Fall 2023

Web Traffic Time Series Forecasting

Summanth Redde Mulkkalla
San Jose State University

Web Traffic Time Series Forecasting

A Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Summanth Redde Mulkkalla

December 2023

The Designated Project Committee Approves the Project Titled

Web Traffic Time Series Forecasting

by

Summanth Redde Mulkkalla

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSÉ STATE UNIVERSITY

December 2023

Dr. Genya Ishigaki	Department of Computer Science
Dr. Navrati Saxena	Department of Computer Science
Dr. Saptarshi Sengupta	Department of Computer Science

ABSTRACT

Web Traffic Time Series Forecasting

by Summanth Redde Mulkkalla

Online web traffic forecasting is one of the most crucial elements of maintaining and improving websites and digital platforms. Traffic patterns usually predict future online traffic, including page views, unique visitors, session duration, and bounce rates. However, it is challenging to forecast non-stationary online web traffic, particularly when the data has spikes or irregular patterns. This non-stationary property demands a more advanced forecasting technique. In this study, we provide a neural network-based method, Spiking Neural Networks (SNNs), for dealing with the data spikes and irregular patterns in non-stationary data.

In our study, we compared the forecasting results of SNNs with traditional and popular time-series prediction methods like Long Short-Term Memory (LSTM) networks and Seasonal AutoRegressive Integrated Moving Average with exogenous variables (SARIMAX). The evaluation was based on prediction error metrics such as Mean Square Error (RMSE) and the Mean Absolute Error (MAE). Our results found that SNNs worked better in forecasting the non-stationary web traffic data when compared to the traditional methods. This effective forecasting technique by SNNs can be crucial in sectors like e-commerce and digital marketing, where accurately predicting the traffic helps optimize resources and improve digital strategies.

Keywords: Time Series, Traffic Forecasting, Predictive Learning, Non-Stationarity, Spiking Neural Networks, Long Short-Term Memory.

ACKNOWLEDGMENTS

I am deeply appreciative of Dr. Genya Ishigaki, Dr. Navrati Saxena, and Dr. Saptarshi Sengupta for their continuous support and essential counsel during this study. Their knowledge and perceptions have greatly influenced the course and outcome of our work. Their assistance has not only improved the project's quality but also tremendously aided in my own and my career's development.

I am incredibly appreciative of their mentoring and the chance to work under their direction.

TABLE OF CONTENTS

CHAPTER

CHAPTER 1

Introduction

A time series prediction problem involves forecasting future values based on observed data collected over time. This data is sequential, with measurements taken at regular intervals, capturing trends, seasonal patterns, and other temporal dynamics. The goal is to build a model that can accurately predict future points in the series using statistical, machine learning, or computational techniques [1]. These predictions are crucial in finance, weather forecasting, and supply chain management. Time series prediction is challenging due to factors like noise in data, non-stationarity, and the complexity of underlying patterns [2]. Therefore, adequate time series models must handle these challenges while providing reliable forecasts.

Over the last few years, we have seen an increase in usage of the internet to browse a large number of websites. A data point is created whenever visitors visit a website at a time stamp. Since the internet's inception, many websites have generated a vast amount of data each time a person accesses [3]. Web time series data is the compilation of these observations across time [4].

There are multiple ways in which this time series data is being recorded. One type of data is web traffic data. Web traffic data refers to data collection whenever a user visits or interacts with websites over the internet. The data can be the number of visitors to a website, the pages they visit, how long they stay on those pages, and where they came from (e.g., search engines, direct visits, or referral sites). Additionally, web traffic data can include more detailed analytics such as user behavior patterns, click-through, bounce, and conversion rates. This web traffic data helps provide essential insights like how users engage the website, website behavior, and website performance [5].

Web traffic data differs from traditional data types in that it often exhibits

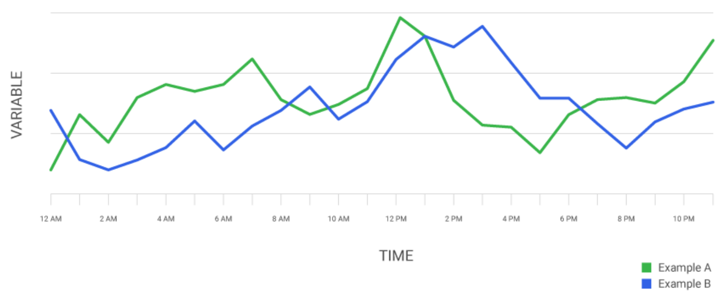

Figure 1: Example Image of a Time Series Data over a period of time [6]

non-stationary properties, meaning its statistical characteristics like mean, variance, and autocorrelation are not constant over time. The non-stationarity in web traffic data is primarily due to seasonal trends and seasonal patterns, which are regular variations in online user behavior that occur at specific periods throughout the year. Holidays, weather patterns, and consumer behavior are a few examples of the variables that can affect these trends and spikes in traffic due to noise and randomness that change user behaviors over time. Traditional data analysis methods may need help with this type of data as they often rely on the assumption of data being consistent over time. The unpredictable nature of web traffic, with its varying peaks and troughs, requires more dynamic and flexible analytical approaches to adapt to these rapid changes and provide accurate insights [7].

Spike Neural Networks (SNNs) [8] are particularly effective for analyzing web traffic data characterized by seasonal spikes and trends due to their temporal sensitivity. SNNs, which mimic the functioning of biological neurons, excel in identifying and adapting to patterns where the timing and sequence of events are crucial. This is key for web traffic analysis, where patterns and trends can shift rapidly. SNNs can adapt to these changes more effectively than standard neural networks, providing more accurate predictions and insights into dynamic, time-sensitive data [9]. Their biologically-inspired architecture allows them to respond to the sporadic and unpredictable nature of web traffic, making them a better fit for such complex, evolving datasets.

This study is primarily focused on evaluating the effectiveness of Spiking Neural Networks (SNN) in analyzing web traffic data and comparing their performance with other machine learning technologies such as LSTM (Long Short-Term Memory), SARIMAX (Seasonal Auto-Regressive Integrated Moving Average) and their variants. Inspired by previous research [10], the main agenda is to specifically test the capabilities of SNNs in handling the dataset's non-stationary properties, such as seasonal trends and unpredictable spikes, and to benchmark these results against the algorithms above. This comparative analysis aims to highlight the strengths and limitations of SNNs in the context of complex web traffic data.

Let us discuss the flow of the paper. Chapter 2 discusses different types of time series data and reviews previous research in this area, establishing a foundational understanding of the field. In Chapter 3, we discuss the methodologies used in the study, detailing the theoretical understudy. Chapter 4 thoroughly examines the dataset, including preprocessing steps and the analysis of results. Finally, Chapter 5 is dedicated to discussing potential avenues for future work.

CHAPTER 2

Background Work

This chapter discusses about the background of the research and related work that has been researched in depth.

2.1 Background of time series

In recent years, there has been a significant increase in the usage of various online websites. Every time a user uses a website for different purposes, data is stored whenever a user registers on websites or clicks on products, or uses them for streaming purposes. This data plays a significant role in analyzing the website's users by providing valuable product insights. Due to this, there has been an increase in research in machine learning by using the data to predict or provide valuable insights [7].

The data that has been collected can be used to serve and predict a wide range of insights and outcomes. Some of them are:

- **User Behaviour:** Utilizing users' past interests and preferences based on their interactions helps predict future user behavior.
- **Product Recommendation:** Product recommendations that a user might likely be interested in based on their past behavior [11].
- **Ads:** Data can be used to create more likely ads based on user's interests.
- **Fraud Detection:** Detecting potentially fraudulent activities such as fraudulent transactions.
- **Traffic and Usage patterns:** Predicting website or application traffic to allocate resources optimally.
- **Conversion Rates:** Predicting conversion rates for various goals such as completing online purchases or downloading the app.
- **Content Engagement:** Predicting how long a user can be engaged to specific

content based on the user's interests.

Time series data is the data that has been collected over some time; this data can be spaced at uniform intervals like hourly, monthly, and yearly.

The different types of time series data are

- **Uniform Time series:** This data type has a single observation over time, like temperature.
- **Multivariate Time series:** This data type has multiple variables observation over time.
- **Discrete Time series:** Observations are recorded at distinct and separate points like stock prices.
- **Event Driven Time Series:** This series of Observations, like transaction logs and earthquakes, don't occur regularly.
- **Seasonal Time series:** This type of data observation occurs at specific timelines, like an increase in shopping sales during the festive season.
- **Stationary Time series:** A time series data in which statistical properties like mean and variance are constant over time. Stationarity is the common assumption in many time series techniques.
- **Non-Stationary Time series:** A type of data with properties that can change over time. Trends, cycles, or randomness can cause it.
- **Stochastic Time series:** Random factors typically influence this time series, incorporating random variables.

