

# PAML Final Report: Walmart Time-Series Price Forecasting

Sanjay Thallam  
Cornell Tech  
Electrical and Computer Engineering  
sct86@cornell.edu

Quynh Dang  
Cornell Tech  
Operations Research and Industrial Engineering  
qtd2@cornell.edu

Link to Github Repo: [GitHub Repository](#)

## Abstract

*This project addresses the problem of multivariate time series forecasting, which is crucial for predicting variables like weather conditions or market demands. Due to the inherent complexity and stochastic nature of such data, accurately forecasting these variables is essential for effective decision-making in various sectors. Our approach benchmarks newly proposed neural networks, including RNN, LSTM, GRU, CNN against the classical algorithm ARIMA and the baseline of Simple Averaging, using large-scale real-world dataset from Walmart's M5 sales data analysis challenge on Kaggle. In our design, the front-end was built with Streamlit to facilitate data input and visualization of forecasts, while the back-end manages data processing and model execution. Models are evaluated using the metric of Mean Squared Error (MSE). Preliminary results suggest that Neural Networks offer significant improvements over the traditional method of ARIMA and Simple Averaging. This study contributes to the machine learning field by providing insights into the scalability and effectiveness of advanced neural architectures in handling complex, real-world forecasting tasks.*

## 1. Introduction

In the realm of data science, multivariate time series forecasting is a pivotal area of study due to its vast applicability across various critical fields such as meteorology, finance, and supply chain management. Accurate forecasting models can significantly improve decision-making processes by predicting future events based on historical data. However, the inherent complexity and unpredictability of real-world data present substantial challenges in developing robust forecasting systems. Traditional statistical methods, while being foundational in time series analysis, often fail to capture the nonlinear relationships and complex dependencies present in multivariate time series data.

The advent of machine learning, particularly advanced neural network architectures, has introduced promising alternatives to conventional models. These neural networks have emerged as particularly effective due to their ability to discern intricate patterns and dependencies in extensive data sequences. Despite their advantages, there is a knowledge gap regarding their performance in practical, real-world scenarios when compared to traditional forecasting methods, especially under constraints of limited data availability.

This project aims to bridge this gap by systematically comparing the performance of four neural networks, including Recurrent Neural Network (RNN, which is first generation), Long Short-Term Memory (LSTM), GRU (Gated Recurrent Units), and Convolutional Neural Network (CNN), and against the classical algorithms Auto Regressive Integrated Moving Average (ARIMA) models in the context of large, real-world datasets. The Walmart's M5 Sales Forecasting Dataset provide a robust platform for evaluating the effectiveness of these models in real-world applications.

**Technical Focus and Novelty** Our approach leverages the latest advancements in neural networks, which allow the model to focus on relevant temporal features without being overwhelmed by data scale or noise. The project explores not only the predictive accuracy of these models but also their computational efficiency and scalability, which are crucial for real-time applications. The novelty of this work lies in its comprehensive benchmarking of advanced machine learning models against time-tested traditional methods across diverse datasets, providing a nuanced understanding of their practical benefits and limitations.

**Prior Work** Historically, time series forecasting has relied heavily on models like ARIMA, which are well-suited for uni-variate series with stable seasonal patterns. However, the explosion of available data and the advent of complex, dynamic systems have exposed the limitations of these models. Recent research has increasingly focused on neural networks due to their flexibility and capacity for modeling complex nonlinear relationships [7]. Yet, much of this research has been confined to theoretical or simulated envi-

ronments [6]. Our work extends this research by applying these models to real-world data, providing insights into their operational effectiveness.

**Machine Learning Pipeline** The project’s machine learning pipeline includes data exploration to identify key patterns and anomalies, pre-processing to handle non-stationary and normalize inputs, and rigorous model training phases. Each model’s performance is evaluated using appropriate metrics—Mean Squared Error (MSE) for retail forecasting. We anticipate that our findings will offer clear insights into which models perform best under various real-world conditions and why.

