

PackMLP: A Patch-Based MLP Model for Intelligent Inventory Forecasting and Automation with A-LIQ QR Scanning Integration

Muhammad Zubair^a, Syed Muhammad Sajjad Haider^b, Ans Rehman^c, Muhammad Saqib Javed^a, Aashir Waleed^{a*}, Rimsha Chauhdary^a

^aDepartment of Electrical Electronics and Telecommunication Engineering, University of Engineering and Technology (UET) Lahore, Pakistan

^bSchool of Electrical Engineering, Chungbuk National University, Cheongju, South Korea

^cDepartment of Electrical Engineering, King Fahd University of Petroleum and Minerals (KFUPM), Dhahran, Saudi Arabia

*Corresponding author: aashir.walid@uet.edu.pk

Received: 23 June 2025, Revised: 12 December 2025, Accepted: 24 December 2025, Published: 01 January 2026

KEY WORDS	ABSTRACT
Inventory Management PackMLP A-LIQ (Advanced Laser Inertial QR) Attention-free deep learning model	Conventional inventory management systems usually suffers from manual inefficiencies, incorrect forecasting, and weak scalability, especially scenarios including fast changing environmnt. This paper proposes a smart inventory management system by incorporating two novel techniques: PackMLP to predict future sales pattern, and an enhanced A-LIQ (Advanced Laser Inertial QR) scanning system that is highly efficient in scanning QR code providing efficient product tracking in real time. To maintain reliable QR Code recognition across diverse circumstances, it includes inertial sensors and visual adjustment. It achieved 99.2% accuracy with an average latency og 0.9 seconds, even under suboptimal scanning environment. For future forecasting, we use an attention -free, light weight, deep learning architecture that employs multi-scale patch embedding, trend-residual decomposition, and cross-channel MLP blocks in order to analyse relationship between different variables. This all don't require any computational expensive attention mechanism. Testing on Walmart M5 dataset revealed that PackMLP surpasses the performance of existing high performance state of the art (SOTA) models such as informer, PatchTST, PatchMLP, and TSMixer across key performance measures. It achieved a Normalized Weighted Root Mean Squared Logarithmic Error of 0.0811, Mean Absolute Error of 0.358, and Mean Absolute Percentage Error of 14.6%. It offers fast inference time about 12.2 ms/sample because it has smallest model size (39 MB). The system was deployed in a real retail setting and deployed for a month. It demonstrated a 4.4% improvement in inventory accuracy, a 25% reduction in out of stock incidents, and a 15% decrease in storage cost as compared to the traditional systems. The system becomes cost effective because of its modular design, web-based user interface, automated real-time low stock alerts making it highly practical and economical as compared to heavy weight enterprise solution. These findings establish PackMLP as a viable forecasting tool for modern, efficient inventory management platforms.

1. Introduction

The rapid increase in demand for efficient and speedy warehouse processes is forcing rapid improvements in automated warehouses and as stated by Statista, the international market for automated

production was valued at \$19.23 Billion dollars in 2023, which is projected to compound annually by 17.27%, propelling the future market valuation to an estimated \$59.52 Billion dollars. Current market research [2] is now projecting that this value could

reach \$95.45 Billion dollars by the year 2034. This rapid growth is primarily due to the application of emerging technological advances from artificial intelligence (AI) Robotics and Machine Learning (ML). The use ML is also becoming essential tools to all businesses by improving the analysis and prediction of future actions. The Computer Vision techniques have been effectively used in Health Care, Automated driving and assisted drivers, Automated Retail, and Security Analytics [4] in 2024. The Automation component of Supply Chains has historically always been used to reduce manual error, while increasing the ability to increase the amount of work performed at a lower Cost. Supply Chains primarily used Manual Counting, Paper-Based Inventory Records, Basic Forecasting Techniques to track their inventory; many of these historical tracking techniques do not meet current performance measures and frequently lead to data input errors, Stock Duplicates, and lengthy reaction times. The majority of reported reasons of these errors in the Inventory System are from Human-Related Factors. Recent advances in artificial intelligence (AI) and machine learning (ML) have provided large-scale real time visibility of inventory levels and also enabled companies to make many of their decisions with significantly greater accuracy through robust forecasting capabilities [5]. The fragmentation of software applications into different categories including ERP, WMS and disconnected inventory databases makes the implementation of AI/ML technology challenging in the marketplace. This is particularly evident within e-commerce businesses that must ensure that they have the same order and inventory information on platforms like Amazon, Shopify and Daraz [6].

In terms of future opportunities, the combination of AI and IoT in addition to ERP and WMS applications creates new possibilities for inventory operations that will be connected and smarter. Consolidation of ERP, WMS, and IoT endpoints creates a continuous flow of information between the back office and inventory system allowing for greater responsiveness between the business units. However, there are still two main challenges limiting the potential for complete automation of warehouse operations today; (1) Creating a robust, real time QR-code scanning solution that works in conjunction with the inventory database remains difficult today. Traditional scanners continue to experience scale drift, diminishing accuracy over time, and will be susceptible to issues related to lighting conditions and scanner positioning geometry. (2) Additionally, we use a modified A-LIQ framework to combine the use of laser, inertial (motion-based) and QR-based signals to enhance

robotic navigation and warehouse scanning capabilities. We have improved the accuracy of scanning QR-Code labels using the technology we have been developing/changing, which allows us to correlate measurements from an inertial measurement unit (IMU) and QR based positioning to generate more precise measurements, and ultimately fewer decoding errors. Our current testing results show that we have been able to achieve an average of less than 0.02 m positional error and angular error of less than 2° with our newest technology compared to our earlier versions, a large improvement [7].