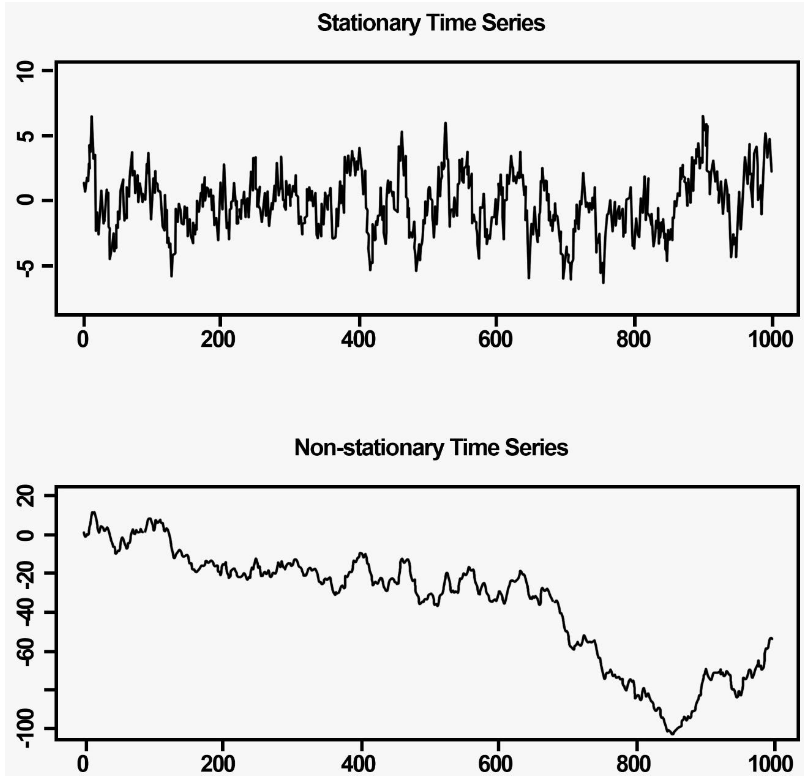

Figure 2: Example Image of a Stationary Time Series and a Non-Stationary Time Series [12]

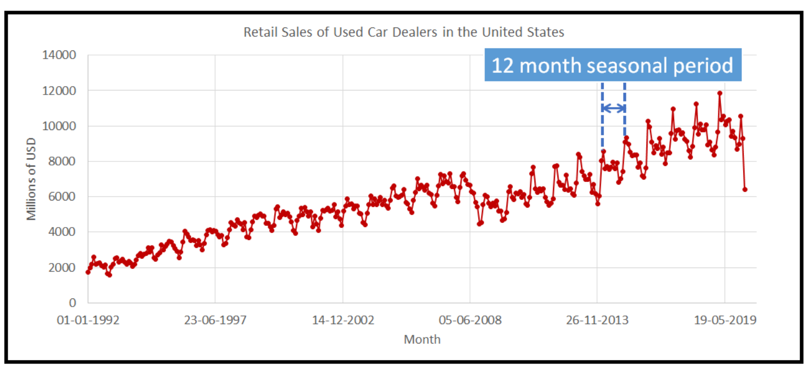

Figure 3: Example Image of a Seasonal Time Series Data [13]

2.2 Related Work

With the increase in the number of online websites over the past few years, there has been a significant increase in the analysis of historical data to forecast the future.

This is one of the critical research areas in web traffic prediction. It extends beyond the analysis by using the historical data to forecast the future.

Many models and techniques have been used in this field to forecast the future to anticipate trends or potential issues affecting a website's performance. Some of the standard models are ARIMA (Auto Regressive Integrated Moving Average), SARIMA (Seasonal Auto-Regressive Integrated Moving Average), Exponential Smoothing, Random Forest, Facebook's Prophet and Bayesian Structural Time series.

In this section, we provide a comprehensive overview of recent studies and research that have been done in web traffic forecasting in the field of machine learning. This research and studies investigate various techniques used in forecasting con, considering multiple factors like visitors for the web page, session data, traffic source, bounce rate, and device usage.

Three methods have been employed by Shuang et al. [9] to forecast the time series data. Three machine learning techniques have been used: Sequence to Sequence with Causal CNN (SeqtoSeq CNN), LSTM-based recurrent networks (LSTM), and KNN (K Nearest Neighbours). For the Wikipedia webpage data, they have employed web traffic time series. They have applied several pre-processing techniques to the data collection, such as normalizing the data to have a zero mean.

Following the machine learning procedures mentioned above, the model's performance was assessed using the Symmetric Mean Absolute Percentage Error (SMAPE). SeqtoSeq CNN outperformed KNN and LSTM in terms of SMAPE. Upon examining the graphs, it is evident that all three approaches struggled to forecast the sporadic spikes accurately. Hence, there is a scope to improve the results in the future.

Soheila et al.'s research paper [14] presents a novel technique for time series prediction. The ARIMA approach was extended to conduct this study. In the expanded approach, the data was divided, the training data was applied to the

ARIMA model technique, and the model's error was computed. The mean estimate error in the forecasting process was calculated for the model's error.

The research proposed by Indrajeet et al. [15] offers a simple data augmentation technique that can enhance such networks' performance greatly. Utilizing predictions from statistical models, the Augmented-Neural-Network approach can help unleash the potential of neural networks on intermediate length time-series and yield competitive outcomes. It demonstrates how data augmentation can assist in determining the optimal neural architecture for a specific time series when combined with Automated Machine Learning approaches like Neural Architecture Search. When these three neural network-based models are combined, the forecasting accuracy of a COVID-19 dataset is significantly improved. The maximum improvement in forecasting accuracy is 21.41% [15].

A specific kind of spiking neural network called a Polychronous Spiking Network was proposed by David Reidl et al. [9] and is used to predict financial time series. According to the research, spiking neural networks' innate temporal capabilities make them suitable for non-stationary input. Three other systems were used as benchmarks for their spiking neural network: a Multi-Layer Perceptron neural network, two "traditional," rate-encoded neural networks, a Dynamic Ridge Polynomial neural network, and a standard linear predictor coefficients model. In this comparison, three noisy and non-stationary time series were used: IBM stock data, US/Euro exchange rate data, and the price of Brent crude oil. The trials demonstrated that the Spiking Neural Network yielded superior prediction results when compared to the other algorithms.

Inspired by the research mentioned above, we tried to forecast the results of our study by adding a spiking neural network to the web traffic data set. Research is being done to determine whether spiking neural networks can predict random and non-stationary data values. The models' outcomes are contrasted with those of other machine learning techniques applied to the dataset, and standard metrics will be used for evaluation.

CHAPTER 3

Methodology

This chapter discusses the different methodologies that have been used in this research.

3.1 SARIMAX (Seasonal Auto-Regressive Integrated Moving Average with exogenous factors)

SARIMAX is one of the popular techniques used in time series forecasting. SARIMAX models are widely used in various fields, including economics, finance, and environmental sciences, for time series forecasting and analysis when there is seasonality and the influence of external factors [16]. SARIMAX can be broke down into ARIMA(Autoregressive Integrated Moving Average), AR(Auto Regressive), and MA(Moving Average).

3.1.1 Auto Regressive AR(p)

AR(p) here p describes the order of the model, i.e., it determines the value in the current time series based on how many previous values.
It performs regression of the time series values onto itself.

The following formula describes it:

$$Y_t = c + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \varepsilon_t,$$

where

- Y_t: Value of the time series at time t.
- c: Constant term.
- $\phi_1, \phi_2, \ldots, \phi_p$: Parameters of the model.
- ε_t: Error term, generally noise.

3.1.2 Moving Average MA(q)

A Simple Moving Average(MA(q)) is a common average of previous data points. Here, q refers to the order of the moving average. In a simple moving average, each point of the data is equally weighed.

$$MA(q) = \frac{1}{q} \sum_{i=n-q+1}^{n} P_i,$$

where

- $MA(q)$ is the moving average over q periods.
- P_i represents the price (or value) at period i.
- n is the current period.
- The summation $\sum_{i=n-q+1}^{n} P_i$ indicates the sum of the prices from the period $n - q + 1$ to the current period n.

3.1.3 Integrated Moving Average

IMA (Integrated Moving Average) generally refers to making the time series stationary; It is performed by differencing the data. The order of I relates to several times the data has been differenced.

3.1.4 ARIMA (Autoregressive Integrated Moving Average)

ARIMA is one of the popular approaches in time series forecasting.
It is developed by combining the techniques mentioned above autoregressive (AR), differencing (I – for Integrated), and moving average (MA) components.
ARIMA is generally represented by ARIMA(p,d,q), and formula is :

$$Y'_t = c + \phi_1 Y'_{t-1} + \phi_2 Y'_{t-2} + \ldots + \phi_p Y'_{t-p} + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \ldots + \theta_q \varepsilon_{t-q} + \varepsilon_t,$$

where:

- Y'_t is the differenced series, which may be the same as Y_t if $d = 0$.
- c is a constant (which may be zero).

- $\phi_1, \phi_2, \ldots, \phi_p$ are the coefficients for the AR terms.
- $\theta_1, \theta_2, \ldots, \theta_q$ are the coefficients for the MA terms.
- ε_t is the error term at time t.
- p is the number of AR terms.
- q is the number of MA terms.
- d is the differencing order required to make the time series stationary, and it determines how Y_t' is calculated from Y_t.