**Impact and Ethical Considerations** The potential social impact of improved forecasting models is immense, particularly in sectors like public safety, healthcare, and urban planning where timely and accurate predictions can lead to better resource allocation and potentially save lives. However, the deployment of these models raises ethical considerations, particularly regarding data privacy, the potential for biased outcomes, and the transparency of algorithmic decisions. Addressing these issues is crucial to ensure the responsible use of forecasting technology.

## 2. Background

Multivariate time series forecasting is a well-studied problem with applications in various domains such as finance, retail, and weather prediction. Classical statistical models like Auto regressive Integrated Moving Average (ARIMA) [2] have been widely used for this task. However, these linear models may struggle to capture the complex patterns and nonlinearities present in real-world multivariate time series data.

In recent years, neural network-based models have shown promising results in multivariate time series forecasting [7]. Recurrent Neural Networks (RNNs) are effective for processing sequential data for predictions but have limitations due to short-term memory issues. Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs) were developed to address these limitations by incorporating gating mechanisms [6].

Long Short-Term Memory (LSTM) [3], a type of recurrent neural network, has been extensively applied to this problem due to its ability to learn long-term dependencies.

The Gated Recurrent Unit (GRU) represents a more recent evolution of Recurrent Neural Networks, sharing many similarities with Long Short-Term Memory (LSTM) networks. Unlike LSTMs, GRUs eliminate the cell state and rely solely on the hidden state for information transfer [5]. Additionally, GRUs simplify the architecture by utilizing only two gates: a reset gate and an update gate. These gates are capable of learning which data in a sequence is important to retain or discard. By selectively passing relevant information along the sequence, they enhance prediction ac-

curacy. Most state-of-the-art outcomes in recurrent neural network applications are achieved using LSTMs and GRUs. These networks are also prevalent in tasks such as speech recognition, speech synthesis, and text generation, and can even be utilized for generating video captions [5].

There is also another breakthrough neural network developed by Yann Lecun and his team in 1998 which is called Convolutional Neural Network (CNN) [4]. CNN is a class of deep learning models primarily used for processing structured grid data, such as images. CNNs utilize convolutional layers to automatically and adaptively learn spatial hierarchies of features from input data, making them highly effective for image recognition and classification tasks.

The proposed work in this project builds upon these recent advancements in neural network-based multivariate time series forecasting for the Walmart M5 data set[1]. The key differences between the proposed work and prior research are: 1. Comprehensive evaluation of four different neural network models on a real-world multivariate time series datasets. 2. Detailed analysis of the strengths and weaknesses of the neural models compared to classical approaches, providing insights into when and why each type of model may be preferred. 3. Exploration of techniques to improve the performance of the neural models, such as careful hyper parameter tuning and architectural modifications.

The proposed work aims to provide a comprehensive understanding of the trade-offs between neural and classical methods for multivariate time series forecasting.

## 3. End-to-End ML Pipeline

### 3.1. Back-End

For our models, we are using the Walmart M5 dataset from Kaggle which was released in 2020. The Walmart M5 dataset is a dataset containing unit sales for over 3000 different products sold at Walmart. Each item has over 5 years’ worth of daily information at 9 different Walmart locations. Some of the given information includes Product Category, Selling Prices, Promotional Activities, and Calendar Information. For our project, we were attempting to forecast how many sales a certain item would get across a 28-day validation period. This dataset had a separate training dataset (5 years) and a validation dataset (28 days). In this dataset, there is a mix of Floating Point (price), Boolean (Promotions, Holidays), and Categorical (Product Category) data types. Ground truth labels are provided for this case.

For data exploration, we showed the time series for item sales for a given product based on which product the user wants to project for. Figure 1 below shows an example plot for a given item.

This type of plot made the most sense for our application as we are working with a time series prediction for our

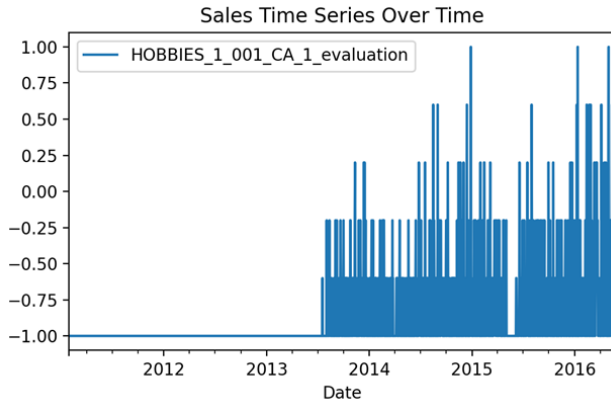


Figure 1: Example visualization of time series data

project and this gives a good visual for sales over time.