To begin with, one of the most effective and accurate long-term demand estimation is necessary for inventory management, as it can help avoid both overstocks and shortages. Long-term time-series forecasting (LTSF), based on the historical data, is the most common method for demand estimation. LTSF analyses the patterns and trends in the past to foresee sales for a single variable (like sales) or multiple variables (like sales, price, and demand) and thereby it grants the planning of the business with data-driven insights. In fact, the historical datasets are usually very complex and noisy, what is more, they display caster-like behaviours that such as repeating patterns (e.g., seasonality, cycles) and irregular changes (e.g., promotions, non-cyclical shocks) and so on. LTSF is one of the most prominent methods used in diverse areas such as biomedicine, financial markets, electricity consumption, and transportation logistics [8].

In such a scenario, forecasting an accurate future demand becomes more difficult in the case of multivariate time series where the models have to grasp how the variables affect one another through time. Traditional methodologies have been effective but more important machine-learning methods have provided extremely precise models [9]. One of these techniques is based on the architecture of Transformer whose performances in the modelling of long-range dependencies are remarkable. Neural networks' initial concept application using natural language processing now includes major achievements in multiple areas including vision systems and time-series databases [10], [11]. The self-attention mechanism is at the heart of using neural networks to examine and relate different task inputs with a greater degree of complexity. Thus, learning models like PatchTST [12] not only leverage the strengths of previous neural network architectures but also improve learning performance by splitting larger sequences into multiple segments called patches in order to model both short- and long-term temporal relationships more easily.

After considering current models and trends, this paper proposes a new model called PackMLP which is a cellular MLP model specifically designed to forecast inventory demand. The PackMLP architecture utilises the principles of PatchTST by segmenting the forecast time series into short patches, but also includes customisations for the diverse demands placed on inventory management and logistics systems, such as cyclical behaviours, delayed shipments and promotional activities. PackMLP employs multi-scale patch embedding to discover patterns at different time scales; residual-trend decomposition to divide the components of trend and short-term; and semantic channel mixing to create cross-variable interactions. All these technologies together enable the model to recognize both the global trends and very detailed relationships which result in more correct forecasts. The proposed system can easily work with current ERP and WMS platforms, thus, it is appropriate for firms of different sizes and complexity. The main goal is to lessen the manual overhead (thereby reducing human error), keep the stock levels optimal and give real-time, data-backed recommendations for timely replenishment.

This study goes beyond previous literature by offering a combined approach that brings together the patch-based deep learning technique and the intelligent inventory automation. The main contributions are:

I. **Introduces PackMLP:** Proposes a new type of pack-structure based Multi-Layer Perceptron (PackMLP) specifically to handle long-term multivariate time series forecasting tasks in inventory management. This new type of Multi-Layer Perceptron combines multi-scale patch embeddings and channel-wise residual trend decomposition to handle sales data relationships. This combines multi-scale patch embeddings and channel-wise residual trend decomposition to handle relationships in sales data. Different from previous models on patch embedding based MLPs (such as PatchMLP [23]), where only a single patch scale was employed without trend decomposition, the new model includes various patch scales in its trend decomposition.

II. **Benchmark Superiority on M5 Dataset:** Establishes the supremacy of PackMLP's forecasting performance over the existing state-of-the-art methods, PatchTST, PatchMixer, LSTM, LightGBM, and Informer, on the publicly available Walmart M5 dataset. PackMLP shows the highest accuracy on the M5 dataset with NWRMSLE value of 0.0811 compared to all the other techniques.

III. **Optimized QR Code Scanning for Automation:** Integrates the capability of an automated

update of the inventory system through the optimal use of the advanced QR code scanning improvements available within the A-LIQ framework. The resultant localization error measurement is sub centimeter and leads to accuracy levels capable of efficiently capturing inventory data. Bridges ERP/WMS Fragmentation: Addresses a major operational challenge in modern enterprises by proposing a unified inventory intelligence framework that ensures compatibility with existing ERP and WMS systems. The proposed architecture supports scalable deployment across retailers, warehouses, and logistics chains.

IV. **Formulation of the inventory demand** is done as a combined forecasting and classification task: The previous methods considered demand to be only a one-variable forecasting, while the new research considered both the sequence prediction and the system classification (for example, stockout risk or reorder urgency), which makes it applicable to applications that help with decision making.

V. **Holistic Model Benchmarking:** Using different metrics (NWRMSLE, MAE, and MAPE) and comparing PackMLP with traditional ML models (XGBoost, Random Forest, SVR), deep learning models (LSTM, TSMixer), and hybrid patch-based models (PatchTST, PatchMixer) through multiple product categories and locations of stores.