3.1.5 SARIMA-Seasonal ARIMA

The SARIMA model is similar to the ARIMA model, except for a slight difference. In the SARIMA model, we introduce additional autoregressive and moving average components.

3.1.6 SARIMAX(p,d,q)

Exogenous variables, or outside data, are incorporated into the forecast in the SARIMAX model, an expanded version of SARIMA and ARIMA. Due to its ability to incorporate the impact of external noise into the time series, this addition may prove beneficial for forecasting [17].

3.2 Popular Neural Network Models for Time-Series Data

3.2.1 Recurrent Neural Networks (RNNs)

The primary application of recurrent neural networks as a neural network design is the pattern recognition of sequence data. The standard data used are genome time series, generally industry-setting data [18].

They are called recurrent because they repeat the tasks for every element in the sequence, with the following output dependent on previous inputs. Some of the standard applications of RNNs are in the fields of Language Processing, Speech Recognition, Image Processing, etc [19].

The common difference between RNN and a general feed-forward network is how the information gets passed through the network. In a feed-forward network, the data is passed without any cycles, while the RNNs have cycles in their architecture that transmit the information to themselves. This cyclic nature in their architecture helps them consider previous inputs apart from current ones.

RNN's architecture depends upon the type of input you are trying to solve, i.e., from a singular input to many.

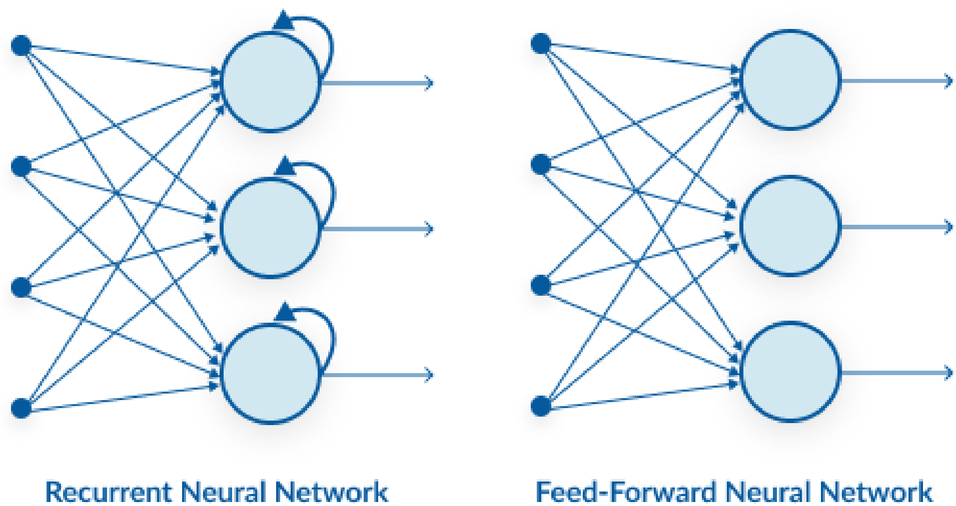

Figure 4: Architecture difference between Feed Forward neural network and Recurrent Neural Network [20]

The typical architecture of RNNs is generally similar to the feed-forward neural network. Each layer is responsible for processing and forwarding the information to the next layer. The common layers are:

1. Input Layer: Typically, data enters the RNN through the input layer. In general, it resembles the input layer of a simple neural network. It transfers the data to the hidden layer. Each neuron in the hidden layer represents a feature from the input data.

2. Hidden Layer: The recurrent loops occur in the hidden layer of RNN. Generally,

loop connections in RNNs allow them to use the memory even in an invisible layer of a feed-forward neural network, where data flow is only forward.

3. Weights: The transformation of input data to the hidden states is governed by the weights learned during the training

4. Backpropagation: Backpropagation is a core training method in Recurrent Neural Networks (RNNs), where the output from one step is used as the input for the next. Due to their sequential data processing, RNNs can efficiently capture temporal dynamics and dependencies in time-series data.

5. Output Layer: The output layer produces the network's final output. In RNN, the timestamp 't' output can be given as input at 't+1'.

6. Challenges: Sometimes RNNs are difficult to train because of mainly two problems: vanishing gradient and exploding gradient. This problem occurs whenever the gradients are either too small or too large. Long Short-Term Memory (LSTM) has been developed to overcome these challenges.

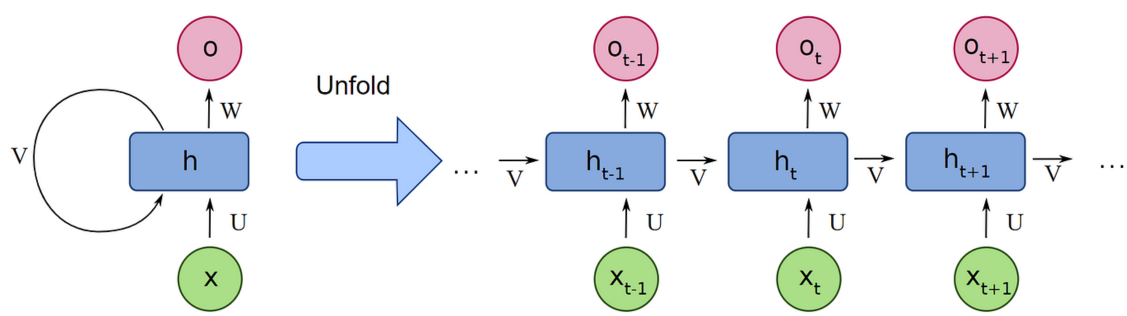

Figure 5: Unfolding a Recurrent Neural Network [21]

3.2.2 Long Short-Term Memory (LSTM)

One kind of recurrent neural network (RNN) architecture utilized in deep learning is the Long Short-Term Memory (LSTM) model. Texts or time series data can be recognized for patterns using LSTM [22]. Traditional RNNs have vanishing gradient and exploding gradient problems. Because of this problem, important information can be lost over time. LSTM are generally efficient in retaining the information for a more extended period. A vanilla LSTM unit can be described as:

1. Cell State: This is the central part of the LSTM unit, which serves as the network's "memory." The network may make predictions or choices based on the information it has "seen" several steps back in the sequence since it can carry relevant data throughout the processing of sequences.

2. Input Gate: The input gate regulates the amount of fresh data that enters the cell state. Based on the data from the previous output and the current input, it determines which values are changed.

3. Output Gate: By regulating the information flow from the cell state to the output of the LSTM unit, the output gate ascertains the subsequent concealed state.

4. Forget Gate: By controlling the memory of the cell state, the forget gate—a key component first proposed by Gers et al.—plays an essential part in Long Short-Term Memory (LSTM) networks. It works by removing data points that are considered unnecessary from the cell state. This enables the LSTM to dynamically modify its memory focus and "forget" unnecessary information, improving the model's performance and applicability in sequential data processing.

Together, the above-discussed elements control how information moves across the network. The LSTM may efficiently learn over extended periods without the risk of the gradient vanishing or exploding because the gates selectively determine whether information enters, exits, or is forgotten.

An LSTM's memory block, which consists of one or more LSTM units, is made to keep its state in time, giving the network a kind of internal memory that helps in processing data sequences.

Because of this, the LSTM can function effectively in tasks like speech recognition, natural language processing, and time series prediction, where interpreting present data requires a grasp of context provided by prior information.

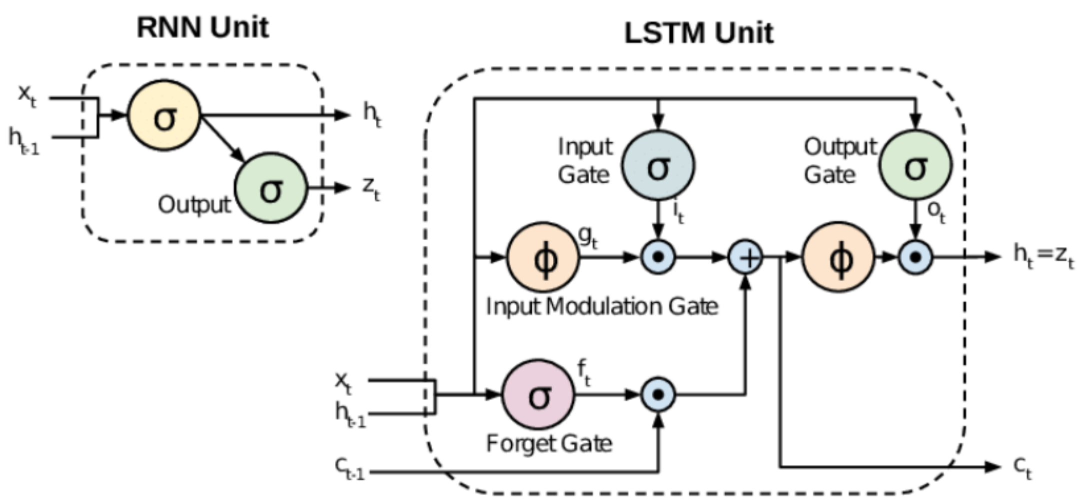

Figure 6: RNN and LSTM Architecture [23]

LSTMs are helpful in traffic prediction, which is a time series-based prediction. Some benefits are that web traffic is generally influenced by long-term dependency, and LSTMs can quickly adapt to the long-term dependencies because of the gates. They are better suited for the time series data [24]. The length of the web traffic sequence data can vary significantly. Unlike many machine learning models, LSTMs can handle these variations without needing fixed-size inputs. The length of the web traffic sequence data can differ considerably. Unlike many machine learning models, LSTMs can take these variations without needing fixed-size inputs.