Multiple preprocessing techniques were used to ensure the data was ready for machine learning models. The first of these was data cleaning where we had to deal with some messy data in terms of sales frequency. In the cases where data was missing, it was assumed that no sales for that given product were made on that day. Additionally, if the promotion data was missing, it was also assumed that no promotions were ongoing. We also performed regularization on the sales data to help simplify model training and help reduce outlier information. We integer-encoded the categorical data as well to help use it as a training feature.

### 3.2. Data Collection, Exploration & Processing

Describe the dataset(s) you plan to use for training machine learning models including the type of data provided, quantity of data, year it was released (if available), source of the dataset (e.g., Kaggle), and how you plan to use it in the project. State whether ground truth labels are provided. If there are none, state how you plan to generate annotations or provide justification for the planned evaluation procedure. At least one dataset is required.

State the preprocessing techniques used to prepare the dataset for ML algorithms. Provide justification for these techniques and how they are used in the next steps of the ML pipeline.

### 3.3. Methods and Model Training

Various machine-learning model architectures were used in this project. The first was the Recurrent Neural Network (RNN) model. When compared to the traditional CNN, the RNN is specialized in processing sequential data and has internal memory which lets them reference previous elements in the sequence. This makes it ideal for time-series forecasting, hence why we used it. The next model used was the Gated Recurrent Unit (GRU) model. When compared to

| Location | Category    | Value |
|----------|-------------|-------|
| CA_1     | HOBBIES_1   | 23    |
| TX_2     | HOUSEHOLD_2 | 47    |
| WI_3     | FOODS_3     | 94    |

Table 1: Items being Tested for Model Accuracy

the RNN, the GRU has a simpler internal structures which is based on internal gating. This makes GRUs better at deciding what past information to save to help defeat the vanishing gradient problem. One of the issues with the GRU is that the simpler structure may make it struggle in cases where there are extremely long sequences. The final model used was the Long Short-Term Memory (LSTM) model. The big advantage that the LSTM has over the RNN is the fact that it overcomes the vanishing gradient problem as a result of its more complex internal structure. This makes them better at learning long-term dependencies compared to the RNN.

The inputs to these different models include the Date, item price, promotion data, and holiday information and the outputs from this model are the projected sales on that given day. During training, we did an 80/20 sequential train/test split meaning we used the first 4 years for a given item as training data and the last year as testing data. For validation, we used the separate 28-day dataset that was given for validation. In terms of model parameters, we made it fully configurable for the user through our streamlit page. We decided to do it this way to let them experiment with different options so they can also better understand machine learning models. In our case, over fitting is a larger issue than under fitting. We helped minimize this by regularizing our data preprocessing and running a limited number of training epochs.

### 3.4. Model Evaluation

Describe the experiments and error analysis.

When it comes to the evaluation of the model, we decided to run a set of simple experiments to see how the different models performed under similar conditions. Each of the models was tested with the same 3 items under test to see how well they would perform in terms of forecasting the future demand for those items. The items which were forecasted are listed in table 1 below.

The methods that are being evaluated are the traditional RNN, the GRU and LSTM models. We also did build out the ARIMA and CNN models but were having issues integrating them with streamlit. In terms of metrics for evaluating how well each model did, we will be using the Mean Squared Error and Root Mean Squared Error. These are the 2 main types of errors used when measuring the performance of time series data. The hyper parameters used were 100 epochs, a training rate of 1E-4, 1 layer, and a layer size

of 64.