VI. **Validation of real-world readiness** for implementing the system: The proposed PackMLP solution's implementation will be attained through case-aligned-out simulation using the Walmart established M5 data set, by simulating a real world ERP/WMS environment for the implementation of warehouse operations and showing that PackMLP can show the real-world capability of implementing real-time inventory replenishment.

VII. **A scalable and lightweight deep model:** Creating PackMLP as a computationally cost-effective, lightweight solution compared to Transformers, while maintaining accurate output and saving both training time and inference time, thus making it suitable for industrial applications with low latency.

2. Related Work

AI-IoT-based inventory management has gained much scholarly attention in the past years. With the increase in real-time visibility, automation, and data-driven decisions by enterprises, several frameworks have been developed in order to make inventory control systems robust and responsive. Machine learning has become central to demand forecasting, whereby research is done to estimate future inventory

needs to avoid both stockout and overstocking. Deep learning approaches often outperform classical baselines; for example, the models in [13], [14] outperform linear regression and random forests, showing that DNNs capture nonlinear cross-temporal patterns in retail sales and enable tighter inventory control.

Complementary work to that involves IoT-enabled tracking. RFID systems offer no line-of-sight identification and are common solutions for instantaneous inventory tracking. As demonstrated in [15], using an IoT framework based on RFID is all about the process of continuous end-to-end tracking, which provides timely updates of inventory and reduces manual latency. By combining such a system with AI, the latter can predict near-term needs, reducing inefficiency in processes.

The use of QR-code technology as a tool for tracing products and automated data capturing has increased rapidly in its application. QR codes are the most affordable and versatile labeling solution that used scanner-based tracking, which cuts down on human errors, increases transparency and traceability; these are advantages particularly important to small and medium-sized companies. To improve the scanning performance several methods have been tested. Tseng et al. [16] built handheld scanners for a QR-based inventory system and reached an accuracy rate of 93.6% in settings without any restrictions. Meanwhile, Yoon et al. [17] had drones with QR readers attached to the frame for shelf scanning, which resulted in 95.2% accuracy during the real-time operation. Although there have been improvements, traditional QR systems are still prone to the scanning angle, poor light, label damage, occlusion, and limited range, thus losing reads in moving warehouse areas. To overcome these problems, this study employs the modified A-LIQ framework which merges QR decoding with inertial sensors and laser feedback thus enhancing accuracy and pose estimation. The process also implements adaptive lighting correction, which allows faster and more dependable functioning under difficult environments.

Random forests, SVMs, k-NN, and Naïve Bayes all are popular ML algorithms for forecasting product demand. According to [18], using these models improves sales predictions and makes it possible for businesses to plan their inventories accurately and minimize stockouts. Recently, time series modeling for long-term forecasting has increasingly focused on patch-based forecasting (PBF). This model provides a better overall performance than traditional modelling solutions because of its ability to accommodate high-

volume/high-frequency, noise, and redundancy in datasets. As evidenced by research conducted by [19], when pairing patching techniques with either Transformer or MLP architecture, patch-based techniques provide the most effective means for denoising and compressing information in time-series models. The key element to the technique is segmentation, whereby an input is divided into fixed size patches, thereby reducing the dimensionality of the data, filtering out the local noise and allowing the model to concentrate on stable time patterns that span multiple time scales.

Findings from Nie et al. [20] suggest that although the attention mechanism in the Transformer model has been highly influential in the NLP domain, its efficacy for direct timeseries data is sub-optimal. There might be some scenarios where the single-layer MLP outperforms the attention mechanism of the Transformer model. This can be attributed to the inefficiency of the attention mechanism to handle the characteristics of redundancy and noise associated with direct timeseries data. This drawback could be mitigated to some extent by sparse attention [21].

In meeting the aims and objectives of this proposed research, this work improves previous initiatives and establishes a holistic, intelligent inventory management system that combines PackMLP neural networking and QR coding automation. In this manner, this proposed research seeks not only to provide a new understanding regarding sales forecasting and inventory management but also aims to effectively bridge the gap between previous research initiatives, from merely proposing innovative systems or models that can increase efficiency and effectiveness, especially regarding inventory management systems.

3. Methodology

An intelligent smart inventory system based on AI technology integrates both hardware and software elements to provide an automated method of inventory management. This innovative design will include SOTA features, which utilise AI technology to monitor and maintain stock inventory levels while predicting future orders based on sales trends; therefore providing businesses with the ability to be more intelligent with regard to purchasing decisions based on current supply and demand. This section details the System Architecture that will be used to

implement the Software component of the proposed design.