3.3 Spiking Neural Networks

Spiking Neural Networks (SNNs) represent a significant advancement in neural network models, offering unique advantages, especially in applications where energy efficiency and real-time processing are critical. They serve as a bridge between neuroscience and machine learning, utilizing biologically realistic models of neurons for computation. This makes them compatible with temporal coding and capable of handling the sparsity observed in biological systems [8].

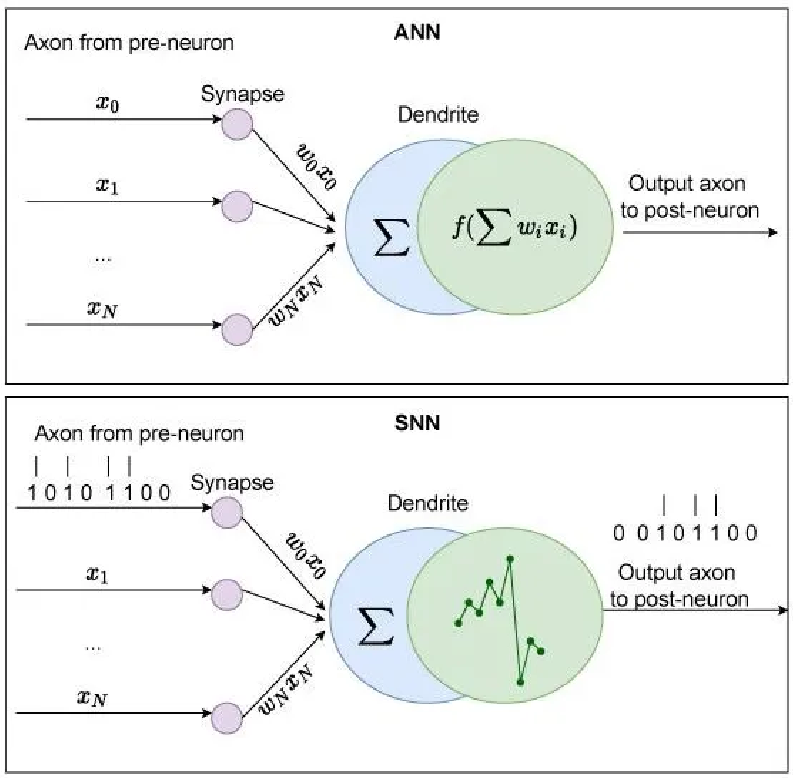

Figure 7: ANN and SNN Architecture [9]

SNNs differ from traditional firing rate models in neural networks by focusing on the timing of spikes and variations in sub-threshold membrane potentials. This aspect is crucial as it allows for the recording of activities of multiple cells, where the time difference between spikes in different neurons and the spike timing itself can have significant functional implications [25].

One of the challenges in SNN research is the development of robust training methods. Unlike Artificial Neural Networks (ANNs) that are typically trained using techniques like stochastic gradient descent and backpropagation, SNNs lack solidified training methods, a crucial area for future research and development. A critical step in utilizing SNNs is encoding actual data into spikes, a substantial part of creating SNNs. Spiking neural networks are a kind of neural network that mimics the natural neural network. In contrast to conventional artificial neural networks, which process data continuously, SNNs include time in their operational model.

CHAPTER 4

Results

4.1 Data set

The web traffic data set consists of a daily view of different Wikipedia articles from July 2015 to December 2016. This dataset serves as a fertile ground for statisticians and data scientists to test hypotheses, benchmark models, and explore the behavior of a diverse range of topics. Still, it also presents unique challenges due to its inherent non-stationarity. In this context, 'non-stationarity refers to the time series data's changing statistical properties over time, such as mean, variance, and autocorrelation. The analysis of fine temporal dynamics, such as day-of-week effects, holiday impacts, and other short-term patterns hidden in less granular data, is made possible by the granularity of the daily views.

The dataset, which spans over a year and a half and includes data from several seasons and events, offers a rich background for researching longitudinal effects and cyclic trends. A multifaceted study is made possible by adding metadata regarding the type of traffic (mobile, desktop, and spider). Researchers can study both the impact of web crawlers on traffic data and the trends in device usage on content consumption. It's crucial to remember that the non-stationary character of this dataset necessitates careful thought while selecting the analytical techniques. The dynamic patterns and trends present in this data may need to be better captured by models that assume stationarity.

Given that Wikipedia is among the most popular websites worldwide, the knowledge gained from this data set has practical applications. Wikipedia traffic patterns can guide content design, server load management, and even sociological studies of the public's interest in different subjects over time. The data set can be downloaded from the Kaggle website, but the user must accept the terms and conditions of the

Kaggle user agreement.

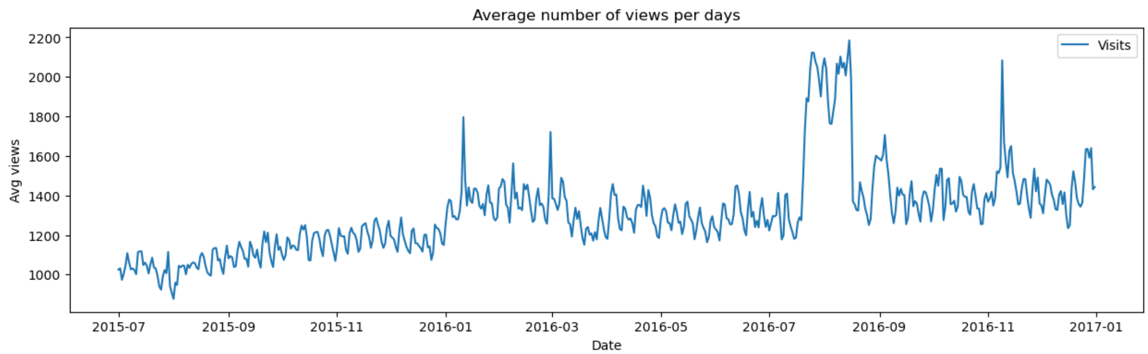

Figure 8: Web Traffic Dataset

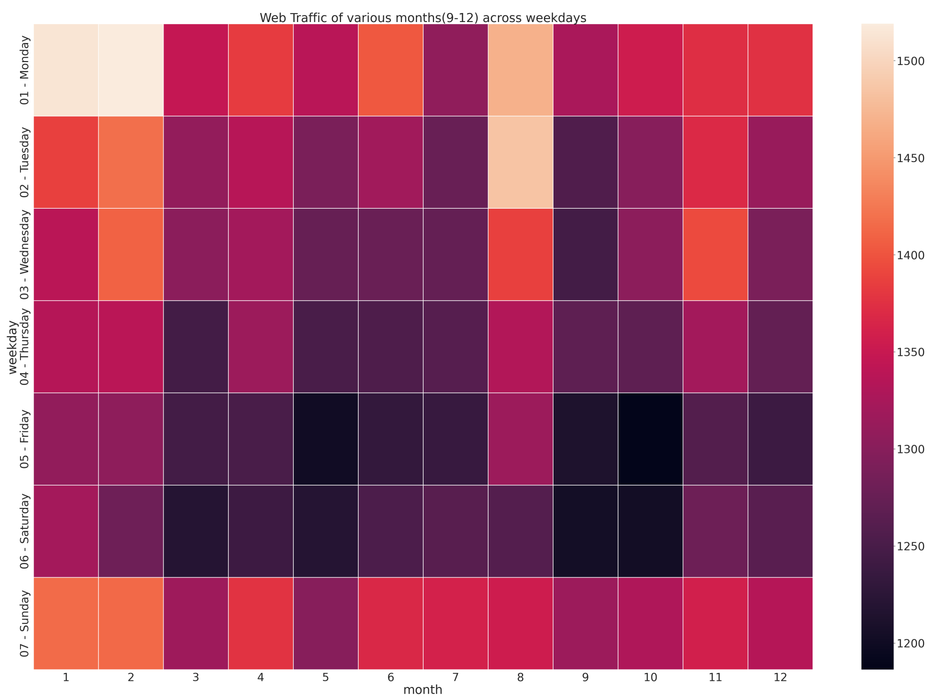

Figure 9: Temporal Data Characteristics

4.2 Data Pre-processing

Several pre-processing steps have been performed on the data set. The data set consists of null values, which are dropped as the data set size is enormous.The data set has been split into train and test sets after a number of pre-processing stages were finished; the test set is made up of the traffic from the last 180 days, and the training set is made up of the remaining data.

