JOURNAL OF
EMERGING INVESTIGATORS

Article

Deep sequential models versus statistical models for web traffic forecasting

Izhan Ali[1], Uzma Mushtaque[2]

[1] Niskayuna High School, Niskayuna, New York

[2] Department of Computer Science, Rensselaer Polytechnic Institute, Troy, New York

SUMMARY

Web traffic forecasting is a challenging problem for most websites; however, accurate forecasting leads to improved customer experience. Most websites collect data on the number of visitors visiting their site each day. This is a time series problem and there are various techniques that can be used to make accurate forecasts using this data. Machine learning (ML) and deep learning (DL) techniques (also termed artificial intelligence (AI)) are used in various domains for time series analysis. However, before the advent of AI, many statistical techniques were utilized in solving forecasting problems. Due to the sophisticated structure of DL models, we hypothesized that deep sequential models would perform better on time series data for web traffic forecasting as a multi-horizon problem than traditional statistical models like Auto-regressive integrated moving average (ARIMA) and Seasonal auto-regressive integrated moving average (SARIMA). We analyzed Wikipedia web traffic data to compare the performance of statistical models with that of the popular deep sequential models like Long short-term memory (LSTM), gated recurrent units (GRU), convolution neural networks (CNN) and bi-directional LSTM (BiLSTM). Under the two-evaluation metrics used in this study, the GRU and BiLSTM models showed the best performance. In general, DL models outperformed statistical models for web-traffic forecasting. Therefore, our data supports the hypothesis that deep sequential models are a better choice compared to statistical models for web traffic forecasting. Our results can be used to further investigate the performance of deep sequential models for time series analysis under other domains.

INTRODUCTION

Of particular interest to website providers is the problem of web traffic forecasting. Websites require an accurate demand forecast (number of visitors visiting the website) for many days in the future in order to provide high quality service to customers (1). This problem is a time series forecasting problem, which can be defined as a process of using past and present data to predict (forecast) future datapoints (2). Depending on the number of future steps to be predicted, this problem can be formulated for any number of time periods in the future. For a single time period it's referred to as a single horizon problem, and for multiple time periods it's called a multi-horizon time-series problem.

There are two sets of models used in time-series forecasting: the traditional parametric statistical models and the more recent deep learning (DL) based data driven techniques. Statistical models are based on defining mathematical relationships built on assumptions underlying the data (2). There are various statistical models used to determine future values of a time series. We used auto-regressive integrated moving average (ARIMA) and seasonal auto-regressive integrated moving average (SARIMA). These two models were chosen because both are built on the well-known auto-regressive moving average (ARMA) model which is mathematically shown to be sufficient for modeling a stochastic process like a time series process (3). The ARIMA *(p, d, q)* model is an autoregressive integrated moving average process, which is the combination of an autoregressive process (AR; of order p), an integration process (I; differenced d times), and a moving average process (MA; of order q) (2). The model states that the present value of the variable (to be predicted) is dependent on the past values that come from the AR portion and past errors coming from an MA process (2). The series is converted to a stationary series by differencing order d, which makes up the integrated part of the model. Seasonality can be defined as a characteristic of time-series in which the data experiences a regular and predictable change that recurs over a fixed time (for example, every year) (2). The SARIMA $(p, d, q)\ (P,D,Q)_m$ model is the seasonal counterpart of the ARIMA model with the additional set of parameters $(P,D,Q)_m$ that are the seasonal counterparts of (p, d, q) (2). Because seasonality is cyclical, parameter m determines the frequency of the repeat of the seasonal cycle.