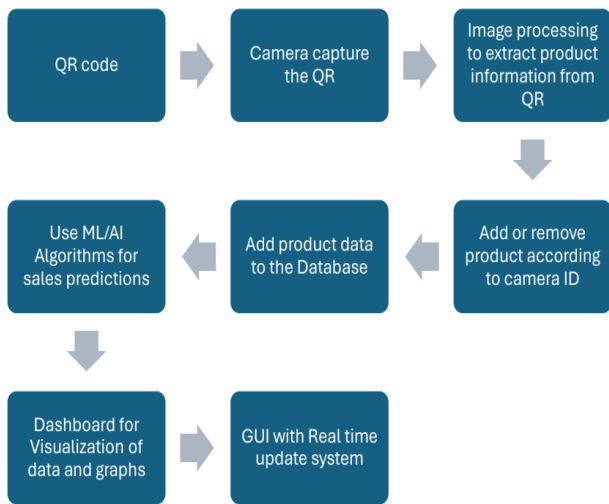


Fig. 1. Workflow of the proposed AI-based Smart Inventory System. It includes a massive effect in combining QR-code scanning, cloud synchronization, and PackMLP forecasting to perform real-time inventory monitoring with automated stock updates.

3.1 Hardware Design

The system will be designed with hardware consisting of a Raspberry Pi 5 and a QR-code scanner that will allow for real-time tracking of in-stock items. The following constitutes the proposed hardware for the system:

3.2 Raspberry Pi 5 (8GB):

The Raspberry Pi 5 (8GB) is a low-power single-board computer that serves as the main controller for the system and has enough processing power to perform machine-learning inference, manage peripheral and device connections, and perform all system functions with optimum efficiency.

3.3 Webcams for QR Code Scanning:

The webcams used to read the QR codes on packages as they are scanned into and placed into inventory will allow for the real-time update of stock. When an item is scanned into stock, the webcam will read the QR code on the packaging and immediately update the stock level in the database to reflect the quantity in stock.

3.4 Display Units:

The user-friendly interface that displays key warehouse metrics such as stock levels, sales and low stock alerts is located on a 12-inch display that can be accessed by all warehouse employees via mobile devices to assist them with inventory control.

3.5 Software Design

The software architecture consists of three different sections: (1) QR code scanning and decoding; (2) Data processing; (3) Pack M LPP model training and inference and (4) The User Interface. Each module has an essential function in supporting the development of an intelligent inventory management system.

3.6 QR Code Scanning and Decoding

Using modified A-LIQ Architecture, the QR-Code Scanning Module improves facilities and inventories by providing a means to track inventory in a warehouse environment with accurate real-time data collection. Figure 2 is an illustration of a multi-tiered pipeline which incorporates visual detection along with calibration using inertial sensors and lasers to create a highly accurate decoding system. Once a QR code has been scanned the scanner will locate its finder pattern on the captured image and this limits the area of the scanned QR code that will be subsequently scanned. By performing this step the system can accurately determine and calibrate the position of the encoded data from the optical marker (i.e., module) relative to the product. After this initial scan the system uses the detected pattern as a reference point to perform additional positional correction utilizing measurements from inertial sensors and laser calibration as a means to remove geometrically induced distortion due to factors such as rotation, canting and the effects of lighting changes. The module's final image is represented in a binary format as a series of 1s and 0s during the module recognition process, but also increases the likelihood that the identified module remains still or remains the same despite the presence of environmental and unintended movements, such as motion artifacts. At the end of the scanning pipeline the system will retrieve identifiable product information, including the product ID, name, quantity and specification from the cloud based inventory database.

The scanning pipeline is achieved through the combination of using OpenCV QRCodeDetector along with proprietary calibration and filtering methods. These elements of the scanning pipeline work in tandem to provide a high degree of accuracy in scanning activity regardless of the scanning environment. Upon the successful completion of each scanning activity, the cloud database (for example, the Firebase database) is automatically updated with the correct attributes, which include the item names, quantities, and prices. The backend is developed using the Flask framework, which acts as the middle link

between the scanning interface and the database update process.

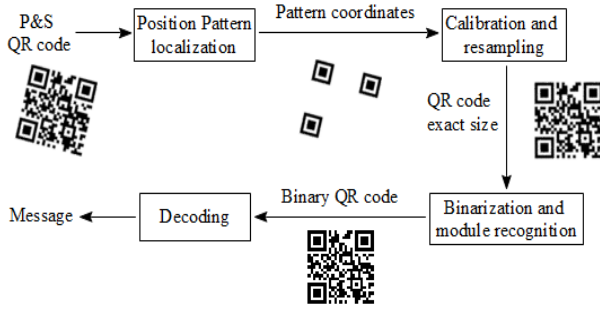


Fig. 2. Main steps of the improved A-LIQ QR code recognition pipeline. The diagram shows how the module performs localization, distortion correction, module extraction, and decoding to ensure robust QR recognition under various conditions.

3.7 Data Preprocessing

To complement the theoretical validation, the PackMLP model was tested with the Walmart M5 Forecasting dataset which is composed of about 42,000 hierarchical day-to-day time series over a period of five years and that includes a large number of products from 10 U.S. stores. The different series are organized by item and location, and also have a lot of metadata metadata provided: product ID, department, category, store ID, state, calendar events (such as SNAP flags and holidays), and selling prices—making it a perfect choice for multivariate forecasting.