During the training step, the Sarimax model was fitted using a try-except block to find the optimal AR order (p) and the MA order(q), ranging from 0 to 6. The differencing order (d) is fixed at 1, suggesting an assumption of first-order differencing to render the time series stationary. The Bayesian Information Criterion (BIC) and Akaike Information Criterion (AIC) are calculated and saved if the model fits successfully. These factors are essential for choosing the suitable model since they indicate its quality and balance its complexity and goodness of fit, punishing adding extra elements.

After completing the iterations, the combinations of p and q and the related AIC and BIC values comprise the findings, organized into a data frame for additional examination. The model with the best balance between parsimony and fit quality is thus easily identified by sorting this data frame according to its AIC and BIC values. Using this method makes it possible to choose a suitable SARIMAX model for the time series data while taking into account the trade-off between the model's complexity and its capacity to reflect the underlying data patterns accurately.

During the training of LSTM, various mini batches were created, and it was found that a batch size of 16 yielded better results. The training epoch was set to 200, and the RMSprop optimizer, which is an optimization algorithm used explicitly in handling the vanishing gradients or exploding gradients problem, was used in the model. Hence, this makes a suitable choice. During the model building of the SNN model, various tuning steps were performed to find the best model by changing the number of hidden layers we created in the network and specifying the number of neurons we used. In Spiking Neural Networks (SNNs), the Leaky Integrate-and-Fire (LIF) model is frequently used for neurons. It's an abridged depiction of how actual brain neurons function. The LIF model captures the fundamental principle that neurons integrate incoming data and produce a spike (or fire) when a threshold is achieved.

Table 1: SARIMAX Model Hyperparameters

Hyperparameter	**Tested Values**
AR order (p)	0 to 6
MA order (q)	0 to 6
Differencing order (d)	1
Criteria	AIC, BIC

Table 2: LSTM Model Hyperparameters

Hyperparameter	**Value**
Number of LSTM Units	256
Loss Function	Mean Squared Error
Optimizer	RMSprop
Epochs	200
Batch Size	16

Table 3: SNN Model Hyperparameters

Hyperparameter	**Tested Values**
Hidden layers	1-3 layers
Neurons per layer	50-200 neurons
Neuron model	Leaky Integrate-and-Fire (LIF)
Learning rate	0.01, 0.001
Regularization	L1, L2

4.2.1 Rolling moving average

A rolling or moving average in time series data highlights longer-term patterns and smooths out short-term variations. The process involves calculating the mean of any given collection of data. The average of the most recent 'N' data points is considered at each stage in our scenario.

$$MA(t) = \frac{1}{k} \sum_{i=t-k+1}^{t} x_i,$$

where MA(t) represents the moving average at time t, k is the window size, and x_i is a data point.

After calculating the rolling average, a threshold determines the binary result (0 or 1). If the rolling average exceeds the threshold, the result is 1; if not, it is 0. This approach can identify notable increases or decreases in web traffic. This method has many academic applications, like signal processing, data analysis, and other domains.

Binary values are created after performing the rolling average on the given data and using the threshold. We trained the models and conducted several tests to assess the models.

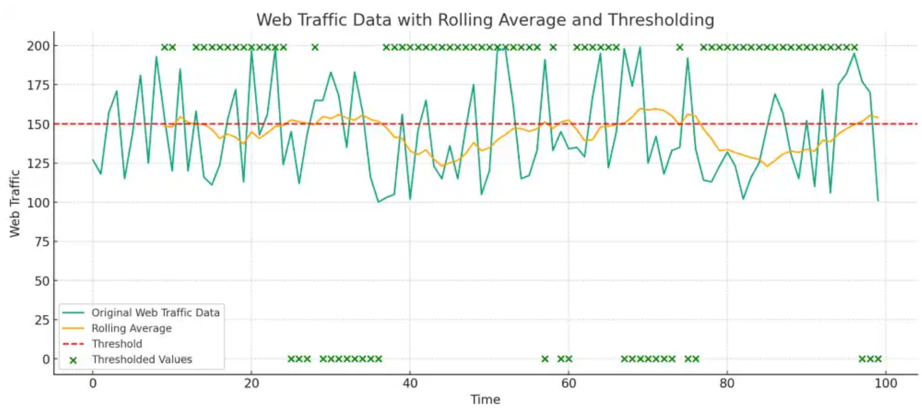

Figure 10: Web Traffic Data with Rolling Average and Thresholding [26]

4.3 Evaluation Metrics

4.3.1 Root Mean Square Error (RMSE)

RMSE is a commonly used metric to assess a model's ability to predict quantitative data. It is mostly utilized in machine learning prediction models, engineering, and statistical analysis.

The square root of the mean square of all mistakes is known as the root mean square error, or RMSE. The difference between the actual and model-predicted values is how the error is computed.

The RMSE (Root Mean Square Error) is calculated as follows:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{1}$$

- y_i represents the actual values.
- $\hat{y}_i$ represents the predicted values.
- n is the number of observations.
- The squared difference between the actual and anticipated values is expressed as $(y_i - \hat{y}_i)^2$.
- The sum of these squared differences is averaged over n observations, and the square root of this average is taken to compute RMSE.

RMSE is widely used in various industries, like finance, for forecasting models, stock market prediction, and the energy sector to measure the accuracy of energy demand, climate science, logistics, and sales. A lower RMSE value indicates a better model than a higher RMSE value.

4.3.2 Mean Absolute Error (MAE)

The mean absolute error counts the average number of errors in a set of forecasts without taking into account the direction of the errors. The average absolute difference between the expected values and the observed data is used to calculate it.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right|, \tag{2}$$

where

- n is the number of observations.
- y_i is the actual value of the ith observation.
- $\hat{y}_i$ is the predicted value of the ith observation.

- $|y_i - \hat{y}_i|$ is the absolute difference between the actual and predicted values for each observation.

Learning issues can be easily transformed into optimization problems by using the widely used MAE loss function in regression situations. It also serves as a straightforward, quantitative method of calculating mistakes in regression scenarios.

4.4 Results

4.4.1 Experiments with Raw Web Traffic Data

We trained the models using various hyperparameter settings and conducted several tests to assess the models. We have completed many tests to determine the ideal hyperparameters and, thus, the optimal solution for every model. To benchmark the evaluation, we have contrasted the suggested models with a few currently used ones.

Table 4: Results: Metrics Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) are calculated for traffic forecasting

Model	RMSE	MAE
ARMA	2.81	2.1
SARIMAX	2.21	1.70
LSTM	2.19	1.8

Table 4 shows the model results obtained by training the model and evaluating using the respective metrics for prediction. We can see that LSTM performed better in predicting traffic when compared to all the other models in terms of RMSE and MAE values. However, the main issue with ARIMA, ARIMAX, and LSTM models is that they need to predict the random or seasonal spikes in the data, which was inspected using the prediction graphs below. Figures 11-13 show that the models were able to capture the general trend but failed to predict the spikes in the data.

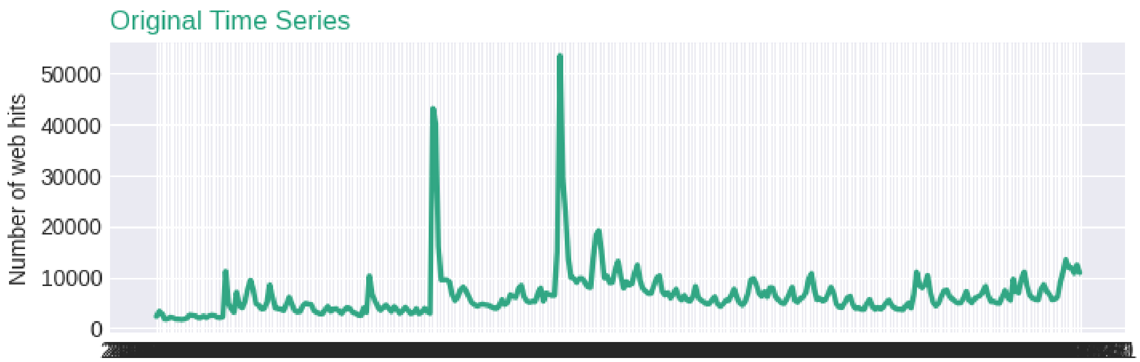

Figure 11: Original Time Series Data (Web Traffic)