DL approaches have gained significant attention due to the ability of these models to learn data representations and solve complex problems in areas such as computer vision, natural language processing, and speech recognition (4). We focused on sequential DL models that are primarily used with sequential data, for instance with problems involving forecasting using time series data. Specifically, recurrent neural networks (RNNs) are DL models used for time series type sequential data. RNNs are basic building blocks of various other types of DL models that have shown better performance with sequential data due to the general order that exists with such problems (5). In our experiments we worked with models that fell under three categories of sequential models: long short-term memory (LSTM), gated recurrent units (GRU) and convolution neural networks (CNN). Multiple studies show their effectiveness with temporal data (2, 4, 6, 7). LSTM is a well-known deep sequential model with a key feature being their ability to capture long-term dependencies in sequential data while mitigating some of the known issues with longer sequences such as the vanishing gradient problem that can

https://doi.org/10.59720/24-025

affect traditional RNNs (8). This makes LSTM especially effective in tasks where understanding and remembering information from earlier time-steps is crucial. LSTM model achieves this by introducing a memory cell (that has gates for effective information flow) that can store and retrieve information over long sequences (4). The Stacked LSTM is a variant of LSTMs in which multiple LSTM cells are stacked on top of one another, and those cells share common information. The output of one LSTM layer is used as the input to the next LSTM layer in the stack. This process is repeated for all the LSTM layers in the stack. BiLSTM are another advanced variant of the LSTM with two LSTM layers (8). Sequential information is stored in both directions leading to enhanced ability to capture more information. GRU is a sequential model and is often favored over LSTMs in applications requiring computation efficiency because they have a simpler architecture with fewer gates, making them computationally less expensive and sometimes easier to train (6). CNN models are used for image processing tasks (4). However, these have been successfully used in various time series studies due to their ability to learn temporal information easily. CNNs can extract complex features easily without the intervention of a human.

Various comparative studies conducted in different domains concluded that DL models like LSTMs perform better than their traditional counterparts like ARIMA and its variants (6, 9). On the other hand, there are different set of comparative analyses performed on stock exchange data that reveals the ARIMA model as a better performer when compared to LSTMs (10). Additionally, there are various hybrid approaches that demonstrate the effectiveness of combining models to approach the time series forecasting problem (7). Specifically in stock market forecasting, there is no clear distinction on effectiveness when comparing the variants of ARIMA with DL techniques (11). Similarly in the web traffic prediction domain, there are studies that compare various ML models for web traffic prediction (12). Additionally, there are studies that focus on different procedures for network traffic prediction (13). Both types of studies come to different conclusions (12, 13).

With increasing evidence pointing towards the higher effectiveness of one set of models over another based on the application domain and experimental set-up, our goal in this study was to establish which set of models is more effective for web traffic prediction for a busy website, using Wikipedia as our data source. Establishing effective prediction techniques for web traffic data is important because general network management techniques face the challenge of monitoring web-traffic in real-time. Through this comparative study, we additionally emphasized the relevance of model selection in implementing and/or adopting any ML or DL technique in a particular domain. There are two distinct pathways suggested for network traffic management: active or passive (14). Our contribution falls under the latter category because we are focusing on daily average web traffic predictions. We focused on the multi-horizon time series problem in which we predicted the variable of interest (average number of users) for multiple future time steps. We hypothesized that under the domain of web traffic forecasting, deep sequential models will outperform traditional statistical models on any given performance metric when forecasting a multi-horizon time series.

RESULTS

We performed a comparative analysis between two sets of models: deep sequential models and statistical models to test our hypothesis that deep sequential models will outperform statistical models on the web-traffic forecasting problem. We used the Web Traffic Time Series Forecasting Dataset from the Kaggle website (15). Our goal was to predict the average number of users that visited a particular website; therefore, we calculated the average number of users across all Wikipedia pages and created a new dataset with this accumulated information. Our variable of interest was the calculated average number of total site users/visitors per day. We found that there were no missing observations in the dataset. We visualized the monthly and annual user count to identify trends and did not observe any trends in the monthly average (**Figure 1**). For the weekly average, the trend is similar for both 2015 and 2016 except for a few weeks during mid-2016 (**Figure 2**). We used the same data set to compare the statistical and DL models.