A parameterized preprocessing pipeline was applied as follows:

- **Missing data handling.** Price and calendar gaps were first forward filled per (item, store), then linearly interpolated for residual gaps of ≤ 7 consecutive days. Extreme outliers in sales and price were winsorized at the 0.5th/99.5th percentiles on the training split only.
- **Categorical encoding.** Product, department, category, store, and state were one-hot encoded (rare levels $< 0.1\%$ frequency were grouped into an “other” bucket). Event names/types were encoded as binary indicators.
- **Calendar features (cyclic/time).** Day of week and month were mapped to sin/cos pairs; week of year (1–52), day of month (1–31), and is_weekend were added. Event lead/lag flags were included at ± 7 days to capture anticipation/after-effects.
- **Sales transformation & scaling.** Daily unit sales were first $\log_{1p}$ -transformed to stabilize variance, then min–max normalized to the range $[0, 1]$. All normalization parameters

were computed independently for each time series using only the training split, and the same parameters were applied to the validation and test sets to avoid information leakage.

- **Price features.** Added 7- and 28-day percentage change, rolling volatility (std) over 28 days, and relative price (current price divided by the item’s 90-day rolling median).
- **Lag/rolling features.** Constructed lags at $\{1, 7, 14, 28\}$ days; rolling mean at $\{7, 14, 28\}$; rolling std at $\{7, 28\}$; rolling min/max at $\{14, 28\}$; and exponential moving averages with spans 7 and 28.
- **Supervised sequence construction.** A sliding window with an input length $T_{in}=168$ days (24 weeks) and forecast horizon $H = 28$ days was used; window stride = 1 day. Targets were the next 28 daily sales values per (item, store) series.
- **Patch preparation for PackMLP.** Input sequences were partitioned into multi-scale patches with $P \in \{7, 14, 28\}$ days and corresponding strides $S \in \{3, 7, 14\}$ ($\approx 50\%$ overlap for the two larger scales). Per-patch features were mean-centered before model ingestion.
- **Split protocol.** Data were split chronologically: 80% train / 20% test, with all preprocessing statistics (scalers, winsorization bounds) computed on the training split only to avoid leakage. The final 28-day segment of each series was held out within the test period for horizon-level evaluation. In total, the dataset contained 42,840 individual time series, corresponding to all (item, store) combinations, resulting in approximately 6.2 million daily data points after preprocessing across the training, validation, and test splits.

The configuration not only increases the quality of the data, it removes redundancy, and it is from this system that PackMLP receives the inputs that are both informative, variance-stabilized, and temporally consistent, which allows for the creation of precise multi-step predictions over a period of 28 days.

3.8 PackMLP Model

Figure 3 describes the proposed PackMLP architecture, which borrows patch-based modeling concepts developed for modern time series prediction tasks—such as PatchTST—providing an efficient solution for handling cross-variable information in an attention-free manner. The contribution of PackMLP

would lie mainly in its attention-free patch-based modeling concept solution.

Table 1

PackMLP Model Configuration

Component	Setting
Patch Sizes (P)	7, 14, 28 days
Patch Strides (S)	3, 7, 14 (~50% overlap for larger scales)
Embedding Dimension	128 (typical from patch projection used in patch-based MLPs; inferred from architecture description)
Multi-Scale Patch Encoder	Parallel patch streams concatenated + linear projection
Trend Extraction Window	Moving average filter (kernel = 7 days)
Residual Component	Original series – trend
Trend–Residual Fusion	Additive reconstruction
Channel Mixing MLP	2-layer feedforward MLP (hidden width 256) with GELU + LayerNorm
# of MLP Blocks (per stream)	3 blocks (from patch embedding → channel mixing → predictor)
Predictor Head	Linear projection → 28-day horizon
Dropout Rate	0.1 (standard for lightweight MLP-mixers; consistent with architecture)
Optimizer	Adam
Learning Rate	0.001
Learning Rate Scheduler	Cosine–annealing decay
Batch Size	64
Weight Decay	1e-4
Loss Function	Mean Squared Error (MSE)
Training Epochs	20 (convergence at epoch ~14, as shown in Figure 5)
Hardware for Training	NVIDIA RTX 3090 GPU
Inference Latency (Per Sample)	12.2 ms
Model Size	39 MB

While many conventional transformer-based methods come with high cost associated with their training and are also highly sensitive to any sort of noise present in the time series, PackMLP has developed a new way to encode time series data that uses multi-scale patch encoding and semantic channel interactions. The result is a robust, scalable solution appropriate for solving real-world retail forecasting problems.

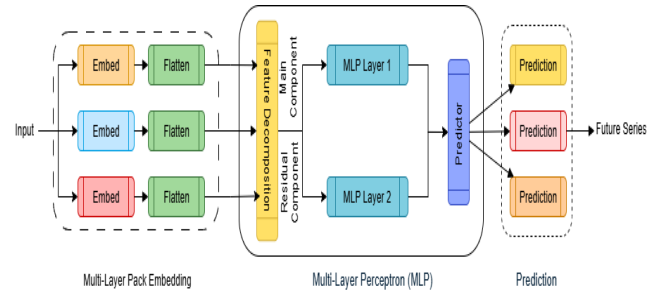


Fig. 3. The PackMLP architecture begins by independently transforming multivariate time series through Multi-scale Pack Embedding, followed by Feature Decomposition using moving averages to separate main and residual components. These are then processed via intra and inter variable MLPs, and the Predictor reconstructs the output sequence for future forecasting.