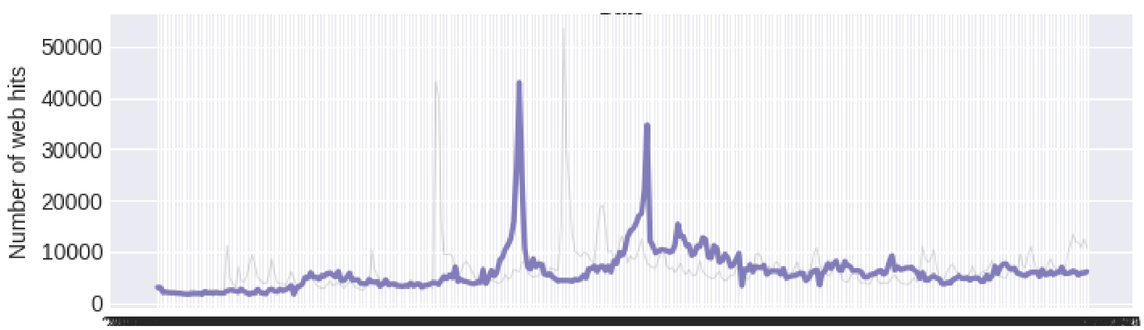

Figure 12: SARIMAX model generated prediction

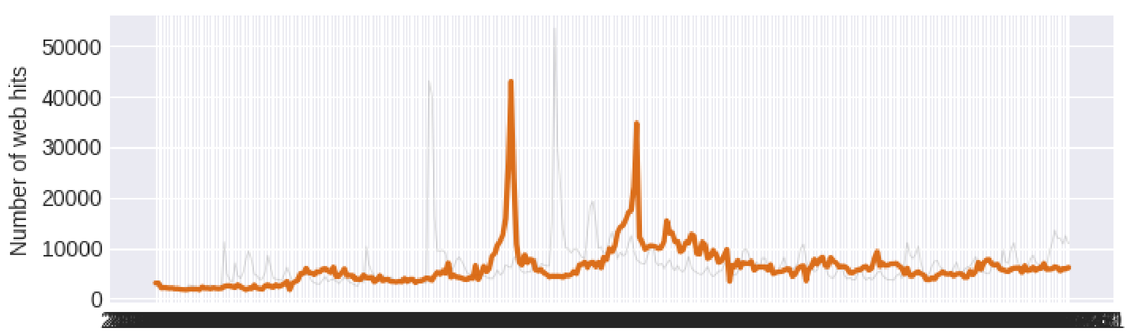

Figure 13: LSTM model generated prediction

4.4.2 Experiments with Smoothed Web Traffic Data

Next, we conducted another experiment to confirm the SNN capability to capture the significant spikes. We preprocessed the dataset by the rolling moving average smoothing with a threshold, which was discussed earlier. This preprocessing reduced

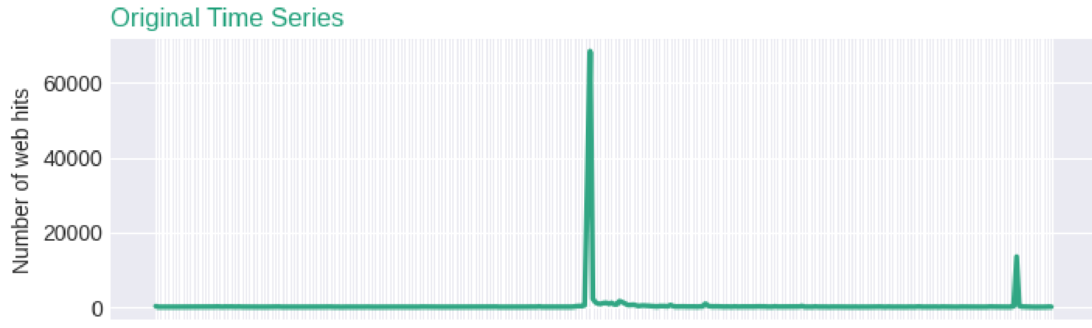

Figure 14: Original Time Series Data (Smoothed)

Table 5: Results with Smoothed Dataset: Metrics Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) are calculated for traffic forecasting

Model	RMSE	MAE
SARIMAX	0.38 ±0.02	0.27 ±0.02
LSTM	0.31 ±0.04	0.19 ±0.04
SNN	0.267 ±0.06	0.124 ±0.02

noise and minor fluctuations and highlighted the remarkable spikes in the dataset. The smoothed data is shown in Figure 14. The reduction of minor fluctuations can be observed in the figure when it is compared to the original dataset in Figure 11. After training the models, we tested the models by forecasting the future traffic for the next 180 days.

Table 5 shows that Spiking Neural Networks (SNNs) performed better than alternative models in predicting the significant spikes in the smoothed dataset. This can be confirmed by comparing the predicted traffic curves by all three methods illustrated in Figures 15-17.

Figures 18–21 show that the Spiking Neural Network performed better than other models even in data predicting scenarios with many spikes. The prediction graphs for the SNN, LSTM, and SARIMAX models are shown in these figures, which clearly show how the SNN model's predictions were than the others.

