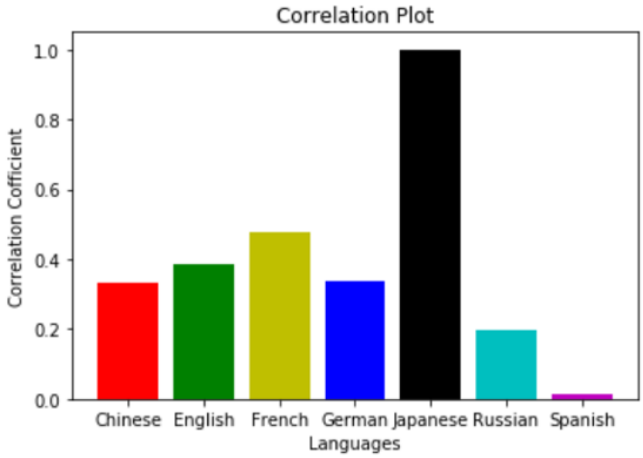


Fig. 2. Correlation plot of Japanese with all the other languages. There is low correlation of Japanese with all the other languages therefore when doing analysis for Japanese, the contribution of all the other languages can be ignored .

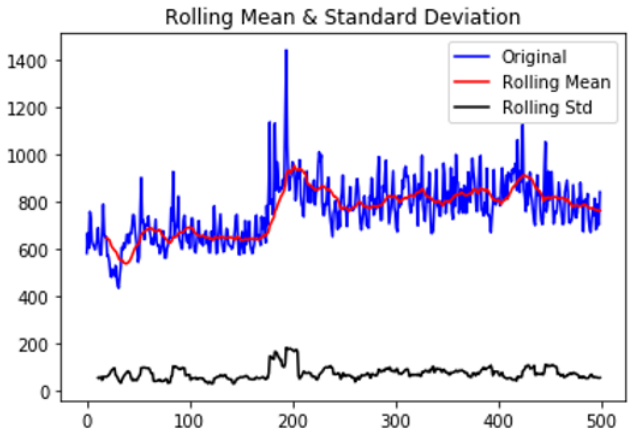


Fig. 3. Plot of statistical measures before differencing. The time series is not stationary as the mean is not equal to zero

To begin with, we apply the Dicky-Fuller test to check the stationarity of the time series. This results in the rejection of the null hypothesis advocating the non stationarity of the time series (Fig. 3). Taking the first difference however makes the series stationary as it results in acceptance of null hypothesis and a mean of zero (Fig. 4). Once, we decide the value of (d) equal to one, we plot the plots for ACF (Auto Correlation Function) and PACF (Partial Auto Correlation Function)(Fig. 5)(Fig. 6). As both the ACF and PACF show downward trend with increasing lags therefore this confirms that the model to be fitted should have both AR and MA components. In order to find the exact numerical values of (p) and (q), we carry out a grid search on different values of p and q both starting from zero to ten. On each combination of p and q with d equal to one we find the corresponding AIC values. The combination of p and q equal to four results in the minimum value of AIC. This combination is our optimal choice for p and q.

To find the best values of p,q,d we do the following:

1) Check Stationarity- By Dickey Fuller Test.

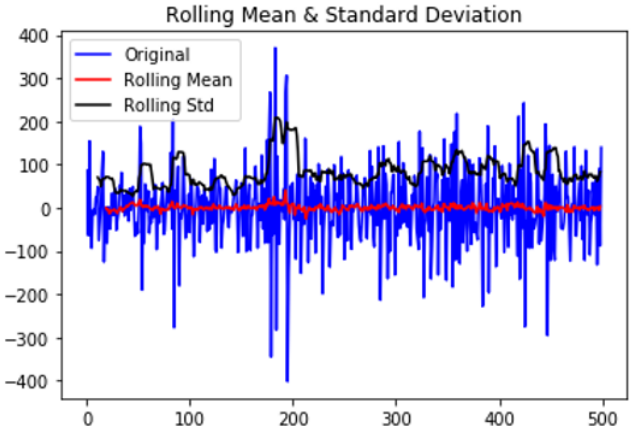


Fig. 4. Plot of statistical measures after differencing. The time series becomes stationary as the mean becomes equal to zero

