University of Brasília — Faculty of Technology
Prof. Dr. Eugenio Liborio Feitosa Fortaleza

SUPPLY CHAIN REPLENISHMENT OPTIMIZATION: INTEGRATION OF LEAD TIME BOOTSTRAPPING AND RECURRENT NEURAL NETWORKS FOR DEMAND FORECASTING

Pedro Trajano Ferreira 190055251

Caio Moreira Santana 190042249

FINAL REPORT

ENM0134 - INTEGRATED MANUFACTURING SYSTEMS

Brasília, 10 de July de 2024

Abstract

In today's competitive scenario, effective replenishment management in supply chains has become an essential strategic advantage for companies. Ensuring continuous product flow through the chain, without excess or shortage, not only improves customer satisfaction but also optimizes operational costs. This study focuses on determining the optimal time to replenish products in a complex system involving suppliers, Distribution Centers (DCs), and stores, using advanced techniques to handle uncertainties in delivery times and forecast future demand.

The integration of methods such as bootstrapping to model lead time uncertainty and predictive analytics to estimate future demand is crucial. These approaches not only provide a more precise and well-founded view for replenishment decisions but also enable more efficient inventory management, essential for uninterrupted supply chain operations.

This study's objective goes beyond theory, seeking to compare the effectiveness of the proposed method with conventional models, evaluating its feasibility for practical implementation in real scenarios. The research aims not only to improve companies' operational efficiency but also to significantly contribute to market competitiveness by ensuring intelligent and optimized management of logistics resources.

Keywords: Logistics · Supply Chain · Prediction ·

Lista de ilustrações

Sumário

1 Introduction

This work aims to investigate supply chain replenishment optimization, specifically in a system involving suppliers, Distribution Centers (DC), and retail stores. The main objective is to develop an effective method to determine the optimal time for purchasing new products, based on historical sales data from stores and the lead times between supplier and DC, and between DC and store.

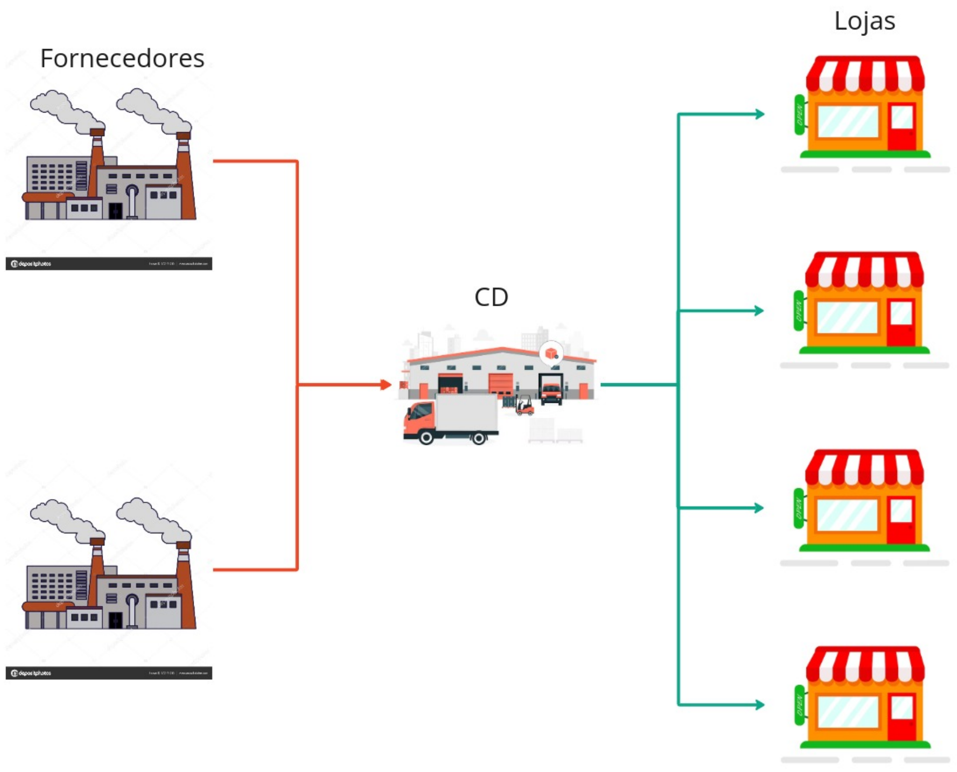

Figura 1 – Inventory movement model

To address the uncertainty associated with lead times, a bootstrapping technique was implemented that combines delivery times from supplier to DC and from DC to store, generating a probability density distribution (KDE). From this distribution, the algorithm defines an acceptable error percentage to guide replenishment decisions, ensuring an adequate service level while minimizing inventory costs.

For forecasting future demand, a Recurrent Neural Network (RNN) was used. This deep learning approach is particularly suitable for capturing complex temporal patterns present in sales data. The RNN was chosen for its ability to model data sequences and provide accurate predictions, which is crucial for making informed replenishment decisions.

The data used in this study is artificially generated, simulating realistic scenarios, but does not correspond to actual sales data, implementing the randomness present in these systems, including holidays, weekends, and promotions that significantly increase sales numbers. This choice provides the necessary flexibility to test and validate the model under various controlled conditions, ensuring the model's robustness and applicability.

The work includes analyses for both high and low turnover products. We aimed to avoid standard approaches that approximate data to probability distributions already known in the literature, such as low turnover products following a Poisson distribution or high turnover products that, depending on the coefficient of variation, follow a Normal or Gamma distribution (SOUZA RODRIGUES, 2014). In real-world product sales cases, many of these assumptions do not apply, for example, product seasonality would require analysis of the time interval needed to approximate the product to the expected distribution.