The first step in statistical time-series forecasting was to determine if the series was stationary, i.e. statistical properties like mean, variance and auto-correlation do not change over time. The pre-processed dataset was found to be non-stationary using the augmented Dickey-Fuller (ADF) test, therefore we applied a first-order differencing transformation (2). Next, using the autocorrelation function (ACF) and partial autocorrelation function (PACF) we determined that the time series is an ARIMA (4,0,6) process. Here the three numbers represent the order of the auto-regressive process, order of differencing and order of moving average underlying the series. We found the ARIMA (4,0,6) model to be the best fit for this time series by using Akaike information criterion (AIC). The AIC is a mathematical method for evaluating how well a model fits the data it was generated from. Our analysis indicated that the residuals have no trend with a variance that seems constant over time, which resembles the behavior of white noise. This is further confirmed by the Q-Q plot, which displays a straight line that lies on y = x with a few exceptions

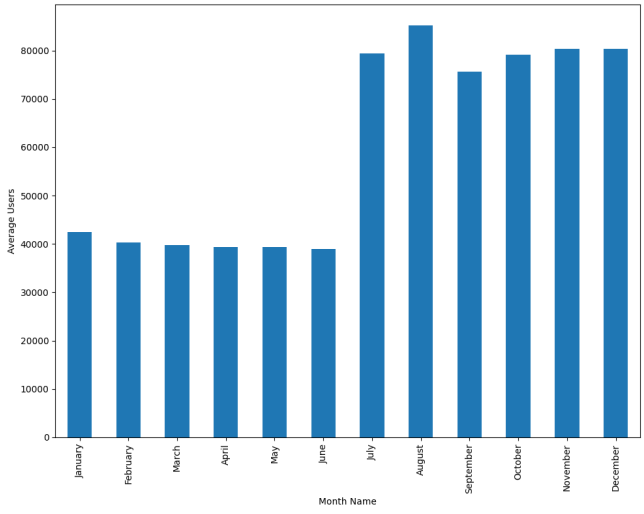

Figure 1: Average number of visitors visiting Wikipedia website. Data shows visitors by month for 2015 and 2016. The first six months have a comparable average value for visitor number of a little above 40,000 visitors. From July to December the numbers are higher than in the first 6 months with values of around 80,000 visitors on average.

https://doi.org/10.59720/24-025

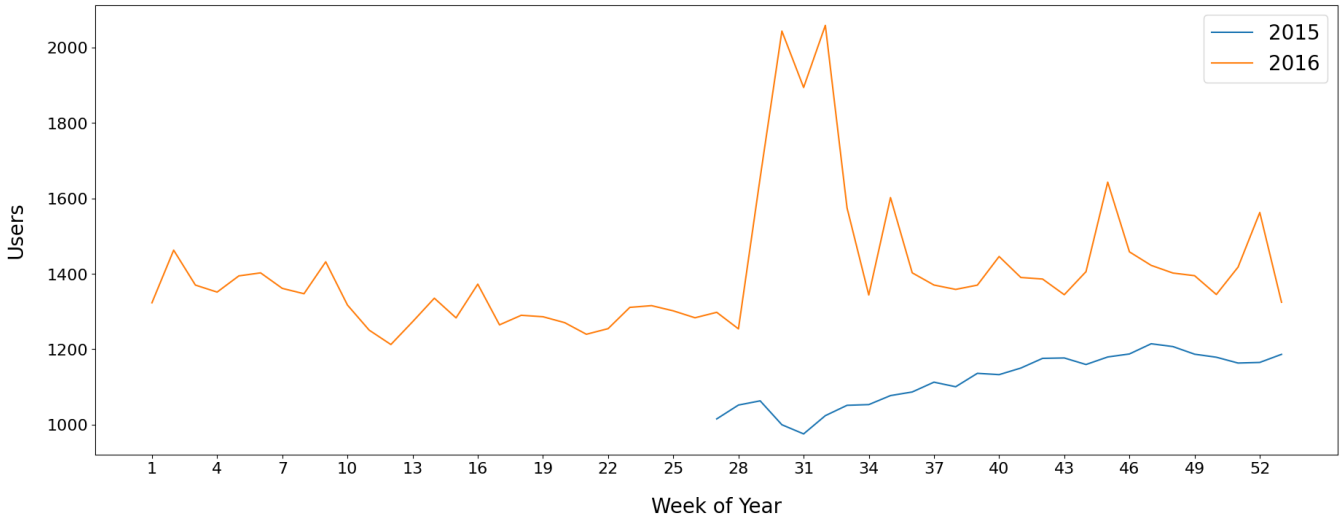

Figure 2: Average weekly users for 2015 and 2016. Average number of users every 3 weeks over the 1.5 year period. The x-axis represents the number of weeks from 1-52 and the y-axis represents the number of users. The dataset has values starting from July 1st, 2015, up until December 31st, 2016.