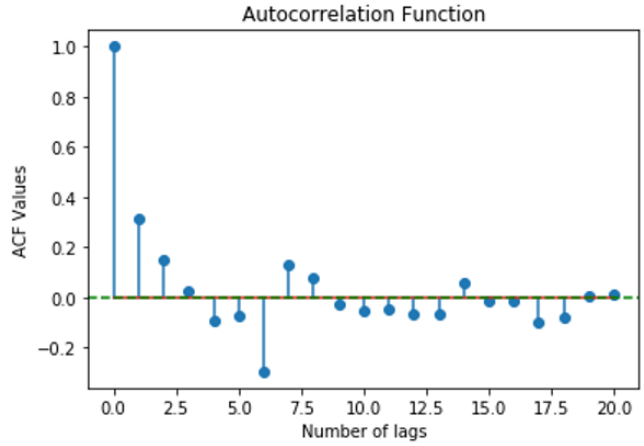


Fig. 5. : Plot of Auto Correlation Function. There is a steep downward trend indicating the presence of a MA(Moving Average) component

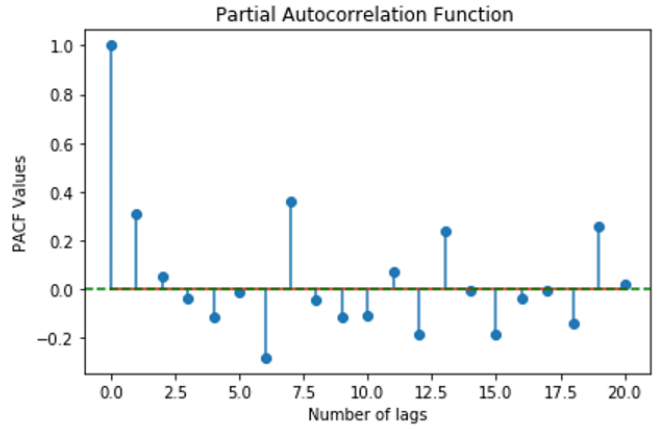


Fig. 6. Plot of Partial Auto Correlation Function. There is again a steep downward trend indicating the presence of a AR(Auto Regressive) component

2) Differencing If the time series is not stationary find the value of d which turns it stationary.
3) Find the values of p and q Use the ACF (Auto-correlation),PACF (Partial Auto- correlation) and AIC (Akaike information criterion) to find the suitable values of p and q.
4) Build the Model then Test and Forecast.

### C. Applying LSTM Model

The most important part of the problem is to prepare data for LSTM. This involves three main steps:

1) Transform time series data in supervised learning problem
2) Transform the series into stationary if required
3) Scale the observation values

Transformation of time series is required because LSTM model in keras takes in data in the form of input (X) and an output (Y). This can be achieved by using the observations of the last w (window) time steps as the input and the current time step observation as the output. This windowing can be done by shifting the target by shifting the column by desired value and then appending it to the original column.

TABLE I
GENERATING SUPERVISED TRAINING DATA IN THE FORM OF INPUT (X) AND OUTPUT (Y) BY USING A WINDOW SIZE OF TWO

| Date | V1 | V2 | V3 |
|---|---|---|---|
| D1 | 5 | 8 | 10 |
| D2 | 8 | 10 | 19 |
| D3 | 10 | 19 | 11 |
| D4 | 19 | 11 | 3 |
| - | - | - | - |
| - | - | - | - |
| - | - | - | - |
| D100 | 6 | 1 | 14 |