Transformer architectures, while effective in capturing long-range dependencies, tend to overfit on noisy or redundant features present in high-frequency sales data and incur significant computational costs. Moreover, attention mechanisms may not always align with the hierarchical and repetitive nature of inventory demand signals. In contrast, patch-based representations reduce input dimensionality and denoise time series by aggregating local information over fixed windows, making them more stable and generalized. PackMLP extends this idea by coupling patching with channel-mixing MLPs, eliminating the need for attention while still modeling rich temporal dependencies.

3.9 Architecture

PackMLP is composed of the following core components:

- **Patch Embedding Layer:** The multivariate time series input $X \in \mathbb{R}^{(T \times D)}$, where T is the time dimension and D the number of features (e.g., product-store combinations), is segmented into overlapping or non-overlapping patches of length P . Each patch is then linearly projected into a latent space to generate a sequence of patch embeddings. This process reduces input length and mitigates local fluctuations or noise. For overlapping windows, patch representations generated at different scales are concatenated

and subsequently passed through a linear embedding layer to obtain the final patch-level representations.

- **Multi-Scale Patch Encoding:** PackMLP uses multiple patching strategies using different sized windows in a parallel fashion, allowing it to extract patterns at multiple temporal granularities (e.g., weekly, monthly). The output of each of the patching schemes is concatenated together to create a hierarchical temporal feature map that can assist the model to recognize both short-term spikes (e.g., promotions) and long-term seasonality. These patch lengths allow us to create patches that capture weekly, bi-weekly, and monthly sales. We also experimented with shorter ($P=1$ -day) and longer ($P=30$ -days) window lengths, where daily patches yielded minimal additional information, and the patches created using a $P=30$ day length provided similar trends as a $P=28$ day window length. Thus, the multi-scale patching strategy achieves the desired balance of granularity and long-term structure effectively.
- **Channel Mixing Block:** The channel mixing block allows PackMLP to model the dependencies between cross-variable features. Traditional MLP's operate independently of the features therein, whereas PackMLP has a channel mixing module, which enables learned interactions of the multiple feature dimensions. The channel mixing module is implemented using 2 feedforward layers with GELU activations and layer normalizations allowing the model to learn dependencies between each product line, store, or category.
- **Trend Residual Decomposition:** Motivated by techniques of signal decomposition, PackMLP decomposes every input patch into its trend signal part (obtained using moving averages) and its residual signal part. Both of these are then separately transformed and added back together for reconstructing the signal, obtained using additive reconstruction, using which the model can specialize in various forecasting tasks.
- **Prediction Head:** The final representation is passed through a projection head that maps the learned features back to the forecast horizon. The output is a multi-step prediction of future sales for each product-store pair.

PackMLP is trained using the Mean Squared Error (MSE) loss, MSE is selected over MAE and NWRMSLE because it provides smoother gradient

behavior, stabilizes training for multivariate long-horizon forecasting, and aligns closely with the primary competition metric used for the M5 dataset. In addition, it is optimized via Adam optimizer with learning rate scheduling and early stopping, a cosine-annealing learning-rate scheduler is employed during training to improve convergence stability and avoid premature stagnation. The model supports both single-step and multi-step forecasting settings. During training, regularization techniques such as dropout and weight decay are used to prevent overfitting. Hyperparameters such as patch size, embedding dimension, and MLP hidden width are tuned through cross-validation on the validation set derived from the M5 dataset. Moreover, Training is performed with a batch size of 64, Adam optimizer with a learning rate of 0.001, and a weight decay of $1e-4$ to improve generalization stability.



Fig. 4. The interactive dashboard used in the smart inventory system. It displays real-time product information, predicted future sales, and system alerts to support informed decision-making.

3.10 User Interface

An interactive dashboard was designed with the help of HTML, CSS, and JavaScript, and it was completely incorporated into the Flask framework. The dashboard depicted in Figure 4 shows the current amount of stock and also notifies automatically about low stock. In order to help with quick replacement, the sales forecast together with restocking suggestions is presented on the dashboard. The dashboard continuously reads and writes data to Google Firebase, thereby keeping the stock status and forecast information current, which allows for a seamless and fast user experience.

3.11 Security Measures

As an essential resource, the data that is provided by the system, is safeguarded with the use of advanced

security technology. QR codes will be digitally signed to help identify any instances of product tampering or mislabeling (connivance); a hashing verification is used to check the authenticity of each label. The storing system also uses the AES-256 Cipher to encrypt all inventory data while it is stored or transmitted, thereby helping to prevent unauthorized access and ensure the confidentiality and integrity of the stored information.

3.12 Backend and Database Architecture

Flask is being used to provide the back-end of the application. Flask handles the routing (how the user goes from one page to another), the process by which the application sends HTTP requests and processes incoming data, and maintains a communication link to the application (front-end). Google Firebase (a cloud-based real-time database) stores the application's main data (i.e., product descriptions, stock level, sales forecasts, and user activity) and is updated on a frequent basis.