We did not address issues such as minimum and maximum quantities of products that a truck can transport, nor the logistics costs of items inside and outside inventory, or supplier errors that might result in shipping a different number of items than expected.

2 Modeling

2.1 Lead Time

To model the lead time, a simple normal distribution model $N(\mu, \sigma^2)$ was applied, since logistics processes depend on various random factors such as weather conditions, traffic, loading and unloading times of parts, allowing us to use the Central Limit Theorem which essentially states that the sum of many independent random variables tends toward a normal distribution.

Since products cannot be negative and are discrete in nature, the following modification was made to the lead time distribution:

$$lt[d] = \left\lceil \left| N(\mu, \sigma^2) \right| \right\rceil$$

Where $lt[d]$ represents the product's lead time on day d and $\lceil \ \rceil$ represent the ceiling function. Note that the unit of measurement will be days, as there is no need to work with smaller time units.

To model the logistics process between supplier and distribution center, large μ values were used (such as one and a half months) to represent the large distances in supply chains where industries are often located in different states or even countries. A high σ^2 value (like 15) was also chosen to represent the uncertainty of the process, since it's outside the company's control. For modeling the lead time between DC and store, small values of μ and σ^2 were used, as in the studied case it's assumed the distribution center is relatively close to the stores and the variance is low because it's an already optimized and recurrent process.

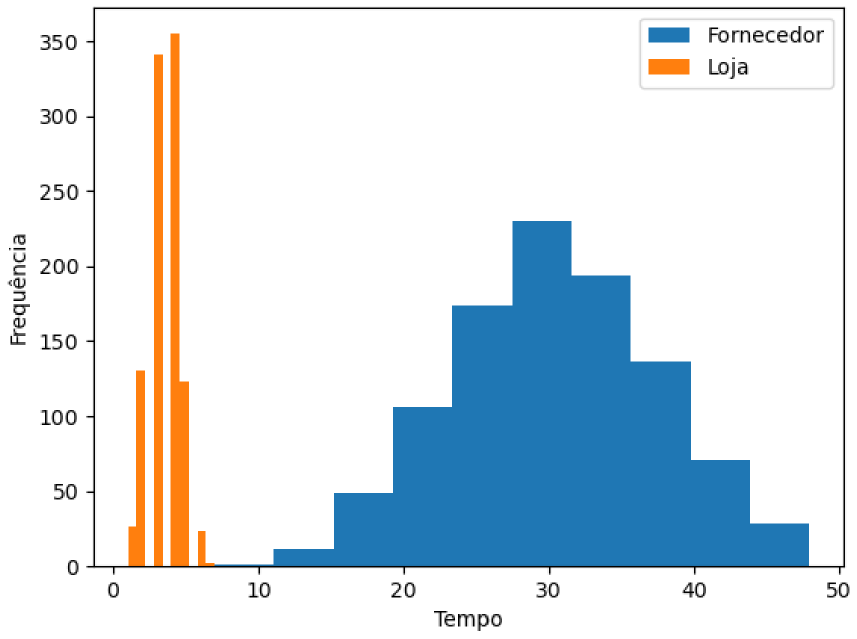

Figura 2 – Example histogram of Transport Times

2.2 Demand

Product sales alone are not the best way to obtain true information about demand, because if, for example, the product runs out in the store, all other people who want to buy it won't be able to. Therefore, the value of how many products should be in the store would be better than the actual sales. The best data for analyzing product demand is customer interest - every person who wants to buy the product should be recorded as having sought the item. However, this requires appropriate company culture and sufficient maturity to collect this type of information.

To model demand, first it's necessary to work with the randomness of sales which is the foundation of the sales process. For this, a normal distribution $N(\mu,\sigma^2)$ around a point was used, treated as follows:

$$sz[d] = \mu\left(1 + \rho_i \operatorname{sen}\left(\frac{2\pi d}{T_i} + \phi_i\right)\right)$$

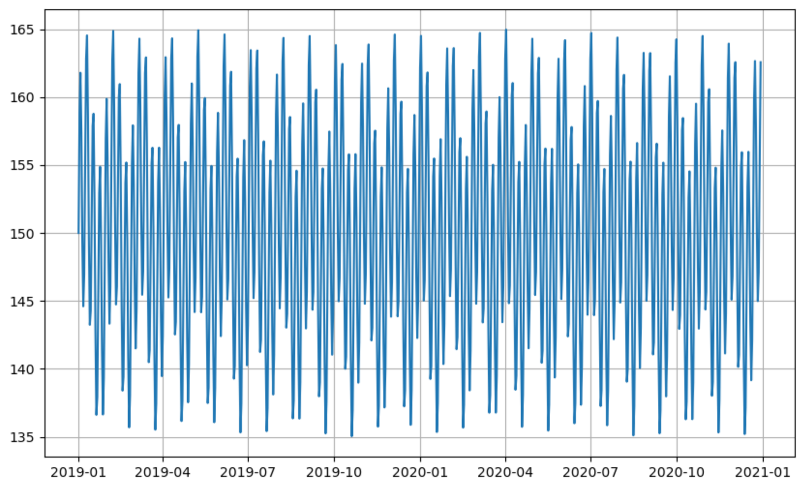

Figura 3 – Seasonality

Where *sz* means seasonality. Three different periods were implemented: weekly, monthly, and annual. The ρ_i indicates how strong the seasonality is, T_i means the number of days the seasonality repeats (in this case 7, 30, and 365), and finally ϕ_i indicates the sine wave phase. Although implemented, this parameter won't be used as it falls outside the project scope.