window size=2

In the example above we have created columns V2 and V3 from V1 by shifting it one and two times respectively. Appending V2 and V3 creates an input(X) as [V1,V2] and output(Y) as [V3]. We use Dicky Fuller Test to test the stationarity of the time series. In this case the null hypothesis is rejected and the series turns out to be non-stationary. However, the first difference converts the series into a stationary time series as the mean becomes zero and the null hypothesis gets accepted. Like other neural networks, LSTMs expect data to be within the scale of the activation function used by the network. The default activation function for LSTMs is the hyperbolic tangent (tanh), which outputs values between -1 and 1. This is the preferred range for the time series data. For this very purpose we transform the data set in the range of (-1,1).To make the experiment fair, the scaling coefficients (min and max) values calculated on the training dataset are applied to scale the test dataset and any forecasts. After data preparation, we move onto designing the network architecture. In order to make sure that we hold on long term dependencies we have set the stateful = True. In stateful model, Keras must propagate the previous states for each sample across the batches.If the model is stateless, the cell states are reset at each sequence. With the

stateful model, all the states are propagated to the next batch. It means that the state of the sample located at index i, Xi will be used in the computation of the sample Xi+bs in the next batch, where bs is the batch size. By default, Keras shuffles (permutes) the samples in X and the dependencies between Xi and Xi+1 are lost therefore we have also set shuffle = False.[14] In addition, at the end of every training epoch we reset the internal state [15].We also know that the output layer consists of a single neuron with a linear activation to predict the number of hits in the future. Now, we have to decide on the number of epochs, number of hidden layers, number of neurons and the activation functions used in all the layers except the output layer.

We begin the hyperparameter tuning with the number of epochs varying them from hundred (100) to nine hundred (900). Calculating MAPE (Mean Absolute Percent Error) along the way we find two hundred (200) to the best value for the number of epochs(Fig. 7) Once the number of epochs is fixed, we move on to the number of layers. Now keeping the number of epochs equal to two hundred we vary the number of hidden layer and choose four (4) as the number of hidden layers because they resulted in the minimum error (Fig. 8). Moving forward with two hundred and four as the number of epochs and number of hidden layers, respectively, we vary the number of neurons in each hidden layer in range of [10,800] and find that one hundred(100) results in the minimum error (Fig. 9). The last thing left to do is to find out the activation function. With the number of epochs as two hundred , number of hidden layers as four , number of neurons as one hundred we vary the activation function along the set [sigmoid, tanh , ReLU , softmax] and find that ReLU performs the best (Fig. 10). ReLU might have been the better activation function because it is the best among the others to solve the vanishing gradient problem.

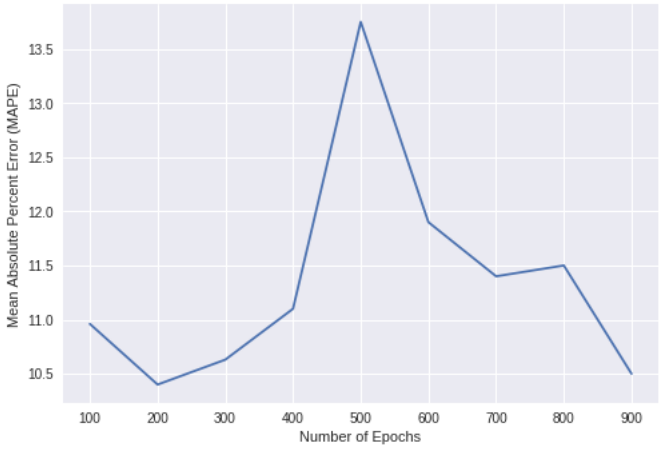


Fig. 7. Plot of MAPE (Mean Absolute Percent Error) against Number of Epochs. The optimal number of epochs can be seen to be two hundred(200) as they result in the minimum MAPE.

The model architecture after hyperparameter tuning is shown in Fig. 11.