on the tail. The correlogram and the box test both confirmed that there is no autocorrelation, the residuals are like white noise, and the model can be used for forecasting. A similar set of steps were performed to fit a SARIMA model to the data. The model with the lowest AIC was selected and it was found to be SARIMA (2,0,1) (0,0,2,52) where the model parameters are (p, d, q) and the second set is (P, D, Q, s) for the seasonal component. Both models were used to predict the average number of daily visitors for the next 55 days.

Model building for DL models is an iterative process with various hyperparameters to be selected (**Table 1**). Finally, all models were used to predict the last 55 weeks of the average number of visitors visiting the website. To evaluate the performance of the models, we used two evaluation metrics. The first one was mean squared error (MSE) that captures the average squared difference between actual and predicted values for the prediction period for time-series analysis. A lower MSE suggests that the model's predictions are closer to the actual values, while a higher MSE indicates larger prediction errors and less accuracy in the forecasting model (4). The second metric was mean absolute percentage error (MAPE). This is the average percentage difference between predicted and actual values for the entire prediction period. MAPE is a useful metric for assessing the relative accuracy of a forecasting model since it accounts for the magnitude of errors relative to the actual values. A lower MAPE indicates better model accuracy, which means the model's predictions are, on average, closer to the actual values in terms of percentage error (2).

Our goal was to compare various statistical techniques with sequential DL models and evaluate the results across two different well-known metrics: MAPE and MSE. The lowest MSE of 14,757.255 was observed for the GRU model and the lowest MAPE of 5.25 was observed for the BiLSTM model, which are both DL models (**Table 1**). Further calculating the average of the MSE for statistical models gives a value of 16,082.06 (**Table 1**). On the other hand, the mean MSE for DL models is 15,987.71 (**Table 1**). Therefore, the average mean MSE value for DL models is lower than statistical models. Similarly, the average MAPE for both sets of models was 6.6274 (statistical models) and 5.9727 (DL models) (**Table 1**).In conclusion, our experimental results support our hypothesis that deep sequential models outperform statistical models for the multi-horizon web-traffic forecasting time series problem.

To establish statistical significance of our results we performed a non-parametric test called Mann-Whitney U test. This test was used to compare the difference in the results of statistical models versus DL models along the two-evaluation metrics. Given we do not have enough data, the results were inconclusive. In summary, MSE data suggested the difference is significant while MAPE data suggested the difference is not statistically significant.

Model	MAPE	MSE
ARIMA (4,0,6)	6.67933821240378	15,588.468797608328
SARIMA (2,0,1) (0,0,2)	6.575494880259815	16,575.65173498912
LSTM (128,64,1)	5.907143377972659	15,358.371555816706
Stacked LSTM-3 stacks	6.401187486266502	19,441.96353736812
GRU	5.97025980140007	14,757.255738758653
Stacked GRU-3 stacks	6.3279812864464295	16,439.137253628924
Bi-LSTM (128,32)	5.250967303795877	14,801.56006785671
CNN (128, MaxPool)	5.978940617115489	15,128.013928574515

Table 1: MAPE and MSE values. Numeric values for both evaluation metrics for each model. Model description (with hyperparameters) and results for both evaluation metrics are shown. These values are calculated from the results obtained on the test-dataset for each model.

