# Demand Forecasting by use of Long-Short Term Memory (LSTM) Architecture

Richard Hoehn

Middle Tennessee State University
CSCI 7850
Prof. Dr. Joshua L. Phillips

December 10, 2023

***Abstract*—This paper examines the use of Long-Short Term Memory (LSTM), a deep learning recurrent neural network architecture, to predict real-world sales demand. Analyzing five years of daily sales data from ten stores, we prepared the dataset into a univariate temporal single-day format and normalized it using MinMax-Scalers for LSTM training. To mitigate real-world noise, a 7-day rolling average was applied to the daily sales.**

**Predictions were based on the LSTM model which is frequently used to analyze and predict time-series (temporal) regression problems. The trained LSTM model deployed worked well using a 90-day sequence(s) for training and validation. Since this problem is of a regression type the loss is calculated by use of the Mean-Squared Error (MSE) function with up to 150 epochs during training.**

**The results, presented in this paper through various plots, demonstrate that the LSTM's predictions closely match actual sales trends, indicating its effectiveness in learning demand patterns and forecasting future sales with a degree of accuracy. Since the predicted values follow the true values, one can infer that the model has good and reliable performance.**

**The paper concludes with the hypothesis that focusing on individual store data might enhance the model's ability to predict store-specific demand, suggesting a direction for further research.**

***Keywords*—LSTM, RNN, Demand Forecast, Deep Learning, Temporal Data, Sales Demand**

## 1. Introduction

In this paper we focused the student learning on demand forecasting, which is the process of predicting future customer demand for products or services using a variety of historical data-points based on a temporal data[1] lineage.

By employing deep learning recurrent neural networks (RNNs) in specific the Long-Short Term Memory (LSTM) architecture the research goal was to provide a model to predict sales demand based on historical daily sales data. The LSTM model is particularly good at capturing complex temporal dependencies over long ranges of time and patterns in historical sales data[1, 4, 6] making it[2] a very good option for demand forecasting tasks.

The motivation for the this paper is two fold:

**Firstly**, a time-series problem is very interesting since we often question how an individual or business can predict the future based on past data; additionally how the future might change if the input of said data is changed as a type scenario forecasting exercise.

**Secondly**, a better understanding of RNNs in specific the LSTM architectures will enhance the student's understanding of this deep learning field and prepare readers for future research in Computation & Data Sciences with regards to forecasting model designs.

In order to complete this research, data from Kaggle's Demand Forecasting challenge[3] released in 2018 was used. The dataset is comprised of five (5) years of store sales data. Each data point is based on a single day; resulting in the dataset being 912,500 daily sales data points.

The key aims for this project are:

- Deploy the dataset in the cloud (AWS) for remote extraction, process, and scrubbing that can then be used for training, validation, testing by the use of LSTM architectures.
- Build, Train, and Validate the LSTM model with the pre-processed Kaggle[3] dataset.
- *and finally*, using multiple potting techniques provide a visual approach to reviewing the performance of the LSTM model for prediction on the Kaggle[3] dataset.

[1] In this project the temporal was daily sales

[2] The LSTM Architecture

## 2. Background

LSTM models have emerged as a ”go-to” tool[1, 4] in the field of demand forecasting. Based on a recurrent neural network’s (RNN) architecture, LSTM models are frequently used[1] to analyze and make predictions based on time-series data, which is a common characteristic in demand forecasting scenarios.

Unlike traditional RNN architectures the LSTM design[5] (Fig. 1) can regulate the forward and backwards flow of information and more importantly be able to forget non-valuable data-traits. This means that LSTMs can remember pertinent information and utilize patterns from longer[4] historical data, which is an important feature for predicting future sales demand trends that may not be linear in nature[7] or have seasonal traits that need to be captured.

In many of the papers reviewed[1, 4] that used the LSTM architecture on demand forecasts, a tabular / numerical approach to validate the loss and accuracy of their models was used; however this paper uses visual plotting techniques to better gauge the perceived accuracy of these predictive regression problems.

## 3. Methods

In this section the mathematical, theoretical, and practical execution of this paper is introduced and references to the technologies, datasets, and visualization techniques presented.