$$\psi[d] = \mu \cdot N(\mu, \sigma^2) \cdot \sum_{i=1}^{3} \left(1 + \rho_i \operatorname{sen}\left(\frac{2\pi d}{T_i} + \phi_i\right)\right)$$

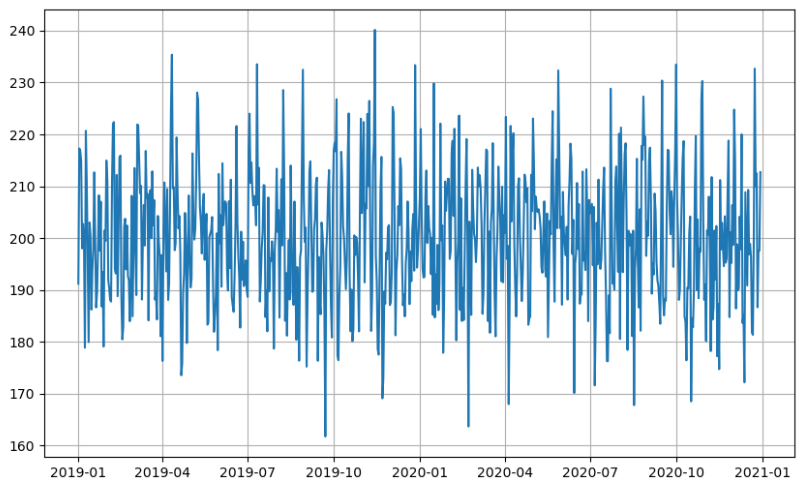

Figura 4 – Seasonality with random term

This formula represents the intermediate step from which the modeled demand is derived. To model a company's gradual growth over time, plus another analysis indicating a relationship with the day of the week (assuming Sunday is day 0, Monday is day 1, and so on), and also defining the set $\mathbb{H}$ representing holiday days, we can build the following function:

$$\Psi[d] = \begin{cases} 0 & \text{if } d \equiv 0 \pmod 7 \text{ or } d \in \mathbb{H} \\ \gamma\psi[d] & \text{if } d \equiv 6 \pmod 7 \\ \alpha\psi[d] & \text{if } \mathrm{rand}(0,1) < \chi \\ \psi[d] & \text{otherwise} \end{cases}$$

Note that in this case it's assumed there are no sales on Sundays, and Saturday sales are multiplied by a parameter γ (a value between 0 and 1 to represent the store being open half-day). A parameter α was also used to multiply if a uniformly distributed random variable is less than a parameter χ, representing an atypical sales day (like a promotion). The chosen value for χ was 0.005 and for α was 1.2.

Finally, the Geometric Brownian Motion was used as a basis - a stochastic process typically used to model price fluctuations - resulting in the following demand model:

$$dem[d] = \lceil |\Psi[d]| \rceil \cdot e^{\frac{d}{\tau}}$$

The parameter τ indicates the relaxation of demand growth, used in the order of 10^5. This gives us our demand model.

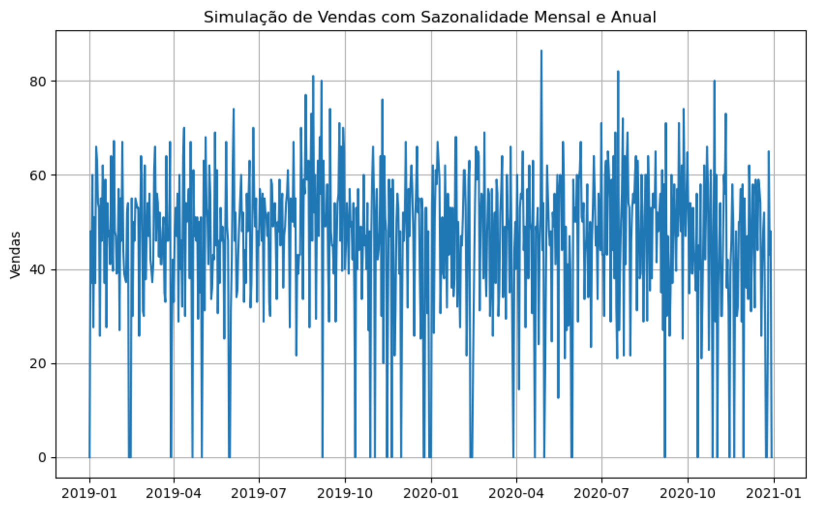

Figura 5 – Example sales profile

Note that the sales profile follows patterns found in medium to large companies that don't open on Sundays or holidays, with noticeable growth over the years, as seen below with a τ value in the order of 10^3.

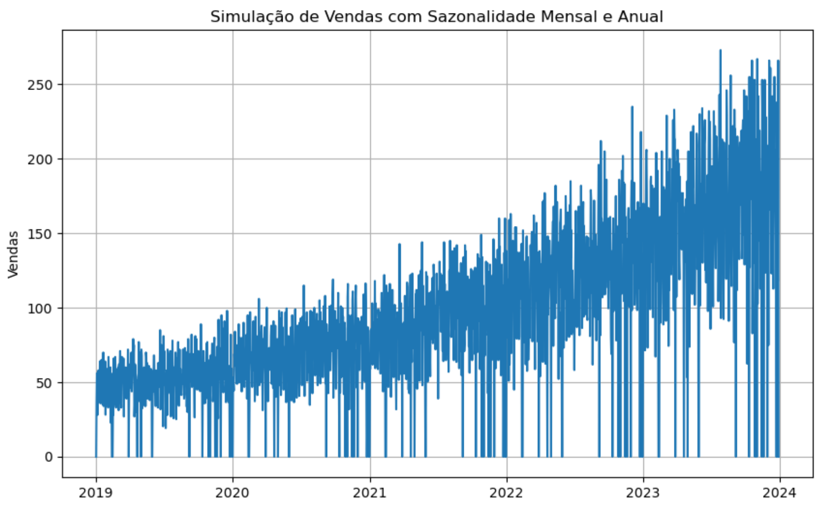

Figura 6 – Example sales profile

3 Bootstrapping

Bootstrapping is a statistical method of random resampling with replacement from the original sample. The purpose of this method is to enable better statistical analysis from the base data without assuming the population distribution type. Starting from an original sample of size n, the method creates multiple new samples by randomly drawing with replacement from the original data.