DISCUSSION

With growing use of the internet, accurate web traffic forecasting is an important problem for service providers. Without proper resources in place websites experience delays that lead to longer customer wait times. There is an increasing need for better web-traffic forecasting. DL models are known to show excellent results in various domains like text, images, recommendations, etc. Additionally, DL models have outperformed various statistical models, even

https://doi.org/10.59720/24-025

on traditional regression tasks. Time series forecasting is a problem that can be solved using sequence to sequence architectures of DL models. In this study, we utilized a few of those architectures to conduct experiments with a publicly available dataset on web-traffic forecasting. Our results indicate better performance (lower average MSE and MAPE) of deep sequential models (LSTM, Stacked LSTM, BiLSTM, GRU, Stacked GRU, CNN) as compared to statistical models (ARIMA and SARIMA). The MSE difference was found to be statistically significant, but the MAPE difference was not statistically significant. One reason for this could be that we used a single dataset for this study. Although significance was not found across all measures, considering the results of our experiments and the calculated statistical values like average MAPE and MSE of both groups of models and minimum value MAPE and MSE of both groups we can conclude that our results support our hypothesis that deep sequential models perform better than statistical models in multi horizon web traffic forecasting problems.

Statistical models do not perform well when the data is non-stationary or weakly stationary (2) . For a longer time series, sometimes the data does not adhere strictly to the assumptions of these statistical models that may also lead to poor performance (2). Many of the deep sequential models can learn long and short-term dependencies in the target variables better than statistical models (9). This explains their better performance in each of the experiments (9). Among the DL models used, LSTMs and GRUs performed better than others because they can effectively store and access long term dependencies through their memory cell and gate structure (6).

Our findings indicate better performance of deep sequential models for average web-traffic prediction on a given website. In general, all models under the category of deep sequential models depicted a lower MAPE as compared to both statistical models. We observed a similar trend when calculating MSE, except for the Stacked LSTM and Stacked GRU models. This could be a special case of overfitting, which is a known issue with more complex models. Overfitting happens where a model fits so well on the training data that it almost mimics it and does not generalize well on the unseen test data (4). After analyzing the average MSE and MAPE in general, our findings indicate that deep sequential models are a better choice for web-traffic forecasting problems.

We explored multiple DL architectures under specific configurations only. Selecting configurations that produce the best results (hyper-parameter tuning) under a more rigorous model selection process may lead to more insights in this direction. In this study, we used the well-known optimizer; Adaptive Moment Estimation for optimizing all DL algorithms. Early stopping is a technique used to avoid overfitting (4). For regularization, early stopping was used across all DL models. Therefore, our experiments and hence our results are dependent on this set-up. Exploring other set-ups can lead to a better understanding of DL model performance.

Through this study we provide an in-depth comparison of two classes of models. In particular, DL is one of the fastest growing fields in recent times. The DL models explored in this study capture temporal dependency in data by maintaining an internal state that can store information over long periods of time. In some scenarios, not every time step carries equally useful information. To solve this problem, there are more sophisticated transformer models (16). Attention-based models involve the use of transformers and transfer learning techniques that have shown superior performance in various prediction tasks including time series forecasting. Instead of processing the entire input sequence in every step, attention mechanisms allow the model to selectively process different parts of the input sequence based on their relevance to the current time step (17). In our future research, we aim to include an analysis of these models as well.

Our study involved an aggregate approach to web-traffic forecasting (i.e., we considered univariate time series only). In our future research, studying patterns across multiple time series can provide further insights into this problem. Another important step in this direction is to explore hybrid models (that combine statistical and DL models) to achieve better forecasts.

Literature does not provide clear evidence about which model (statistical versus DL models) performs better than the other for different time series prediction tasks. In this research, we conducted an in-depth comparison of statistical and deep sequential models for web traffic forecasting. Our results suggest DL models are a better choice when websites are looking for more accurate forecasts for a multi-horizon time series prediction. These models can learn more complex patterns from temporal data and generalize them well, especially with larger datasets. However, it's important to note that these results depend on the specific problem, the amount of data available, data pre-processing techniques, computational methods, configuration of the models and the evaluation of results. Hence the results obtained in this study may not be transferable to every time series problem in general. Regardless, this study can be used as a model selection guideline when facing time series prediction tasks specifically for web traffic forecasting.

MATERIALS AND METHODS

The general mathematical problem that is solved in this study falls under this category of time series forecasting. The mathematical model is summarized here (for *k* steps ahead forecast and *n* steps input):

$$y_{i,t+k}^{predicted} = f(y_{i,t-n:t}^{actual}, x_{i,t-n:t})$$

The left-hand side of the equation is the predicted value (average predicted users) for k time steps. The function on the right-hand side signifies the model function (learnt by the algorithm using historical data – first term on the right-hand side, and any extra variables – second term on the right-hand side –if present) (2).