### 3.1. Data Preparations

The dataset[3] is comprised of five (5) years of store sales data based on 50 different items at 10 different stores. Each data point is a single day’s data-point (rows) with this being said the dataset is comprised of 912,500 rows in a `csv` file.

**Dataset Details**

| | | | |
|---|---|---|---|
| Years | 5 | Days | 1,826 |
| Stores | 10 | Items per Store | 50 |
| Datapoints | 912,500 | File Type | csv |
| Min ($USD) | 11,709 | Max ($USD) | 44,936 |
| Mean | 912 | Std. | 527 |

TABLE 1: Kaggle Demand Forecast Details

With the data being a simple tab-based row-column structure we can infer the daily total sales for all stores can be represented as $s$, where each element $s_i$ is the sum of sales across all stores for day $i$.

$$s_i = \sum_{j=1}^{n} a_{ij} \tag{1}$$

In this Equation 1 the following is given:

- $i$ index for days
- $j$ index for stores
- $aij$ total sales of store $j$ on day $i$
- $si$ total sales across all stores on day $i$

### 3.2. Smoothing & Normalization of Dataset

Based on the description from Kaggle[3] of the data it’s evident that it is derived from real-world sources, suggesting it’s inherent noise. It is also stated in Kaggle’s information that the sales data is in whole US Dollars ($USD), which means that the values are rather large and are whole (real) numbers in nature. This leads to the need for smoothing and normalization parsing of the data.

**3.2.1. Smoothing.** In order to enhance the demand forecasting capabilities of the model, we used a common smoothing algorithm[7] to help the model learn more easily without the daily noise. A good methodical approach was to use a 7day moving average approach.

$$\text{MA}_i = \frac{1}{7} \sum_{k=i-6}^{i} s_k \tag{2}$$

In Equation 2 the details of the moving average are as follows:

- $\text{MA}_i$ is the 7-day moving average on day $i$
- $sk$ represents the sales on day $k$
- The summation $\sum_{k=i-6}^{i} s_k$ calculates the total sales over the 7-day period ending on day $i$ (from day $i - 6$ to day $i$)
- The factor $\frac{1}{7}$ averages the total sales over these 7 days

**3.2.2. Normalization.** Normalization is an important preparation step when preparing sales data for training an LSTM model[7]. This process involves scaling the sales data which is in large whole $USDs[3] so that it falls within a specific range, typically between $-1 < x < 1$. The purpose of normalization is to ensure that all sales contribute equally to the learning process, preventing some sales data with larger swings (magnitudes) from overpowering[7] the model’s learning process.

The process we chose for this project was using the `MinMaxScaler()`[4], where the minimum value of the

[3] See Table 1’s Min, Max, & Std. details

[4] Python: `from sklearn.preprocessing import MinMaxScaler`

data becomes -1 and the maximum value becomes 1. A formulaic representation of this type scaler can be seen in Equation 3 for details.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{3}$$

It should be noted that the pre-processing step this research applied is the same as many of the reviewed articles[2, 7] for this paper. Further, many of the reviewed papers[2] also stated the importance of normalization when training across multiple related time series as the ranges of values of sales demand based projects differ significantly between the temporal series.

### 3.3. Sequence Data

After smoothing and normalization, we divided the data into overlapping 90-day sequences, which is illustrated in Equation 4. As an example, the first sequence consists of sales data from January 1 to March 31, and then the next from January 2 to April 1, and so on. The target variable for each sequence would be the sales demand following the 90-day period, which in our case is the next day's sales data.

If the target is the sales data for the next day, and you represent the sales data for a single day as $S_t$, then the target for sequence $X_i$ would be $S_{t+1}$, where $t$ is the last day in the 90-day sequence.

$$\mathbf{X}_i = \begin{bmatrix} S_{t-89} \\ S_{t-88} \\ \vdots \\ S_t \end{bmatrix} \tag{4}$$

And for the target variable $y_i$, which is the sales data for the day following the 90-day sequence:

$$\mathbf{y}_i = S_{t+1} \tag{5}$$

These sequence(s) represented in Equation 5 and their respective target pairs are then split into a 80-20 split for training and validation purposes.