Algorithm is as follows:

1) Transform the data in supervised learning form
2) Check Stationarity- By Dickey Fuller Test

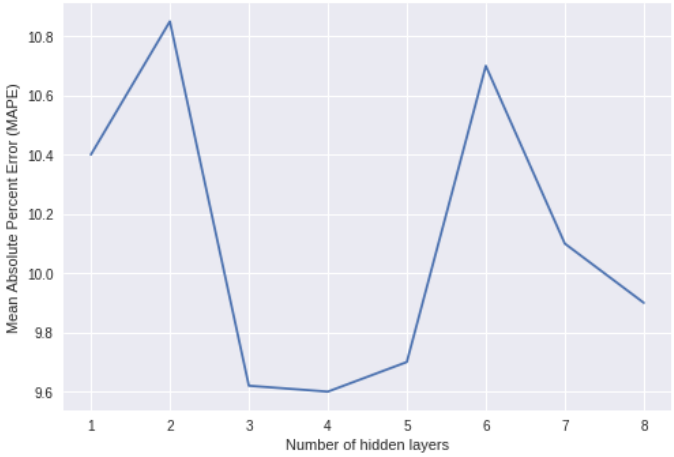


Fig. 8. Plot of MAPE (Mean Absolute Percent Error) against Number of hidden layers. The optimal number of hidden layers can be seen to be four(4) as they result in the minimum MAPE.

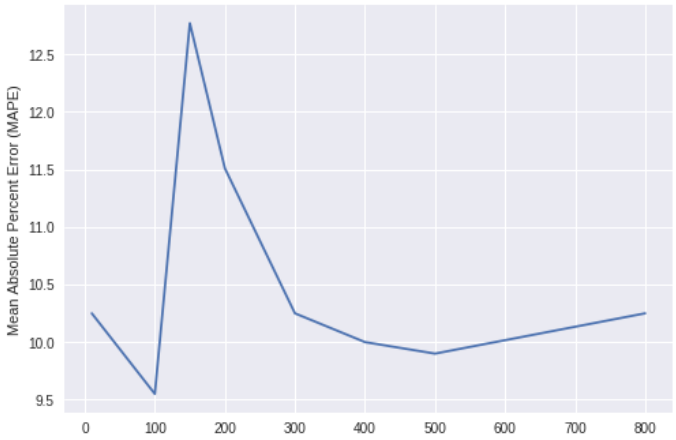


Fig. 9. Plot of MAPE (Mean Absolute Percent Error) against Number of Neurons. The optimal number of neurons can be seen to be one hundred(100) as they result in the minimum MAPE.

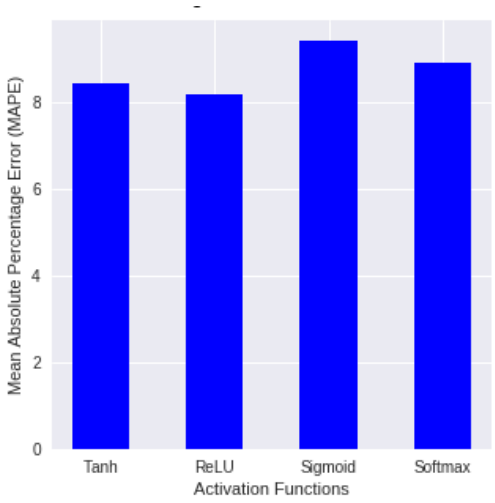


Fig. 10. : Plot of MAPE (Mean Absolute Percent Error) against Activation Functions. The optimal activation function can be seen to ReLU as it results in the minimum value of MAPE

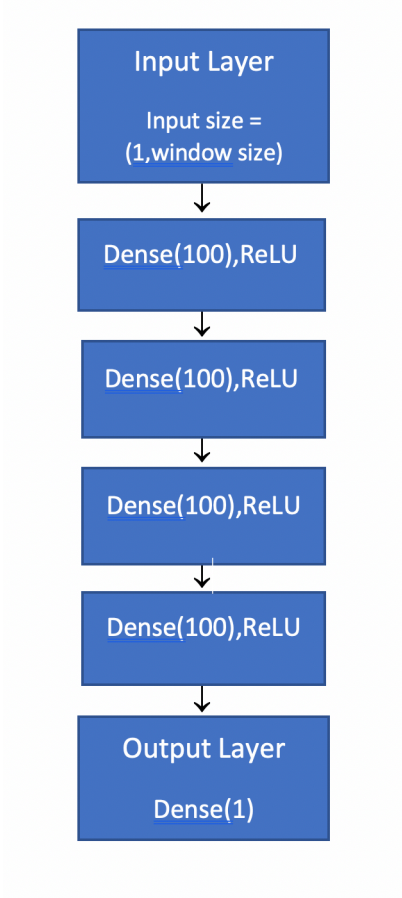


Fig. 11. Model Architecture.