### 3.5. Results

As mentioned earlier, we tested 3 different datasets with each model to see the real world performance of the model. To measure the performance of the models, we used the Mean Squared Error and the Root Mean Squared Error to measure the results for the RNN, GRU and LSTM models. Table 2 shows the mean squared error for each test and the Table 3 shows the root mean squared error for each test.

|      | Hobbies_1_023 | Household_2_047 | Foods_3_094 |
|------|---------------|-----------------|-------------|
| RNN  | 0.0972        | 0.0426          | 0.0918      |
| GRU  | 0.0686        | 0.0343          | 0.0665      |
| LSTM | 0.0683        | 0.0336          | 0.0848      |

Table 2: Mean Squared Errors

|      | Hobbies_1_023 | Household_2_047 | Foods_3_094 |
|------|---------------|-----------------|-------------|
| RNN  | 0.3118        | 0.2066          | 0.3030      |
| GRU  | 0.2620        | 0.1852          | 0.25579     |
| LSTM | 0.2614        | 0.1835          | 0.2913      |

Table 3: Root Mean Squared Errors

Looking at the tables results above, overall it seems that the RNN model performed worse across the board compared to the LSTM and GRU models. This is likely attributed to the fact that the later models are more advanced. The next step was to analyze how well the models did visually speaking. Figures 2, 3, 4 show the plot from the evaluation testing.

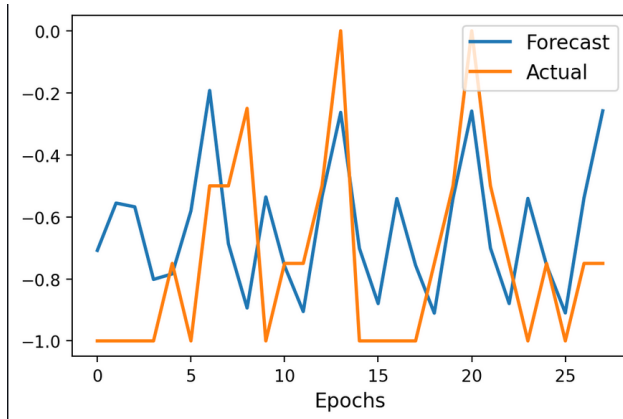


Figure 2: RNN Forecast vs Actual Demand for 3rd Item

In comparison with the error rates, from a visual perspective it seems that the RNN almost did a better job at forecasting compared to the models as the other ones seem compressed in terms of forecasting the demand over time. This

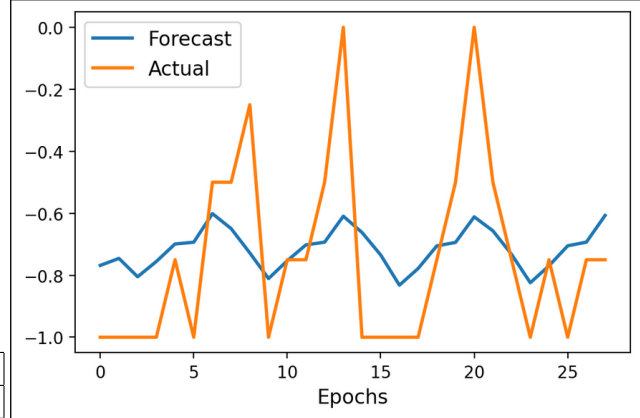


Figure 3: GRU Forecast vs Actual Demand for 3rd Item

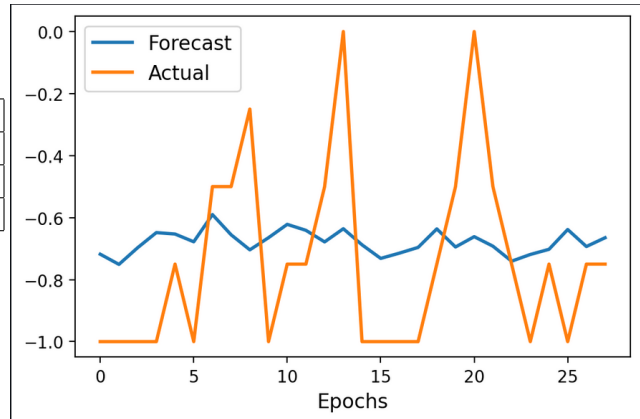


Figure 4: LSTM Forecast vs Actual Demand for 3rd Item

is in contrast with the RNN which seems a lot spiky and in line with real world performance. This may be caused by over fitting due to how the models were trained.