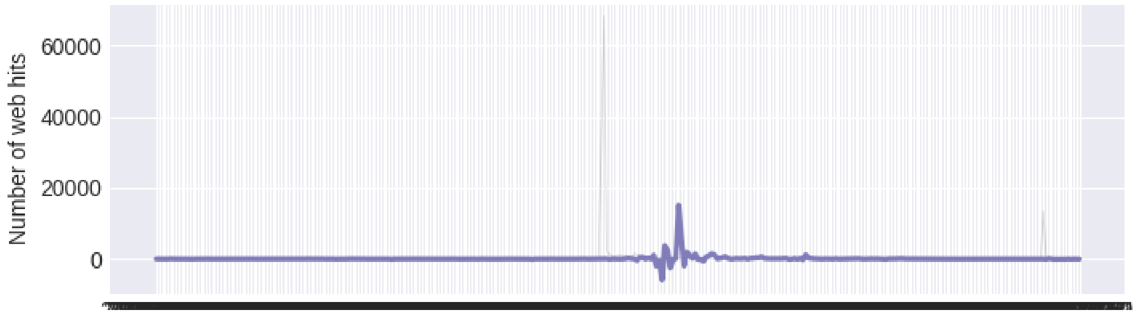

Figure 15: SARIMAX model generated prediction

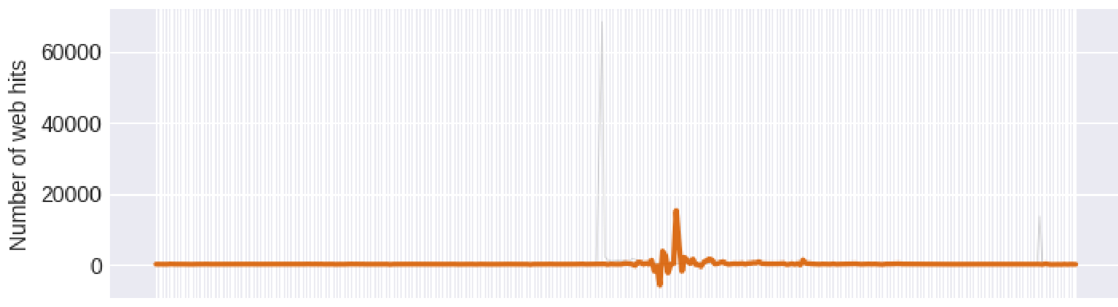

Figure 16: LSTM model generated prediction

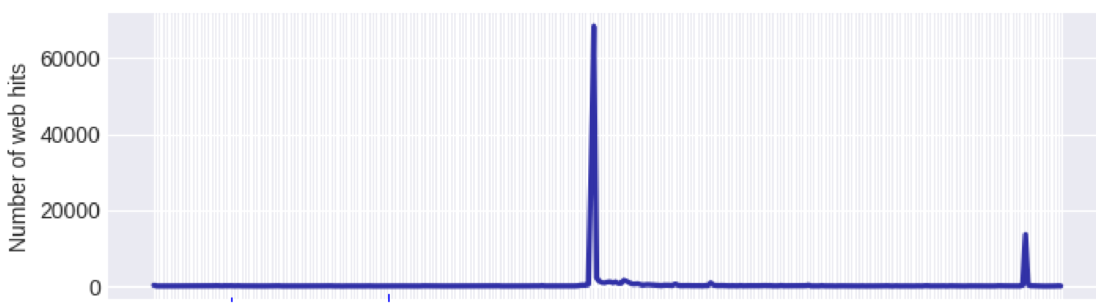

Figure 17: SNN model generated prediction

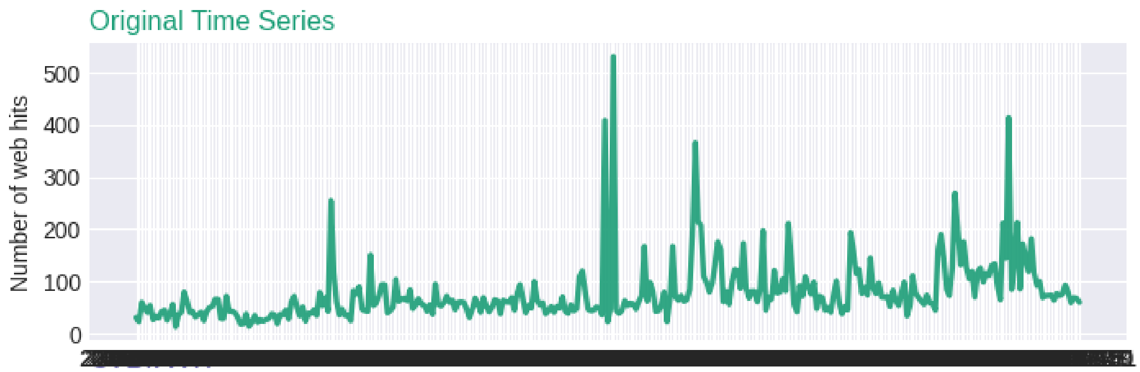

Figure 18: Original Time Series Data

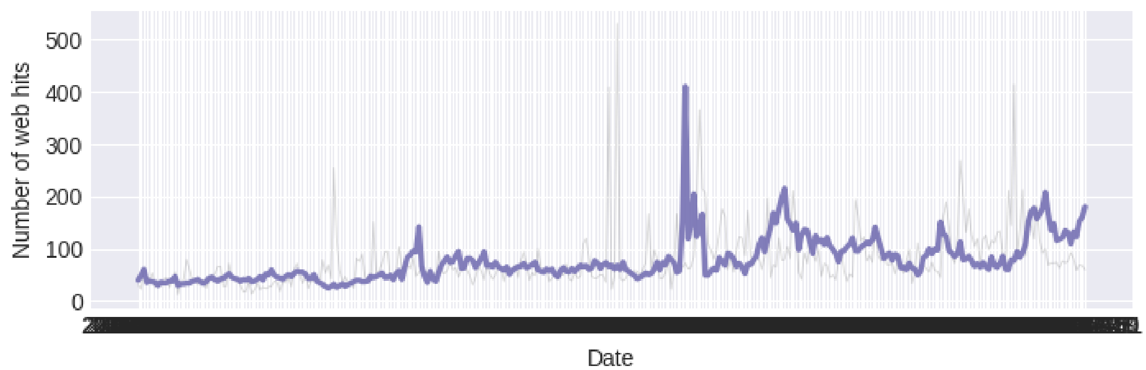

Figure 19: SARIMAX model generated prediction

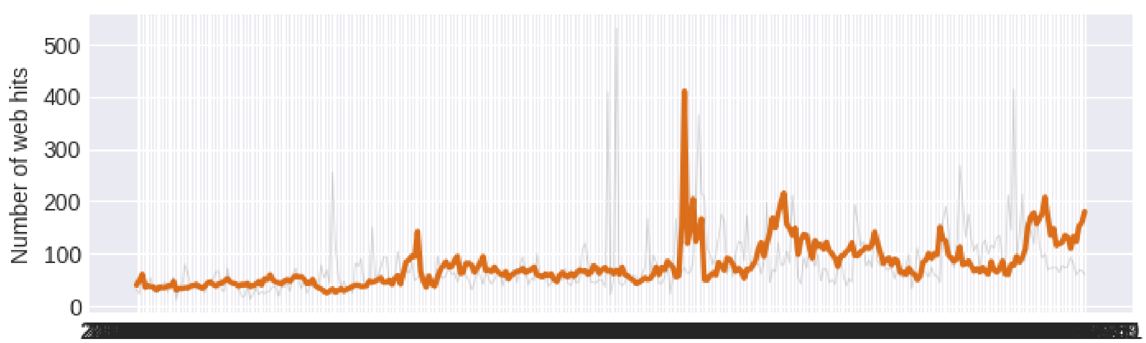

Figure 20: LSTM model generated prediction

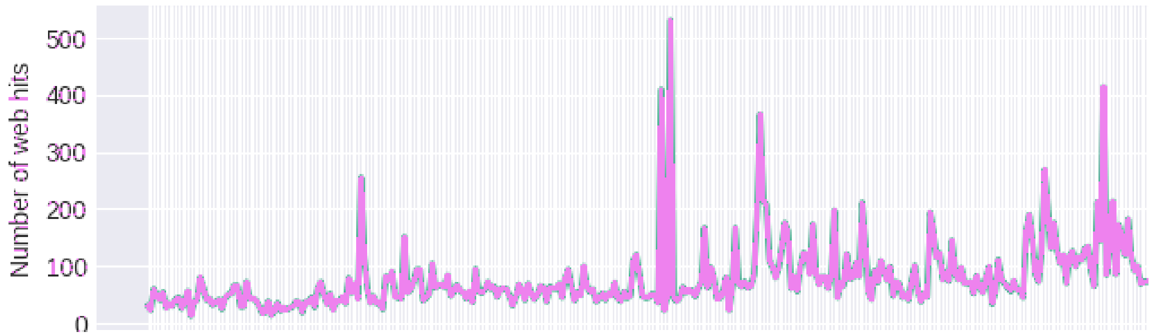

Figure 21: SNN model generated prediction

4.4.3 Experiments with Raw Web Traffic Data with More Spikes

Furthermore, we conducted another experiment to understand the capability of SNNs specifically with the spikes in a raw dataset with minor fluctuations. We extracted a subset of data points where a higher frequency of spikes and variance were observed.

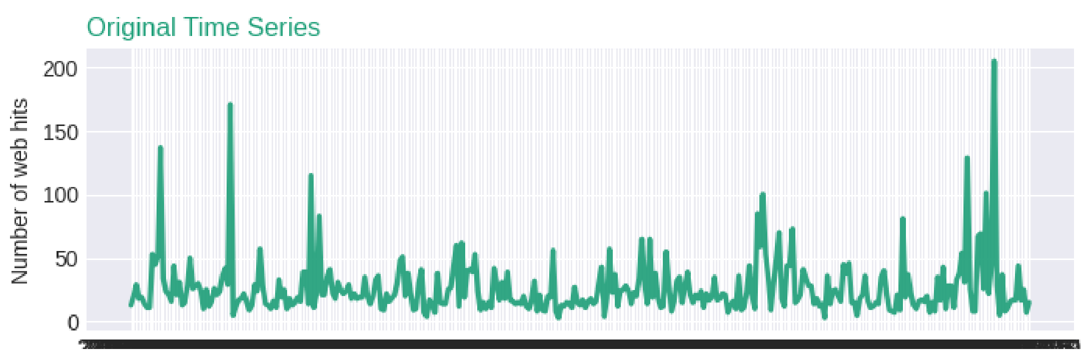

Figure 22: Graphical representation of subdataset selected for the forecasting task of frequent spikes

Figure 22 shows us the frequency and intensity of the spike occurrences and clarifies the increased variance in this subset. This graphical representation justifies our attention to this dataset segment by showing the data distribution in the selected. After selecting the subset, we tried to forecast the data using the above-discussed methods and got the following results.

Table 6: Results with a Subdataset with Frequent Spikes: Metrics Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) are calculated for traffic forecasting on subset data

Model	RMSE	MAE
SARIMAX	3.9±0.2	2.6 ±0.15
LSTM	3.4 ±0.21	2.2 ±0.23
SNN	2.53 ±0.12	1.8 ±0.2

As summarized in Table 6, the SSN model outperforms conventional techniques in forecasting data with notable spikes and variance. The SNN model's better performance in this situation emphasizes the benefits of sophisticated neural network architectures in handling scenarios with complicated and high levels of variation in the data, highlighting its promise as a reliable tool for predictive analytics across various industries.

CHAPTER 5

Conclusion and Future Work

We conducted various experiments on the web traffic data set in this work. The data set is a time series data set. We built different models using several hyperparameter tunings and evaluated the model's performance in forecasting future traffic. We have seen that the LSTM model performed better when compared to their models like Sarimax, Arima, and SNN when evaluating its forecasting in the future to the metrics RMSE and MAE. But even though LSTM had good results, it failed to forecast the seasonal spikes in the trend. To test this, we converted our data into signal form using the moving average threshold technique, as discussed earlier.

Again, when we tried to forecast the data using the modified data, we saw that spiking neural networks performed better than LSTM, Sarimax, and Arima. This is due to the inbuilt nature of understanding and forecasting seasonal trends, which requires the ability to interpret temporal dynamics, a skill that SNN excels at. Their ability to analyze time-based data efficiently makes them appropriate for jobs requiring an understanding of how patterns change over time

When compared to conventional neural networks, SNN usually requires less energy. This results from their spike-based communication, which only turns on specific network segments when required. This efficiency can be quite helpful when working with a huge dataset frequently encountered in seasonal forecasting.

As a researcher deeply interested in neural networks, I see a significant opportunity in merging the unique strengths of Spiking Neural Networks (SNN) and Long Short-Term Memory (LSTM) networks. This hybrid approach could revolutionize forecasting complex patterns, especially in areas like traffic management.

My observation has been that while SNNs excel in processing data with temporal dynamics, such as seasonal trends or signal forms, they might only sometimes be optimal for handling more standard data formats. On the other hand, LSTM shows remarkable proficiency in dealing with regular data sequences but needs help with the intricacies of forecasting seasonal spikes or trends.

The future work lies in developing a model that combines the best of both worlds. This hybrid model would utilize the energy efficiency and real-time processing capability of SNNs alongside the sequential data handling and pattern recognition strengths of LSTM. The goal is to create a system that accurately forecasts traffic flow and adeptly manages the unpredictable nature of seasonal spikes.

LIST OF REFERENCES

[1] Z. Liu, Z. Zhu, J. Gao, and C. Xu, ‘‘Forecast methods for time series data: A survey,’’ *IEEE Access*, vol. 9, pp. 91 896--91 912, 2021.

[2] Z. Han, J. Zhao, H. Leung, K. F. Ma, and W. Wang, ‘‘A review of deep learning models for time series prediction,’’ *IEEE Sensors Journal*, vol. 21, no. 6, pp. 7833--7848, 2021.