This method will be used to find a day value that represents an acceptable deadline for the entire logistics process from product purchase to arrival at the store via the Distribution Center.

3.1 Algorithm

The algorithm is as follows:

Given two datasets, one representing the delivery time from product purchase to arrival at the distribution center and another representing the transport time from the distribution center to the store, we can form a new dataset as follows:

- Let $A = \{a_1, a_2, \ldots, a_m\}$ be the set representing time from supplier to distribution center arrival.
- Let $B = \{b_1, b_2, \ldots, b_n\}$ be the set representing lead time from distribution center to store.

To form the new dataset:

1. For each iteration i from 1 to N:
   a) Randomly select an element a_i from A.
   b) Randomly select an element b_i from B.
   c) Sum the two elements: $c_i = a_i + b_i$.
2. The new set C will consist of the N generated c_i values:

$$C = \{c_i \mid c_i = a_i + b_i,\ a_i \in A,\ b_i \in B,\ \forall i \in \mathbb{N} | i < N\}$$

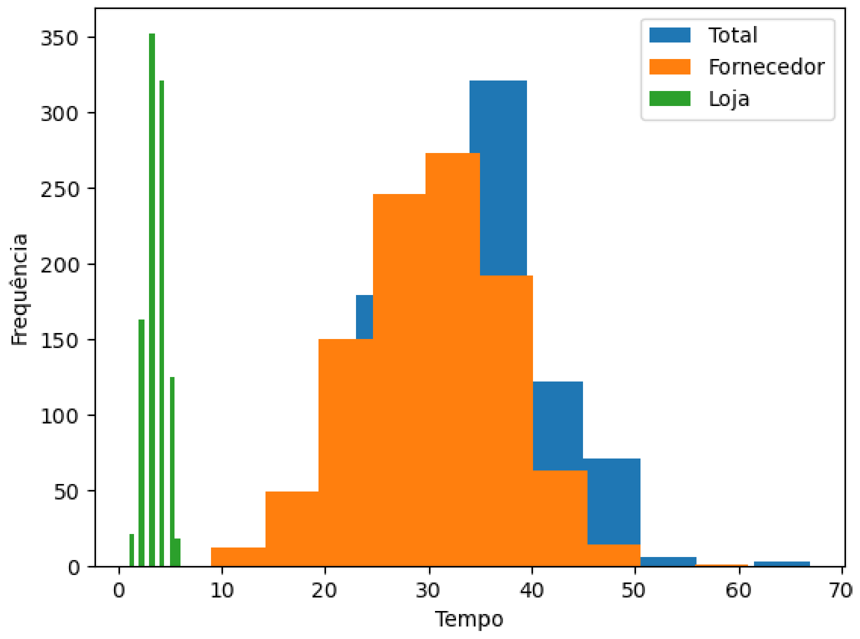

Figura 7 – Transport time bootstrapping

Note that the result resembles a normal distribution, like the two original ones. This is expected since the sum of two normally distributed random variables results in another normal curve.

From this dataset, we calculate the percentile - a measure that divides the sample (when ordered) into 100 parts, each representing approximately equal percentages of data. The k-th percentile P_k is the value of $\chi(x_k)$ corresponding to the cumulative frequency of $N \cdot k\%$, meaning:

1. To calculate the p-th percentile of set C, sort C in ascending order:

$$C_{\text{order}} = \{c_{(1)}, c_{(2)}, \dots, c_{(N)}\} \quad \text{where} \quad c_{(1)} \leq c_{(2)} \leq \cdots \leq c_{(N)}$$

2. The value of the k-th percentile (where $0 \leq k \leq 100$) is given by:

$$P_k = c_{\left(\left\lceil \frac{k}{100} \times N \right\rceil\right)}$$

The chosen value for k was 95, representing that in 100 supplier orders, only in 5 cases would the product not arrive at the store within the expected time.