### 3.4. LSTM Model & Training

The LSTM architecture block diagram shown in Figure 1 consists of a cell state and three (3) gates: the input gate, which regulates the addition of new information to the cell state; the forget gate, which controls the removal of information irrelevant to the prediction task; and the output gate, which determines what information from the cell state to use in the output.

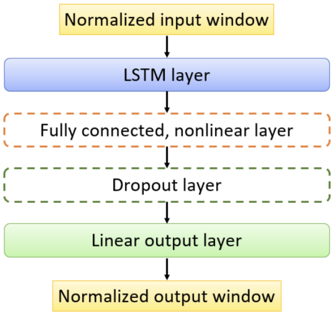

Figure 1: LSTM Model used during training

In order for training the Mean Squared Error (MSE) loss function was used. The MSE is often used in regression problems[7] due to it's simplicity, which is the calculated average of the squared differences between the predicted values and the actual values as can be seen in Equation 6.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{6}$$

Details of the learning rate we employed for training the LSTM model can me seeing in Figure 3 and is explained in detail in the following sections (Results & Discussion) below.

## 4. Results

The subsequent sections are dedicated to presenting both numerical data, used parameters, and visual illustrations that present the work and how it performed in regards to this research.

A substantial amount of the project involved programming, testing, deploying, and training the LSTM model utilizing MTSU[5]'s High-Performance Computing (HPC) resources.

A general analysis is provided, offering a visual review at the model's performance, including loss over time (Fig. 3), as well as other evaluation criteria. In addition to numerical representations, I will include an array of visual aids to convey the general performance of the model, the dynamics of the HPC resource utilization, and the general processes of training and validation. These visuals are designed to enhance the comprehensibility of the data and to bring to light the details of the LSTM's computational work that was performed for this paper.

[5] Middle Tennessee State University

## 4.1. Data Preparation & Normalization

As discussed in detail in the methods section of our research, we observed that the Kaggle dataset[3] exhibits significant noise on a daily basis. This noise (see Std. in Table 1) can prevent model learning the underlying trends and patterns in the data, making it challenging to extract meaningful sales demand insights.

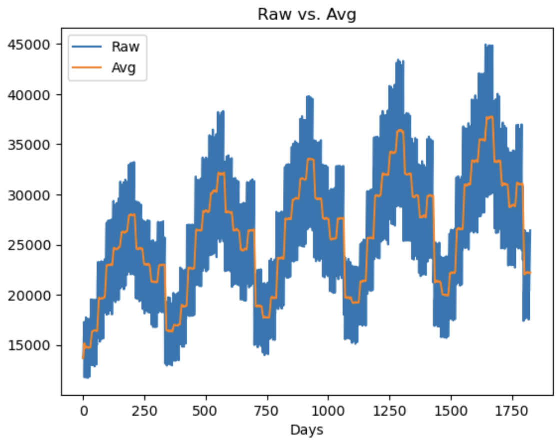

Figure 2: Raw (Noise) Data vs. 7-Day Average

To address this issue, we implemented a smoothing technique by calculating a 7-day moving average (see Equation 2 on page 2). This approach effectively reduces the daily fluctuations, resulting in a much smoother trend line that more accurately reflects the underlying demand patterns. The impact of this smoothing technique is clearly evident when examining Figure 2. The visual shows the blue line representing the raw, un-smoothed data, while the orange line shows the smoothed data using the 7-day average. The comparison between these two lines in the plot highlights the potential effectiveness of the smoothing process, showcasing a more coherent and interpret-able trend in the demand.

## 4.2. Model Training & Hyper-Parameter Tuning

In Table 2 the used for hyper-parameters in the LSTM model setup are listed. Of note should be that we employed a 4-Layer LTSM model, this is due the us using real-world data, which as described in the Methods section was noisy. Adding more layers[6] our LSTM model should be more robust to such irregularities, which allows for distinguishing between genuine trends and outliers[1]. It should be noted that since dropout layer(s) (see Figure 1 & Table 2) are only applied during training, the architecture used in the training phase differs from the one used when performing predictions. The one used during training is schematically illustrated in Figure 1 on page 3.