### 3.6. Model Deployment

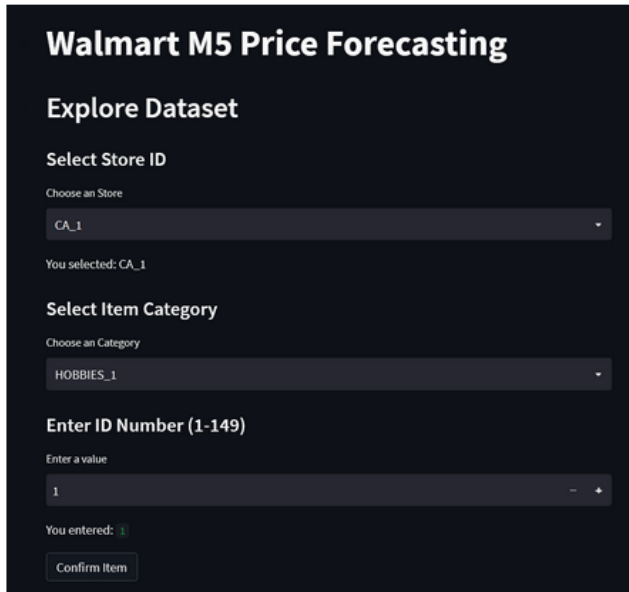
When it comes to this model, the core application for this model is to help retailers better forecast demand for their products. In the retail world, margins are thin and inventory is a major expense, thus by improving product sales forecasting, businesses can better optimize their supply chain. This will directly help them reduce costs, thus, saving the average consumer money while helping increase the venture's profitability.

Thinking about the importance of applications such as these, oftentimes, small businesses lack the expertise to perfectly predict their needs for certain supplies/items for sale so taking advantage of algorithms can help them improve their operations. Going further, when looking at the ethical and societal elements of using algorithms, their failure could lead to negative effects on business performance which may lead to individuals losing their jobs. This is a

negative burden on society, thus we must ensure that these models work reliably before large-scale deployment.

### 3.7. Front-End (Streamlit)

For the front end of our application, our overall goal was to keep it as simple as possible for the end user. At a high level, there are 2 core selection areas. The first is a section allows the user to select the product, store location, and product category that they want to perform the analysis on. Figure 5 below shows this first part of the UI.



The screenshot shows a web interface titled "Walmart M5 Price Forecasting". Under the "Explore Dataset" heading, there are three main sections: "Select Store ID" with a dropdown menu showing "CA\_1", "Select Item Category" with a dropdown menu showing "HOBBIES\_1", and "Enter ID Number (1-149)" with a text input field containing "1". A "Confirm Item" button is located at the bottom of the form.

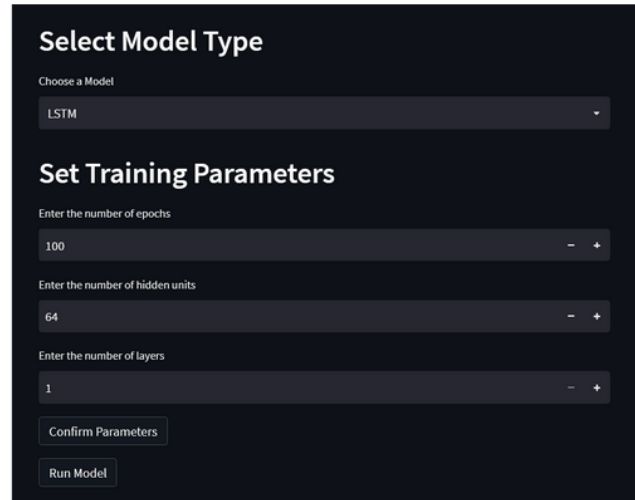
Figure 5: Item selection part of the UI

The front end for this part connects with the back end by essentially transmitting the data in the selection boxes to the back end when the confirm button is pressed. This button will save the values and auto generate the string used to index the internal dataframe. The second part of the UI is focused on letting the user select their training parameters. As a learning experiment, we wanted to make it so the user can learn as much as possible. Therefore, we made it so that the parameters in the model are configurable by the end user. Figure 6 below shows this selection panel

On this configuration panel, the user can select the model type they want to use, the number of training epochs, and the size of the layer under test. This allows a good amount of configurability without getting too complex for the average non-technical user.