[3] A. Horelu, C. Leordeanu, E. Apostol, D. Huru, M. Mocanu, and V. Cristea, ‘‘Forecasting techniques for time series from sensor data,’’ in *2015 17th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, 2015, pp. 261--264.

[4] O. Kelany, S. Aly, and M. A. Ismail, ‘‘Deep learning model for financial time series prediction,’’ in *2020 14th International Conference on Innovations in Information Technology (IIT)*, 2020, pp. 120--125.

[5] D. Sikka and C. Vinoth Kumar, ‘‘Website traffic time series forecasting using regression machine learning,’’ in *2023 IEEE 12th International Conference on Communication Systems and Network Technologies (CSNT)*, 2023, pp. 246--250.

[6] T. Dunning. ‘‘Time series data.’’ [Online]. Available: https://www.scylladb.com/glossary/time-series-data/

[7] L. Li, X. Su, Y. Zhang, Y. Lin, and Z. Li, ‘‘Trend modeling for traffic time series analysis: An integrated study,’’ *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 6, pp. 3430--3439, 2015.

[8] M. Yao, G. Zhao, H. Zhang, Y. Hu, L. Deng, Y. Tian, B. Xu, and G. Li, ‘‘Attention spiking neural networks,’’ *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 8, pp. 9393--9410, 2023.

[9] D. Reid, A. J. Hussain, and H. Tawfik, ‘‘Financial time series prediction using spiking neural networks,’’ *PLoS ONE*, vol. 9, 2014.

[10] R. Casado-Vara, Á. M. del Rey, D. Pérez-Palau, L. de-la Fuente-Valentín, and J. M. Corchado, ‘‘Web traffic time series forecasting using lstm neural networks with distributed asynchronous training,’’ 2021.

[11] K. Anshu, S. K. Singh, and R. Kumari, ‘‘A machine learning model for effective consumer behaviour prediction,’’ in *2021 5th International Conference on Information Systems and Computer Networks (ISCON)*, 2021, pp. 1--5.

[12] S. P. Affek. ‘‘Stationarity in time series analysis.’’ 2019. [Online]. Available: https://towardsdatascience.com/stationarity-in-time-series-analysis-90c94f27322

[13] S. Date. ‘‘How to isolate trend, seasonality and noise from a time series.’’ 2019. [Online]. Available: https://timeseriesreasoning.com/contents/time-series-decomposition/

[14] S. Mehrmolaei and M. R. Keyvanpour, ‘‘Time series forecasting using improved arima,’’ pp. 92--97, 2016.

[15] I. Y. Javeri, M. Toutiaee, I. B. Arpinar, T. W. Miller, and J. A. Miller, ‘‘Improving neural networks for time series forecasting using data augmentation and automl,’’ *CoRR*, vol. abs/2103.01992, 2021. [Online]. Available: https://arxiv.org/abs/2103.01992

[16] C. Vanlalchhuanawmi and S. Deb, ‘‘Solar photovoltaic generation forecasting using lstm and sarimax model,’’ in *2023 IEEE 2nd International Conference on Industrial Electronics: Developments Applications (ICIDeA)*, 2023, pp. 286--291.

[17] S. I. Vagropoulos, G. I. Chouliaras, E. G. Kardakos, C. K. Simoglou, and A. G. Bakirtzis, ‘‘Comparison of sarimax, sarima, modified sarima and ann-based models for short-term pv generation forecasting,’’ in *2016 IEEE International Energy Conference (ENERGYCON)*, 2016, pp. 1--6.

[18] Y. Yu, X. Si, C. Hu, and J. Zhang, ‘‘A review of recurrent neural networks: Lstm cells and network architectures,’’ *Neural Computation*, vol. 31, no. 7, pp. 1235--1270, 2019.

[19] J. Oruh, S. Viriri, and A. Adegun, ‘‘Long short-term memory recurrent neural network for automatic speech recognition,’’ *IEEE Access*, vol. 10, pp. 30 069--30 079, 2022.

[20] A. Pai. ‘‘Analyzing types of neural networks in deep learning.’’ 2020. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/02/cnn-vs-rnn-vs-mlp-analyzing-3-types-of-neural-networks-in-deep-learning/

[21] P. Borges. ‘‘Deep learning: Recurrent neural networks.’’ 2018. [Online]. Available: https://medium.com/deeplearningbrasilia/deep-learning-recurrent-neural-networks-f9482a24d010/

[22] Y. Hua, Z. Zhao, R. Li, X. Chen, Z. Liu, and H. Zhang, ‘‘Deep learning with long short-term memory for time series prediction,’’ *IEEE Communications Magazine*, vol. 57, no. 6, pp. 114--119, 2019.

[23] A. Tripathi. ‘‘Data science duniya.’’ 2021. [Online]. Available: https://ashutoshtripathi.com/2021/07/02/what-is-the-main-difference-between-rnn-and-lstm-nlp-rnn-vs-lstm/

[24] X. Li, L. Liang, and B. Yu, ''Branching time series prediction method based on cnn and lstm,'' in *2023 IEEE 7th Information Technology and Mechatronics Engineering Conference (ITOEC)*, vol. 7, 2023, pp. 851--856.

[25] J. He, Y. Li, Y. Liu, J. Chen, C. Wang, R. Song, and Y. Li, ''The development of spiking neural network: A review,'' in *2022 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2022, pp. 385--390.

[26] Z. Tong. ''Staying in control with moving averages.'' 2016. [Online]. Available: https://www.elastic.co/blog/staying-in-control-with-moving-averages-part-1

APPENDIX A

Applicability to Other Types of Time-Series Data

Spiking Neural Network (SNN) model that we discussed earlier has been tested against SARIMAX (Seasonal AutoRegressive Integrated Moving Average with exogenous variables) and LSTM (Long Short-Term Memory) by benchmarking their results in predicting another type of time-series dataset. The details of this dataset are discussed below:

A.1 Data Set

To evaluate the three models, we will utilize stock exchange data from the Indian financial services company Bajaj Finserv Ltd. The dataset is available from 2008 to the end of 2021. It includes the daily turnover and total volume of traded stocks in addition to the stock price's mean, low, and high values. After agreeing to the terms and conditions, the data set is available to download in Kaggle. A sample of the data is shown in the following figure.

```
1 df.head()
```

Date	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deliverble
2008-05-26	2008-05-26	BAJAJFINSV	EQ	2101.05	600.00	619.00	501.0	505.1	509.10	548.85	3145446	1.726368e+14	NaN	908264	0.2888
2008-05-27	2008-05-27	BAJAJFINSV	EQ	509.10	505.00	610.95	491.1	564.0	554.65	572.15	4349144	2.488370e+14	NaN	677627	0.1558
2008-05-28	2008-05-28	BAJAJFINSV	EQ	554.65	564.00	665.60	564.0	643.0	640.95	618.37	4588759	2.837530e+14	NaN	774895	0.1689
2008-05-29	2008-05-29	BAJAJFINSV	EQ	640.95	656.65	703.00	608.0	634.5	632.40	659.60	4522302	2.982921e+14	NaN	1006161	0.2225
2008-05-30	2008-05-30	BAJAJFINSV	EQ	632.40	642.40	668.00	588.3	647.0	644.00	636.41	3057669	1.945929e+14	NaN	462832	0.1514

Figure A.23: Bajaj Finserv Data

We aim to forecast the Volume Weighted Average Price (VWAP) variable after each day. We separated the time series into train and test time series for the evaluation, with the training series using data up to the end of 2018. After performing the necessary pre-processing steps, we trained the models mentioned above using the training dataset discussed earlier.

Table A.7: SARIMAX Model Hyperparameters

Hyperparameter	**Tested Values**
AR order (p)	0 to 2
MA order (q)	0 to 2
Differencing order (d)	1
Criteria	AIC, BIC

Table A.8: LSTM Model Hyperparameters

Hyperparameter	**Value**
Number of LSTM Units	90
Loss Function	Mean Squared Error
Optimizer	Adam
Epochs	30
Batch Size	28

Table A.9: SNN Model Hyperparameters

Hyperparameter	**Tested Values**
Hidden layers	1-3 layers
Neurons per layer	50-200 neurons
Neuron model	Leaky Integrate-and-Fire (LIF)
Learning rate	0.01, 0.001
Regularization	L1, L2

The above tables provide information of various hyperparameter settings of the models SNN, LSTM, and SARIMAX.

A.2 Results

To evaluate the effectiveness of the models in predicting future stock prices, we assessed their performance using the Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE), as these metrics are most suitable for this type of data. After training the models using the previously discussed hyperparameters, we attempted to predict stock prices using the test dataset. To benchmark the results, we compared the performance of the SNN model with commonly used time series models such as SARIMAX and LSTM.

Table A.10: Results: Metrics Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) are calculated for stock forecasting

Model	RMSE	MAE
SARIMAX	224	160
LSTM	370	190
SNN	234	162

The tabular results presented above show that the SARIMAX model outperformed the other two models, while the difference in results between SARIMAX and SNN is minor.