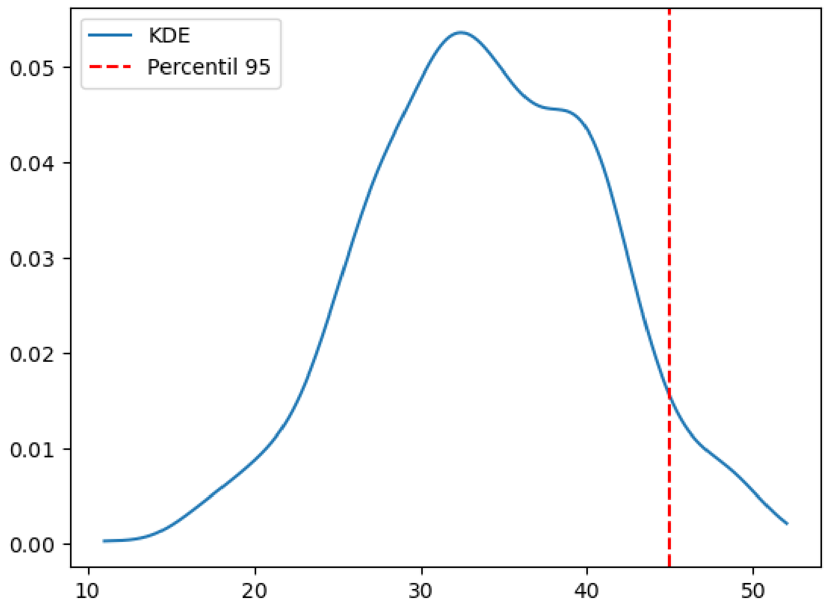

Figura 8 – 95th percentile

In this example, the red line represents the acceptable sales limit where the number of cases to the right equals 95% of cases. The blue line represents the Kernel Density Estimation, which is a way to obtain a probability distribution from a dataset.

4 Training Methodology with Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNN) are a neural network architecture specifically designed to handle sequential data. This type of network is particularly suitable for projects dealing with historical time series data where each data point has certain dependencies on others. Unlike traditional neural networks that assume independence between inputs, RNNs feature recurrent connections that enable information retention across sequences.

4.1 Project Implementation

For training the RNN, preliminary data preparation was required. The implementation steps included:

1. **Data Generation**: Data was generated through code simulating daily sales over a 4-year period, containing information about sales quantities for specific products across different stores.
2. **Data Cleaning and Preprocessing**: The data underwent cleaning procedures where incomplete or inconsistent records were removed.
3. **Data Splitting**: The data was divided into training and test sets following a 75/25 ratio, ensuring fair model evaluation without overlap between the sets.

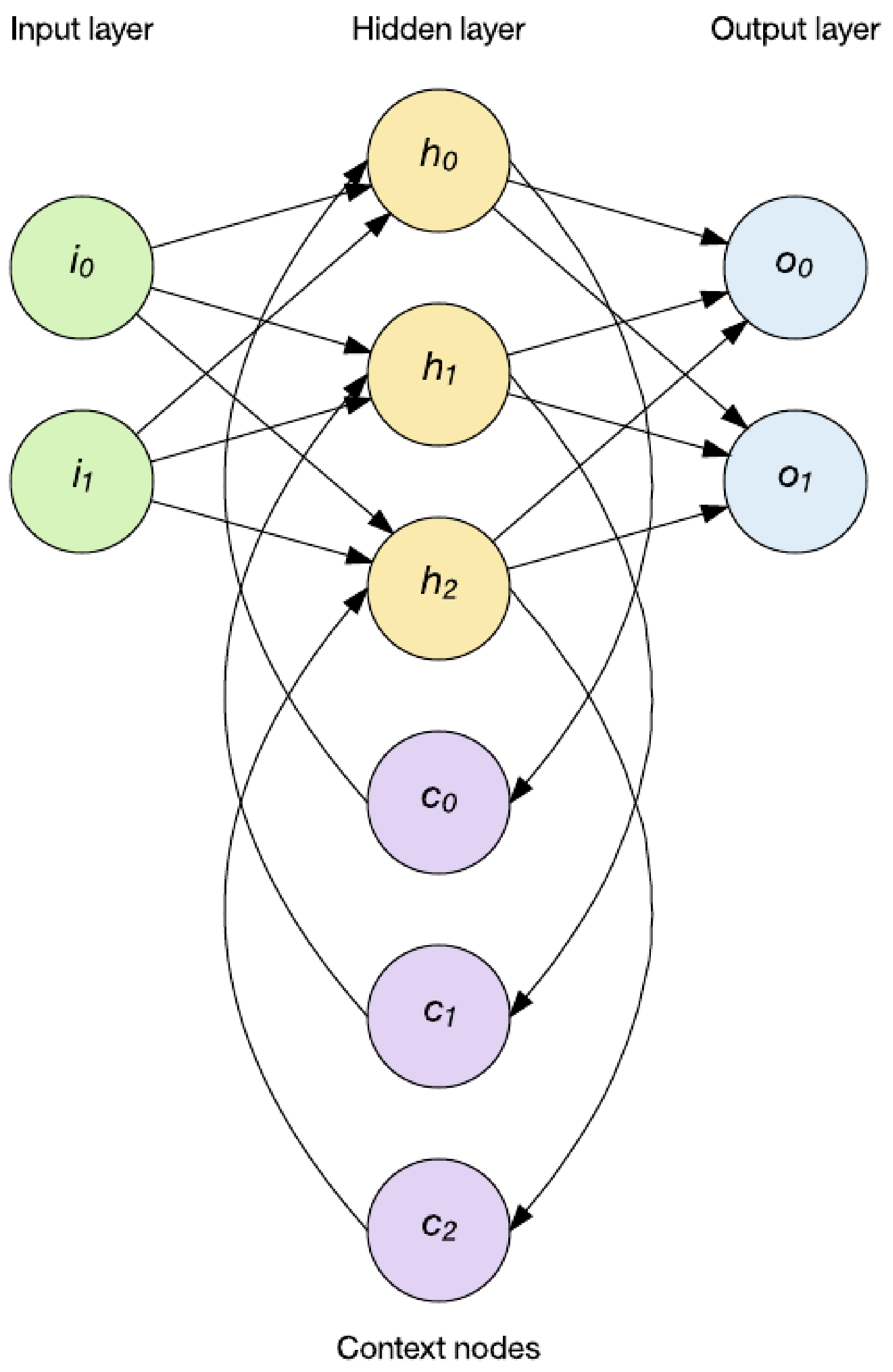

Figura 9 – Example of an RNN architecture (JONES, 2024).