## 4. Risk & Mitigation

Given our limited experience with machine learning and specifically with time series algorithms, a significant portion of the project will be dedicated to learning and implementing these systems effectively. We anticipate challenges



The screenshot shows a web interface titled "Select Model Type". Under the "Set Training Parameters" heading, there are four input fields: "Choose a Model" (dropdown menu showing "LSTM"), "Enter the number of epochs" (text input field showing "100"), "Enter the number of hidden units" (text input field showing "64"), and "Enter the number of layers" (text input field showing "1"). At the bottom, there are two buttons: "Confirm Parameters" and "Run Model".

Figure 6: Model Configuration Options

related to understanding the nuances of time series data and optimizing LSTM and RNN models for our specific needs. To mitigate these challenges, we plan to engage with existing literature, seek peer advice, and possibly consult with experts in the field.

## 5. Conclusion

Overall, performing this project has given our team the ability to learn more about time series forecasting and how it can be applied in the real world. The initial goal of this project was to use time series forecasting to predict item demand using the Walmart M5 dataset. We were successfully able to do this using RNNs, GRUs, and LSTMs for this project and we integrated all 3 into our project successfully. Additionally, we were able to learn more about ARIMA and CNNs but were unable to fully integrate them with streamlit in our end product. The overall approach for this project was to build out the back end and ensure it was robust overall, and then we integrated it with a streamlit front end to make it easy for the end user to use. We found that overall, GRUs and LSTMs performed better than RNNs but that they were also more prone to over fitting. Therefore in our future work, we will look into more optimal training setups for the LSTM and GRU based networks.

In terms of implications of this project on society, we can confirm that businesses can successfully use models such as these to better predict demand for products they sell. This can lead to more operational efficiency, leading to more profit and more people that the same business could potentially employ, therefore positively impacting society.



## 6. Team Member Contribution

Each team member will be responsible for specific aspects of the project to ensure a collaborative and balanced workload:

Sanjay focused on the back-end development, particularly the data preprocessing and the optimization of model training functions. Sanjay will also take the lead on front-end development, designing the user interface and integrating it with the back-end. Quynh concentrated on finding the dataset, reading the research papers, coming up with model implementation and evaluation, and ensuring that both the baseline and advanced models are correctly set up and assessed. Quynh also reviewed Sanjay's code for errors and performance improvement of the neural networks and ARIMA.

By clearly delineating responsibilities, we successfully leveraged individual strengths and facilitate effective teamwork throughout the project.

### 6.1. Technical Components

Sanjay and Quynh have divided the technical components equally for this project, as outlined above.

### 6.2. Writing Components

Sanjay has contributed to the report key ideas and outlines. Quynh has written the proposal. Sanjay and Quynh equally contributed to the final project presentation and this final report.

## References

- [1] M5 forecasting - accuracy. <https://www.kaggle.com/competitions/m5-forecasting-accuracy>. Accessed: 2023-04-26. 2
- [2] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015. 2
- [3] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 2
- [4] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 2
- [5] Christopher Olah. Illustrated guide to lstms and grus: A step by step explanation. 2019. Accessed: 2024-05-15. 2
- [6] Peter T Yamak, Li Yujian, and Pius K Gadosey. A comparison between arima, lstm, and gru for time series forecasting. In *Proceedings of the 2019 2nd international conference on algorithms, computing and artificial intelligence*, pages 49–55, 2019. 2
- [7] G.P. Zhang. Neural networks for time-series forecasting. In Grzegorz Rozenberg, Thomas Bäck, and Joost N. Kok, editors, *Handbook of Natural Computing*. Springer, Berlin, Heidelberg, 2012. 1, 2