**LSTM Parameters**

| | | | |
|---|---|---|---|
| Learning Rate | 0.001 | Dropout Rate | 0.1 |
| Hidden Layers | 50 | LSTM Layers | 4 |
| Number of Epochs | 150 | Sequence Size | 90d |
| Optimizer | ADAM | Training (on GPU) | ∼6min |
| Activation Func. | ReLU | | |

TABLE 2: Hyper-Parameter for LSTM Model

The training time for this compact model was quite brief, averaging only about 6 minutes on the HPC GPUs performing 150 epochs (see Table 2). It should be noted that the model would based on the MSE-Loss curve perform well at 100 or even less epochs, since both curves in Figure 3 smooth out at around 80 epochs.

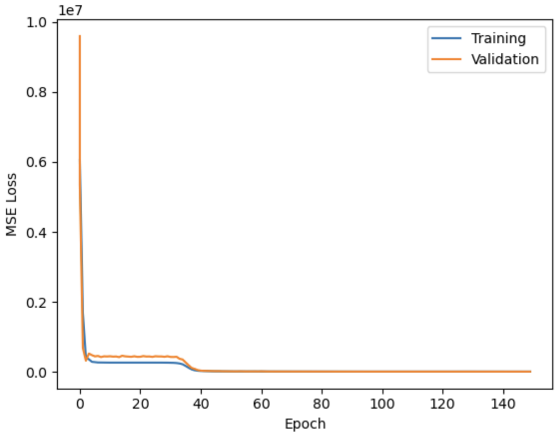

Figure 3: MSE Loss Details on Learning

Furthermore, evidence of overfitting was not observed in my analysis (Fig. 3), which is indicated by the loss curves for both training and validation. This is most likely a results of us using a dropout rate of 0.1 (see Table 2) to prevent overfitting. The curves remained mostly parallel throughout the learning process, suggesting a consistent performance between the training and validation phase. We therefore believe that the parallel trend suggests the model generalized well.

## 4.3. Plotting of Actual and Predicted Values

The training validation of the model are quite promising. Figure 4 illustrates that it[7] overall follows the true path during the training and test phases. Overall this trend indicates that both lines follow a similar pattern,

[6] More than ”1” layer is complex

[7] The LSTM Model

with the predicted values closely mirroring the true values. This suggests that the LSTM model has learned the underlying pattern in the training data well and is able to predict the future values with some degree of accuracy.

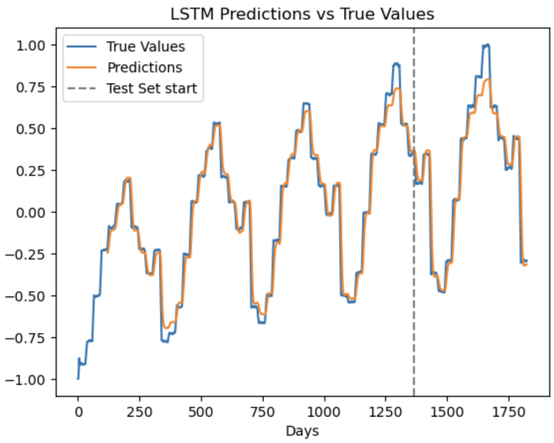

Figure 4: True vs. Predicted during training of model

## 5. Discussion

The following section reviews in detail the True vs. Predicted (see Figures 4 & 5) results of the LSTM model training. Using separate training and test datasets was critical to ensure that the model learns to generalize from sales demand patterns[8] in the data rather than memorizing it. With this in mind the test data provides an unbiased evaluation of the model's predictive performance on unseen data, reflecting its potential effectiveness in real-world applications. As previously mentioned in Table 1 on page 2 the 80 / 20 split rule was used for training and validation data.

### 5.1. Review of Plots

In Figure 5 it is apparent that the LSTM model as able to learn the underlying seasonality of the sales and would therefore be able to predict future demands. However looking closer one can see in Figure 4 as time moves forward that the difference[9] between true and predicted values skew higher over time. A more detailed view of this is also displayed in Figure 5 on page 5.