The model training incorporated weekly and monthly moving averages as the observation window size for each day of the year. After training, the model was evaluated using the test set. The RNN demonstrated satisfactory performance in generating predictions for large datasets, considering 4 product-store combinations over a 4-year period.

The use of Recurrent Neural Networks for data training proved to be an effective approach for sales quantity forecasting throughout the year, which would subsequently inform purchase batch sizes and timing decisions. The RNN’s ability to capture temporal dependencies and sequential patterns significantly contributed to prediction generation and project implementation.

5 Algorithm

The following algorithm describes the process for calculating how many products should be purchased each new day:

1. For each new day t:

   a) Calculate the sales forecast for the period from current day t to day $t + P_k$, where P_k is the k-th percentile.

   $$\text{Forecast}(t, t + P_k) = \sum_{i=t}^{t+P_k} \text{Forecast}_i$$

   b) Calculate the total predicted sales for the period.

   $$\text{Total Predicted Sales} = \sum_{i=t}^{t+P_k} \text{Forecast}_i$$

   c) Sum the current inventory with the virtual inventory expected to arrive before the period ends.

   $$\text{Total Inventory} = \text{Current Inventory} + \sum_{j=t}^{t+P_k} \text{Virtual Inventory}_j$$

   d) Check if the difference between total predicted sales and total inventory is less than 0.

   $$\text{Order Requirement} = \text{Total Predicted Sales} - \text{Total Inventory}$$

   e) If Order Requirement > 0, then place an order for the required quantity.

   $$\text{Order} = \max(0, \text{Order Requirement})$$

   f) If Order Requirement ≤ 0, no order is placed.

This algorithm runs daily to determine how many items need to be ordered. Note that it doesn't only consider sales for the analyzed day, since forecasting accuracy improves when considering multiple days rather than just a single data point.

6 Results Analysis

Four products were selected for analysis. The first called $H7$ with a lead time of slightly over one month between order and store delivery, featuring high demand with significant variation. Additionally, product $H4$ has high sales volume and short lead time from supplier to store. Product $P32$ has an average one-month lead time and low turnover, while $P57$ has the highest demand.

6.1 Sales Profile

The recurrent neural network generated the following sales profiles, compared with actual sales data:

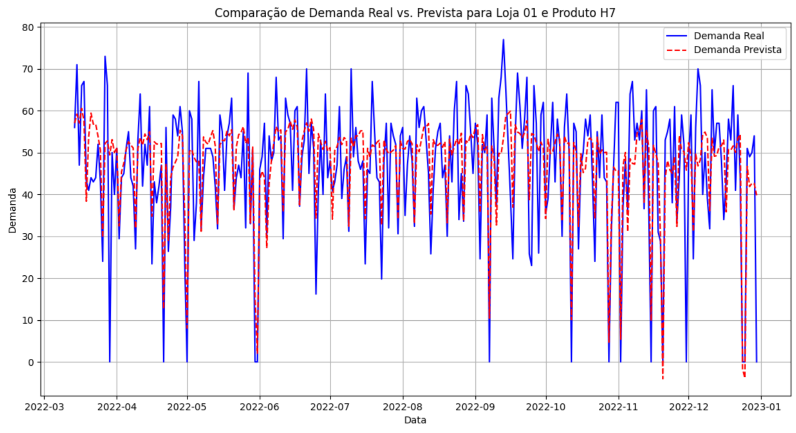

Figura 10 – Sales profile H7

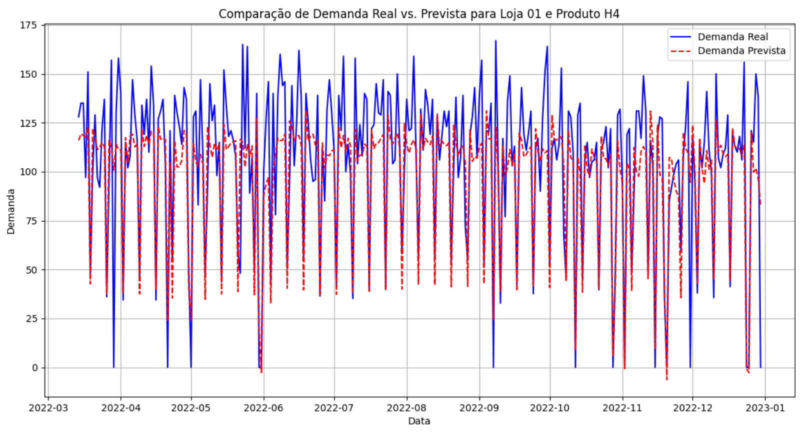

Figura 11 – Sales profile H4

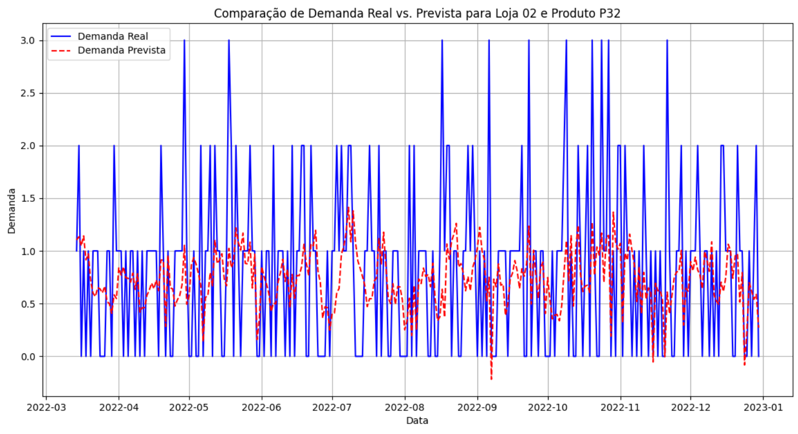

Figura 12 – Sales profile P32

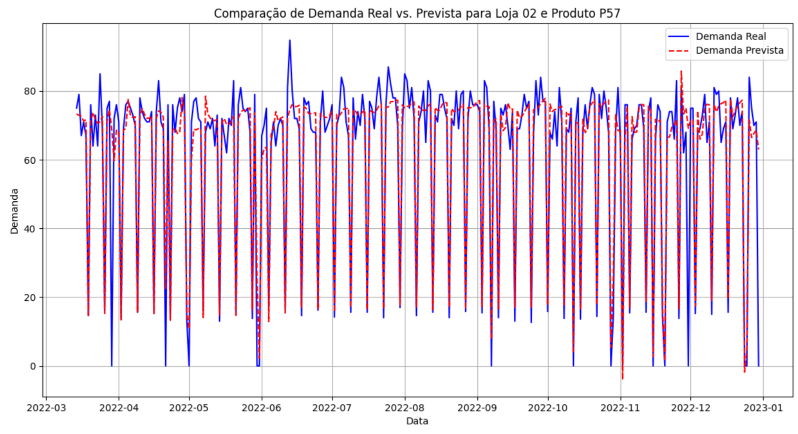

Figura 13 – Sales profile P57

The neural network demonstrates good average accuracy, correctly identifying reduced sales on Sundays and holidays while following the sales patterns with reasonable efficiency. Further parameter tuning could optimize these results. Root Mean Square (RMS) error was calculated for comparison:

Product	**RMS**
H4	23.53
H7	10.58
P32	9.34
P57	0.71

Tabela 1 – Error between actual and predicted sales

6.2 Inventory

The algorithm presented in the previous chapter was applied to analyze inventory for these four products:

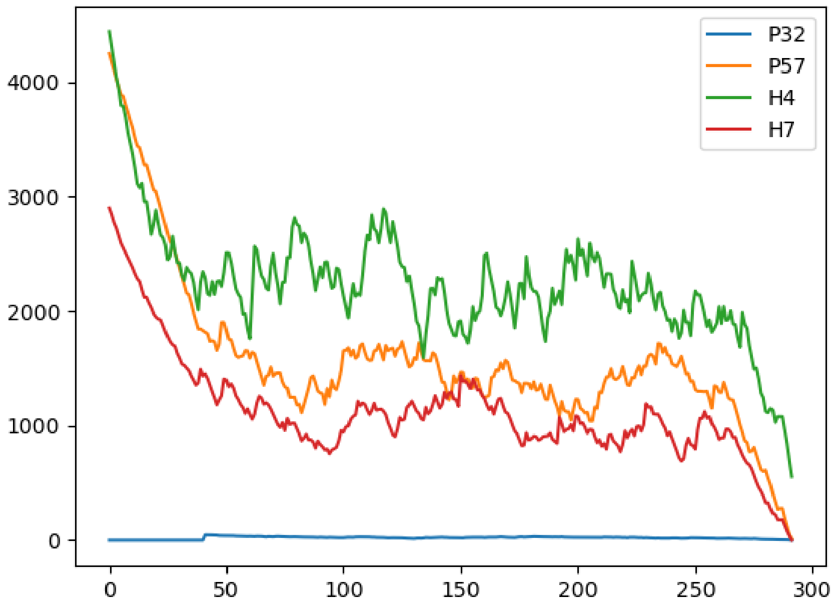

Figura 14 – Inventory profile

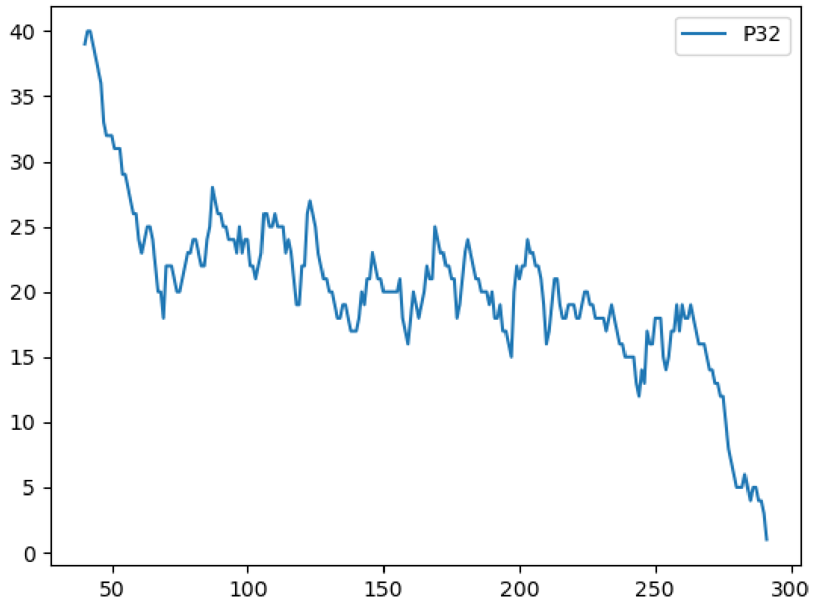

Figura 15 – Inventory profile P32

The initial 40-day transient occurs because the simulation began with a base inventory value (percentile multiplied by previous sales average), requiring approximately one lead time period to stabilize. The inventory decline at the end reflects the forecasting horizon limitation - beyond one lead time, sales predictions default to zero. This wouldn't occur in practice as continuous recalculations would maintain updated forecasts.