The dataset used in this study is called Web Traffic Time Series Forecasting Dataset (15). It is a publicly available dataset that can be downloaded from Kaggle. It involves multiple time series (145,000 Wikipedia pages). Each of these time series represents several daily views of a different Wikipedia page, between July 1st, 2015, and December 31st, 2016. For each time series, the name of the article as well as the type of traffic that this time series represents (all, mobile, desktop, spider) is also provided. There are two files given. We used train_1.csv (stage 1 file only). Here the word stage has no meaning with respect to the problem. In the original Kaggle

https://doi.org/10.59720/24-025

competition, data was provided in two stages and hence the name. We solved the time series forecasting problem by preprocessing the given dataset and calculating the average number of users across all pages on the Wikipedia website for each day. This was done by adding the total number of visitors on all pages divided by the number of pages for each day. Our final pre-processed dataset has 550 observations and 2 columns (Date and Average Users). Before the model building stage, we performed a 90-10 train test split on the pre-processed dataset. This means 90% of the data is used to train each model and the last 10% (equivalent to 55 days) is used to test the models.

We used two statistical models (ARIMA, SARIMA) and six well-known DL models (LSTM, Stacked LSTM, GRU, Stacked GRU, Bi-LSTM, CNN) for this analysis. Both the statistical models (ARIMA and SARIMA) were implemented using the Python module called statsmodels (18). For implementing the statistical models, we followed the procedure outlined in (2). First, we conducted the ADF test and by using its results we determined that the time series data in its original form was not stationary. Therefore, the first order differencing was applied to make it stationary. Next, we plotted the ACF and PACF plots to determine the type of process underlying the given time series. The sinusoidal effect observed pointed to no pattern (i.e., neither a complete MA process nor a complete AR process). Therefore, we fitted the ARIMA model by determining the appropriate coefficients. An ARIMA process is the combination of an AR(p), integration I(d), and the MA(q). The ARIMA process uses a differenced/transformed series and assumes that the present time series value is dependent on past values, coming from the AR(p) portion, and past errors, coming from the MA(q) portion (2). We found the model parameters (p, d, q) iteratively by evaluating the AIC. The AIC value determines the best model in comparison to other similar models. To validate the assumptions of the model we performed residual analysis, visualized Q-Q plots and performed a Ljung box test.

To implement the DL models, we used the Keras library provided by the TensorFlow framework (19). This is a popular DL open-source platform which is used for complex numerical computational tasks and to implement different DL algorithms. We used this library to implement the DL models. For all these models, the data must be split into equal sequences that are used to predict a value or a set of values. Then these set of values are further used as input (in addition to the data) for the next sequence of predictions. For the current set of experiments, we used every two days of data to predict the average users for the next day. We created a function called create_data that can be used to run other configurations of input and output sequences, but we restricted our experimentation to these values only. Because our goal is to compare models, we kept the hyperparameters constant for each model (18). For DL models, we scaled our data using the MinMaxScaler from the sklearn library (20). MinMaxScaler transforms features by scaling them within a given range. We also used validation loss to perform a process called early stopping for all our models (4). This process prevents models from overfitting the training data and ensures better prediction on unseen (test) data by monitoring the value of the Loss function on a randomly selected validation set. In DL models, hyperparameters are input parameters decided by the experimenter depending on the problem at hand. We performed small independent experiments with each model to arrive at these hyperparameters, namely number or layers, number of neurons in each layer, type of activation function, number of batches and epochs to run. None of the known systematic hyperparameter tuning techniques were used.

The software used was a personal version of the Anaconda Software (available for free). The language used was Python (version 3.11). Other prominent Python libraries that were used for analysis are NumPy, Pandas and Scikit-learn (version 1.3.0). The data preprocessing was done by utilizing methods and functions provided by the NumPy and Pandas libraries in Python. We performed an in-depth data visualization using libraries called Matplotlib and Seaborn. All the files pertaining to the methods used and results obtained can be found in the following repo: https://github.com/IzhanAli08/TimeSeriesPaper/tree/main.