3) Differencing If the time series is not stationary find the value of d which turns it stationary
4) Scale the data in the range of [-1,1]
5) Use an architecture with particular hyper-parameters
6) Train the Model.
7) Test the Model.
8) Go through 5-8 iteratively to find the best set of hyperparameters which includes the number of hidden layers, the number of neurons in each hidden layer,the activation functions,the number of epochs.
9) Forecast or Predict.

## IV. Result

When it comes to time series forecasting, some of the common error metrics chosen are SMAPE (Symmetric Mean Absolute Percent Error) and MAPE (Mean Absolute Percent Error). MAPE represents the percentage of average absolute error occurred. We have chosen MAPE because it is independent of the scale of measurementt and does not panelise extreme deviations[2, 10]. For ARIMA we chose the parameter set as p and q being equal to four and d being equal to one. We are forecasting the hits for the future days, one day at a time with the forecasted value being appended in the training data. Using ARIMA results in a MAPE score of 7.27 percent ( Fig. 12).

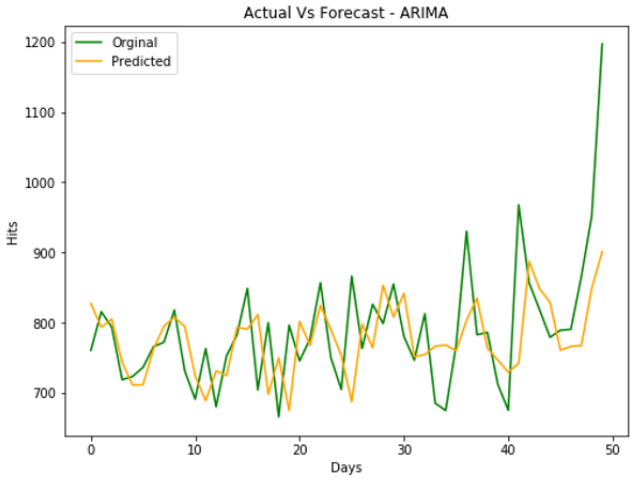


Fig. 12. Plot of Actual and Forecast Values using ARIMA. The Mean Absolute Percent Error(MAPE) of the model is 7.27 percent.

For LSTM, we choose the number of epochs as two hundred(200), number of hidden layers as four (4), number of neurons as one hundred (100) and activation function as ReLU. These parameters are used to predict a value which is inverted by scaler inverse transform to map the value from a range of [-1,1] to an actual forecast value. Using LSTM results in a MAPE score of 8.2 percent (Fig. 13)

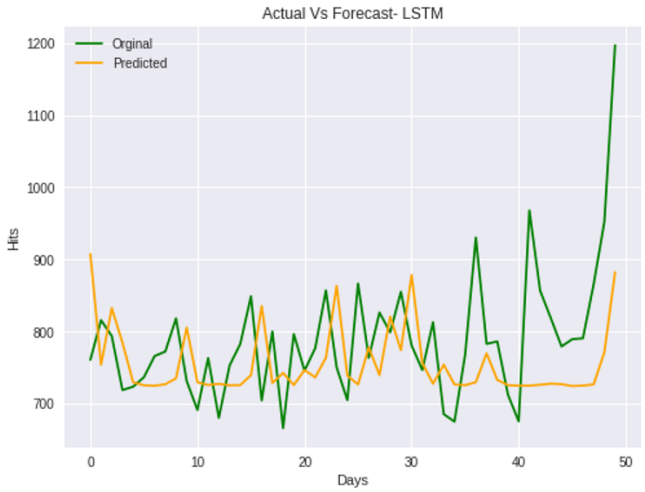


Fig. 13. Plot of Actual and Forecast Values using LSTM. The Mean Absolute Percent Error(MAPE) of the model is 8.2 percent.