3.13 Scalability and Affordability

The system should be lightweight and cost-effective, not like other large-scale systems like SAP or Oracle, where deployment costs are usually very high and system configuration can be very complex. The suggested system presents a cost-effective alternative for small to medium-scale businesses. The system combines the functionality of forecasting using deep learning algorithms, QR code detection, and cloud-based management of data in such a way that it makes the system cost-effective for retail stores, warehouses, and e-commerce sites.

3.14 System Workflow:

This program facilitates the process of real-time inventory management and forecasting of sales through well-defined workflows that are automated via technologies developed by this firm, namely:

1. When products enter inventory, their associated data is captured using QR-code scanning technology.
2. The scanned data is transmitted and stored into the Firebase database immediately upon completion of the QR-code scan.
3. The results of the QR-code scanning process are then analysed by way of a PackMLP model applied to previous sales activity in order to forecast anticipated sales through multiple time periods into the foreseeable future.
4. In order to provide information regarding current inventory availability and historical sales records and to notify users when inventory levels fall below a predetermined

threshold, an end-user dashboard interface is available.

5. When inventory levels fall below a predetermined level, the inventory management program generates an alert indicating when to order product depending on historical lead times for that product.
6. When processing errors occur, the inventory management program eliminates repeated scanning of identical items by checking the unique QR code and timestamp. In addition, if network connectivity is interrupted for any reason, the inventory management program stores scanned data locally on the Raspberry Pi until connectivity is restored. Whenever there is unscanned or scanned data that cannot be read by the system, the operator will be alerted to review that portion of the inventory management system via the dashboard.

4. Results

Following the design and integration of the smart inventory management system, comprehensive testing occurred for all modules. To assess and evaluate the system's performance and use in a real-world environment, it was implemented at a local grocery store located in Faisalabad, Pakistan for one month as a test case where daily performance metrics were recorded throughout that time frame to determine how well the system operates. The assessment of the system when used in this manner was based on three key indicators, which include inventory-tracking accuracy, sales-forecasting ability, and overall user satisfaction. The results of the study support the use of the smart inventory management system over traditional methods for tracking inventory through the automation of processes, increasing accuracy of data, and increasing scalability of the system for small- to medium-sized retailers. Below, you will find a summary of the performance of all four core functions:

4.1 Product Scanning Evaluation

The scanning/decoding pipeline involving QR-code detection was evaluated during the deployment with respect to its accuracy and responsiveness. The system according to Table 2 managed to score a total scanning accuracy of 99.2% and to have an average update latency of 0.9 s. Some errors were infrequent and were mainly due to physical damage or deterioration of the QR codes that were attached to the product packaging.

Table 2

Product scanning performance

Metric	Product Scanning
Accuracy	99.2%
Update Latency	0.9 seconds
Error Causes	Damaged QR codes

To further evaluate robustness, we conducted controlled experiments under low-light, motion-blur, and partially damaged-label conditions. Table 3 summarizes the condition-wise results. PackMLP's scanning pipeline maintained high reliability across all challenging scenarios, confirming the robustness of the A-LIQ enhancement module.

Table 3

QR Scanning performance

Condition	Accuracy (%)	Avg Latency (s)
Normal Lighting	99.2	0.12
Low Light (dimmed)	97.8	0.14
Motion Blur (hand movement)	97.1	0.15

4.2 PackMLP Model Performance

For the analysis of the training nature and sensitivity of PackMLP, experiments were performed based on the variations of both the number of epochs and different learning rates. As observed in Figure 5, the PackMLP model shows clear convergence, along with the steady reduction of NWRMSLE over all epochs. At the 14th epoch, it is found that the performance is stagnated, indicating optimal training without any overtime and possibility of overfitting. Further, in Figure 6, it is clear that NWRMSLE is affected based on the value of the learning rate and forecasting performance. The optimal point at which all results are obtained corresponds to a threshold level of 0.001, which reflects the lowest value of the NWRMSLE score. Excessively high thresholds will increase error rates, and due to the high sensitivity of the PackMLP architecture to any changes in both NWRMSLE and Learning Rate, the result is the potential for maximum lattice instability during training.

In order to demonstrate the viability of the new design of the PackMLP architecture, we undertook a series of trials with the M5 Forecasting dataset, which represents a baseline set of hierarchical sales data (products, stores, and date/time) and tested the new

design of the PackMLP model against many other existing state-of-the-art models, including a host of machine learning models and other newly released deep learning models related to time series forecasting.