Received: October 23, 2023
Accepted: September 17, 2024
Published: July 20, 2025

REFERENCES

1. Casado-Vara Roberto, et al. “Web traffic time series forecasting using LSTM neural networks with distributed asynchronous training.” *Mathematics*, vol. 9, no. 4, 2021, pp. 421. https://doi.org/10.3390/math9040421
2. Peixeiro M. “Time Series Forecasting in Python.” Simon and Schuster, 2022.
3. Box, George EP, et al. “Time series analysis: forecasting and control.” John Wiley & Sons, 2015.
4. Murphy Kevin P. “Probabilistic Machine Learning: An Introduction.” Cambridge: MIT Press, 2022.
5. Medsker, Larry R., and Lakhmi Jain. “Recurrent neural networks.” *Design and Applications*, vol. 5, no. 64–67, 2001, p. 2.
6. Cahuantzi R, et al. “A comparison of LSTM and GRU networks for learning symbolic sequences.” *Science and Information Conference*. 2023, p. 771–785. https://doi.org/10.1007/978-3-031-37963-5_53
7. Abdulrahman Ufi, et al. “A hybrid ARIMA-LSTM model for stock price prediction.” *International Journal of Computer Engineering and Information Technology*. vol. 12, no. 8, 2021, pp. 48–51. https://libproxy.rpi.edu/login?url=https://www.proquest.com/scholarly-journals/hybrid-arima-lstm-model-stock-price-prediction/docview/2565213234/se-2
8. Wang, Shuzhen. “A stock price prediction method based on BiLSTM and improved transformer.” *IEEE Access*. 2023. https://doi.org/10.1109/ACCESS.2023.3296308
9. Siami-Namini S, et al. “A comparison of ARIMA and LSTM in forecasting time series.” *17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 2018, pp. 1394–1401. https://doi.org/10.1109/ICMLA.2018.00227
10. Kobiela, Dariusz, et al. “ARIMA vs LSTM on NASDAQ stock exchange data.” *Procedia Computer Science*, vol. 207, 2022, pp. 3836-3845. https://doi.org/10.1016/j.procs.2022.09.445
11. Dwivedi SA, et al. “Analysis and forecasting of time-series data using S-ARIMA, CNN and LSTM.” *2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*. IEEE, 2021, p. 131–136. https://doi.org/10.1109/ICCCIS51004.2021.9397134

https://doi.org/10.59720/24-025

12. Oliveira TP, et al. “Computer network traffic prediction: a comparison between traditional and deep learning neural networks.” *International Journal of Big Data Intelligence*. vol. 3, no. 1, 2016, pp. 28–37. https://doi.org/10.1504/IJBDI.2016.073903
13. Szostak D, et al. “Machine learning classification and regression approaches for optical network traffic prediction.” *Electronics*. vol. 10, no. 13, 2021, pp.19-41. https://doi.org/10.3390/electronics10131578
14. Abbasi M, et al. “Deep learning for network traffic monitoring and analysis (NTMA): A survey.” *Computer Communications*. vol. 170, 2021. https://doi.org/10.1016/j.comcom.2021.01.021
15. “Web Traffic Forecasting Data. Google” *Kaggle*. https://www.kaggle.com/competitions/web-traffic-time-series-forecasting/data . Accessed 05 Sep. 2023.
16. Weerts, et al. “Importance of tuning hyperparameters of machine learning algorithms.” *arXiv Preprint,* 2020. https://doi.org/10.48550/arXiv.2007.07588
17. Vaswani, Ashish, et al. “Attention is all you need.” *Advances in neural information processing systems* 30, 2017. https://dl.acm.org/doi/10.5555/3295222.3295349
18. Seabold S, Perktold J. “Statsmodels: econometric and statistical modeling with Python.” *SciPy*. vol. 7, no. 1, 2010.
19. Abadi, Martín, et al. “TensorFlow: Large-scale machine learning on heterogeneous systems, software available from TensorFlow. org.” 2019.
20. “Scikit-learn: Machine Learning in Python.” https://scikit-learn.org/stable/. Accessed 01 Sep. 2023.