While inventory levels appear high, they actually represent only about 5 days of sales coverage. Improvements could come from optimizing neural network parameters and adjusting the P_k percentile threshold. Thirty simulations per product generated these KPIs:

Product	**Lost Sales**	**Units Moved**	**Lost Sales (%)**
H4	0	27,201	0.00%
H7	35	10,624	0.33%
P32	7	237	2.95%
P57	25	13,870	0.18%

Tabela 2 – Inventory KPIs

Lost sales percentages remain below the 5% acceptable error threshold defined for lead time, primarily because the RNN tends to overestimate rather than underestimate demand - particularly noticeable on Sundays/holidays where it recognizes reduced sales but doesn't predict zero. This conservative approach ensures generally reliable forecasts despite occasional overestimation.

6.3 Execution Time Analysis

To evaluate the RNN's feasibility, we collected 100 execution times running 8 parallel trainings (matching available CPU cores). Average execution time was approximately 73.95 seconds.

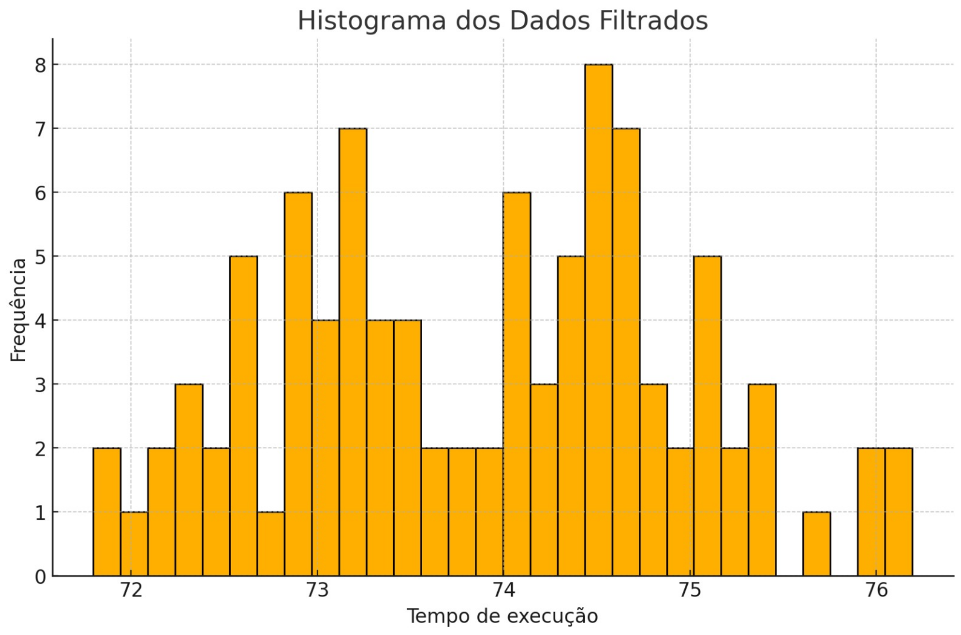

Figura 16 – Execution time histogram

Scalability was assessed by estimating processing time for 100,000 product-store combinations:

$$\text{Time per product-store} = \frac{\text{average}}{4}$$

$$\text{Estimated time for 100,000} = \frac{\text{Time per product-store} \times 100000}{\text{Available Cores}}$$

Converted to hours:

$$\text{Hours} = \frac{\text{Estimated time for 100,000}}{3600}$$

Results showed 63.27, 31.64, and 15.82 hours for 8, 16, and 32 cores respectively. While the test environment (AMD Ryzen 7 laptop) would require weekly rather than daily forecasts at this scale, real-world implementations using clusters could achieve daily forecasts if execution time falls below 14 hours.

In conclusion, while the RNN shows promising potential, large-scale implementation requires consideration of computational resources. The solution becomes viable with robust hardware featuring more cores, representing a manageable cost for enterprises needing daily forecasting capability.

7 Conclusion

This work addressed supply chain replenishment, aiming to determine the optimal timing and quantity to minimize both lost sales and inventory movement. To model store sales profiles, we developed a mathematical model meeting predefined requirements, including lead time modeling from supplier to distribution center and from distribution center to store. Methods such as Bootstrapping were employed to calculate the distribution of product transit time from supplier to store availability, while Recurrent Neural Networks (RNNs) were used to predict demand based on established sales patterns.

The obtained results were promising, demonstrating the feasibility of the proposed model, though requiring parameter adjustments (such as the k-percentile and internal RNN parameters) to improve prediction accuracy. Nevertheless, the project's main objective was achieved by proposing a promising new inventory management model, complemented by temporal analysis validating its practical applicability.

For future work, it's essential to further investigate the mentioned parameters and examine edge cases - such as extremely low-turnover products (e.g., those selling fewer than 9 units per quarter) or products with lead times exceeding one quarter. These studies will help refine the model and enhance its robustness for diverse market scenarios and conditions.

Bibliography

JONES, M. T. **Um mergulho profundo nas redes neurais recorrentes**. 2024. Disponível em: <https://imasters.com.br/data/um-mergulho-profundo-nas-redes-neurais-recorrentes>. Acesso em: 9 jul. 2024. Citado na p. 15.

SOUZA RODRIGUES, F. M. de. **Dimensionamento de Estoque com Base no Padrão de Demanda do Material: Estudo de Caso em uma Empresa do Ramo de Petróleo e Gás**. Jan. 2014. Citado na p. 5.