## V. Conclusion

The MAPE score for LSTM is less than ARIMA because the number of data points are very less. Had there been large volumes of data LSTM would have performed much better. We repeat this analysis for all the languages and create a plot (Fig. 14).

The plot above shows that English and language Japanese have the maximum number of hits predicted in the next two months therefore the web host should invest most of its resources in hosting articles of these languages in order to get maximum profits.

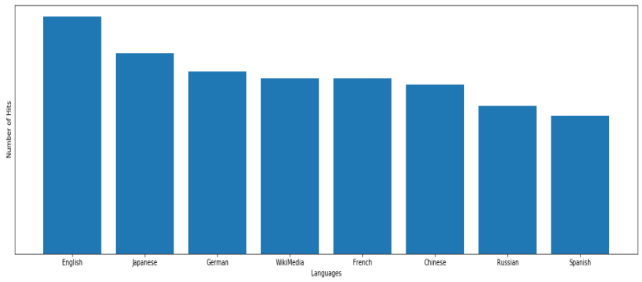


Fig. 14. Bar plot of relative frequency of hits across along each language. English and Japanese clearly seem to be the dominant ones.

## REFERENCES

[1] Burges, C.J.C., A tutorial on support vector machines for pattern recognition, Data Mining and Knowledge Discovery, 2 (1998), pages: 121-167.
[2] C. Hamzacebi, Improving artificial neural networks performance in seasonal time series forecasting, Information Sciences 178 (2008), pages: 4550-4559.
[3] G. Zhang, B.E. Patuwo, M.Y. Hu, Forecasting with artificial neural networks: The state of the art, International Journal of Forecasting 14 (1998), pages: 35-62.
[4] G.P. Zhang, Time series forecasting using a hybrid ARIMA and neural network model, Neurocomputing 50 (2003), pages: 159175.
[5] J. Lee, Univariate time series modeling and forecasting (Box-Jenkins Method), Econ 413, lecture 4.
[6] J.M. Kihoro, R.O. Otieno, C. Wafula, Seasonal Time Series Forecasting: A Comparative Study of ARIMA and ANN Models, African Journal of Science and Technology (AJST) Science and Engineering Series Vol. 5, No. 2, pages: 41-49.
[7] Joarder Kamruzzaman, Rezaul Begg, Ruhul Sarker, Artificial Neural Networks in Finance and Manufacturing, Idea Group Publishing, USA.
[8] John H. Cochrane, Time Series for Macroeconomics and Finance, Graduate School of Business, University of Chicago, spring 1997.
[9] K.W. Hipel, A.I. McLeod, Time Series Modelling of Water Resources and Environmental Systems, Amsterdam, Elsevier 1994.
[10] L.J. Cao and Francis E.H. Tay Support Vector Machine with Adaptive Parameters in Financial Time Series Forecasting, IEEE Transaction on Neural Networks, Vol. 14, No. 6, November 2003, pages: 1506-1518.
[11] T. Raicharoen, C. Lursinsap, P. Sanguanbhoki, Application of critical support vector machine to time series prediction, Circuits and Systems, 2003. ISCAS 03.Proceedings of the 2003 International Symposium on Volume 5, 25-28 May, 2003, pages: V-741-V-744. [32] T. Van Gestel, J.A.K. Suykens, D. Baestaens, A. Lambrechts, G. Lanckriet, B. Vandaele, B. De Moor, and J. Vandewalle, Financial time series prediction using least squares support vector machines within the evidence framework, IEEE Trans. Neural Networks, vol. 12, no. 4, pp. 809 - 821, July 2001.
[12] [Online]. Available: https://people.duke.edu/ rnau/411arim3.html
[13] [Online]. Available: https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21
[14] [Online]. Available: http://philipperemy.github.io/keras-stateful-lstm/
[15] [Online]. Available: https://machinelearningmastery.com/time-series-forecasting-long-short-term-memory-network-python/
[16] [Online]. Available: https://arxiv.org/ftp/arxiv/papers/1302/1302.6613.pdf