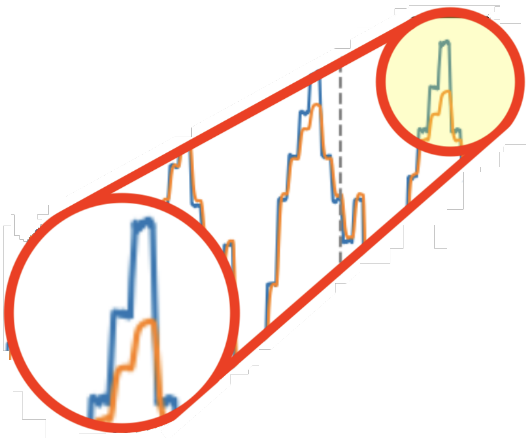

Figure 5: Prediction vs. True detail on peaks

Overall Trend review indicates that both lines follow a similar pattern, with the predicted values closely mirroring the true values. This suggests that the LSTM model has learned the underlying pattern in the training data well and is able to predict the future values with some degree of accuracy. Since the predicted values closely follow the true values, one could infer that the model has a good performance.

It should be noted that detail review of the ”peak” shows a slight degradation over time from the actual true values. This can be observed in Figure 5 in detail as the true (blue) value line peaks higher than the predicted (orange).

### 5.2. Future Work

In order to enhance the demand forecasting capabilities of our models, we anticipate more work to explore the extraction of additional contextual information related to the timestamps, such as seasons, weekdays, holidays, or any other relevant time-based factors. By incorporating these additional features into the dataset, we aim to provide the LSTM model with a richer contextual understanding of the underlying data, which could lead to more accurate predictions.

The strategy here would involve augmenting the dataset with these extracted features and subsequently comparing the performance of the LSTM models with that of the previous non-expanded dataset presented previously. This comparative analysis will shed light on the extent to which the inclusion of more temporal information influences forecasting accuracy.

Lastly, it would certainly be of benefit for the authors to compare the LSTM's predictive output to other similar models such as the Gate Recurrent Unit (GRU) or traditional RNNs models. This would provide additional inference that the LSTM has indeed learned the sales demand from a forecasting and seasonal prediction standpoint.

[8] Expl: Seasonality plays a large role in demand forecasts
[9] The difference is analogous of the error

## Acknowledgments

We extend our sincere gratitude to the two anonymous reviewers whose insightful comments and constructive critiques improved the quality of this paper. Their expertise and thoughtful feedback have been a great help in refining our research, arguments, and design layout of this paper.

## References

[1] Abeselomgebrekidan. *Pharmaceutical Sales prediction Using LSTM Recurrent Neural Network*. URL: https://medium.com/@abeselomgebrekidan12/pharmaceutical-sales-prediction-using-lstm-recurrent-neural-network-db0f980572cc.

[2] Marta Golabek, Robin Senge, and Rainer Neumann. "Demand Forecasting using Long Short-Term Memory Neural Networks". In: *CoRR* abs/2008.08522 (2020). arXiv: 2008.08522. URL: https://arxiv.org/abs/2008.08522.

[3] Kaggle. *Store Item Demand Forecasting Challenge*. 2018. URL: https://kaggle.com/competitions/demand-forecasting-kernels-only.

[4] Barış Karaman. *Predicting Sales, Forecasting the monthly sales with LSTM*. URL: https://towardsdatascience.com/predicting-sales-611cb5a252de.

[5] Ralf C. Staudemeyer and Eric Rothstein Morris. "Understanding LSTM - a tutorial into Long Short-Term Memory Recurrent Neural Networks". In: *CoRR* abs/1909.09586 (2019). arXiv: 1909.09586. URL: http://arxiv.org/abs/1909.09586.

[6] Shudong Yang, Xueying Yu, and Ying Zhou. "LSTM and GRU Neural Network Performance Comparison Study: Taking Yelp Review Dataset as an Example". In: *2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)*. 2020, pp. 98–101. DOI: 10.1109/IWECAI50956.2020.00027.

[7] Tong Zhou. "Improved Sales Forecasting using Trend and Seasonality Decomposition with LightGBM". In: *2023 6th International Conference on Artificial Intelligence and Big Data (ICAIBD)*. IEEE, May 2023. DOI: 10.1109/icaibd57115.2023.10206380. URL: http://dx.doi.org/10.1109/ICAIBD57115.2023.10206380.