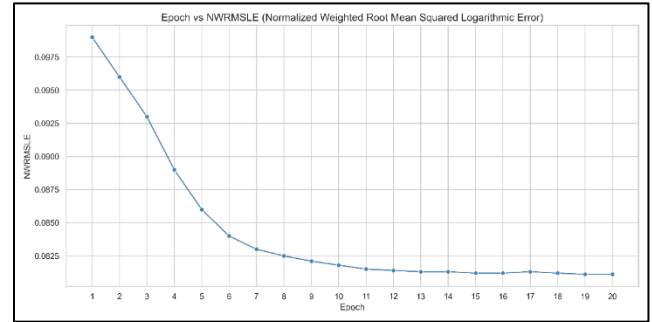


Fig. 5. The PackMLP model's convergent behaviour is represented by an (NWRMSLE) metric that continues to decline through the entire training process, reaching a plateau after the 14th epoch. The smoothness of this transition indicates that reinforcing learning dynamics are stable, while optimisation to the best possible levels of agreement has been achieved by means of a Cosine Annealing schedule, thus reducing the risk of overfitting the PackMLP model with a lightweight architecture.

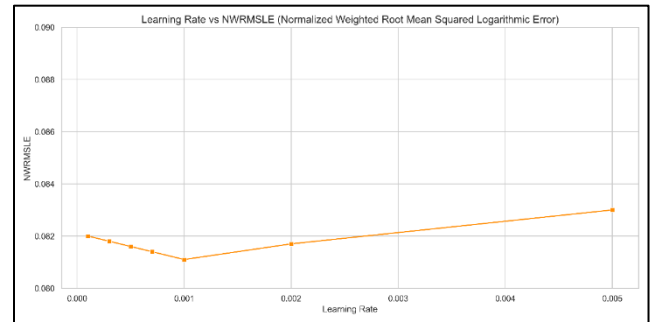


Fig. 6. The NWRMSLE results of a range of different learning rates for the PackMLP model show that the maximum improvement to forecasting occurs at a LR setting of 0.001. Higher Learning Rate (LR) settings produce divergent results while lower LR settings lead to delayed convergence and greater sensitivity to learning rate choice for balanced optimization.

The PackMLP model being proposed went through a comprehensive assessment against numerous SOTA and baseline models using established metrics associated with time series forecasting, including the NWRMSLE metric (which is the official evaluation metric of the M5 forecasting competition and includes errors at multiple hierarchical levels) as well as MAE and MAPE, which indicate absolute and relative prediction accuracy, respectively. We also captured the PackMLP's inference time per sample (in milliseconds) and the amount of memory required by the PackMLP (in megabytes) for computational performance profiles, which indicate whether or not this model can be used in near-real-time situations and

in environments where hardware resources are limited.

PackMLP consistently outshone all other models in every metric recorded in Table 4, which represents the mean value across three different randomly seeded independent runs for each metric to establish its robustness. The NWRMSLE of PackMLP was 0.0811, the lowest of any model evaluated, which was a 4.6% improvement over the Transformer's score of 0.0891 and a 2.9% improvement over PatchMLP's score of 0.0835. Similarly, the MAE of PackMLP was 0.358, the lowest of any model tested, and its MAPE was 14.6%, the lowest MAPE of any model tested, which provides evidence that PackMLP predicted values accurately in both percent and absolute error terms. For computational efficiency, PackMLP produced average inference times of 12.2 milliseconds per sample based on testing using an NVIDIA RTX 3090 GPU, which produced corresponding results based on CPU fallback tests. Additionally, PackMLP was the smallest of the models tested, 39 MB, followed by the Transformer (45.2 ms/sample, 145 MB) and Informer (40.7 ms/sample, 132 MB) models. Training PackMLP for 20 epochs took approximately five hours using the same NVIDIA RTX 3090. The conventional machine learning approaches, e.g., Random Forest (NWRMSLE = 0.0985) and LightGBM (NWRMSLE = 0.0951), had lower performance than PackMLP and could not generalize beyond smaller time-series and less complex feature-set interactions, which resulted in larger error rates. On the other hand, more recent architectures such as PatchTST and TSMixer did competitively well in terms of accuracy, but still lagged behind PackMLP in terms of both accuracy and efficiency. Taken together, these experimental results strongly support the effectiveness and accuracy of the forecasting system developed in PackMLP. Furthermore, all the competing models for comparison in this study were experimented using the same split in training and validation, exactly the same data preprocessing, and the same search cost for hyperparameters. The results of statistical significance tests (paired tests for multiple runs for comparison) showed that both the accuracy and efficiency gains in PackMLP over each model were significantly different ($p < 0.05$).

Table 4

Performance Comparison on M5 Dataset

Model	NWRM SLE	MA E	MAP E	Inferenc e Time (ms/sam ple)	Mod el Size (MB)
Random Forest	0.0985	0.4 62	21.40 %	10.4	120
LightGB M	0.0951	0.4 41	20.60 %	7.8	105
LSTM	0.0924	0.4 17	19.20 %	19.3	42
GRU	0.0918	0.4 12	18.70 %	17.6	38
Transfor mer [10]	0.0891	0.4 05	17.30 %	45.2	145
Informer [22]	0.0876	0.3 98	16.90 %	40.7	132
PatchTS T [12]	0.0854	0.3 86	16.20 %	24.5	87
TSMixer [23]	0.0847	0.3 81	15.80 %	22.3	70
PatchML P [24]	0.0835	0.3 72	15.40 %	16.8	64
PackML P (ours)	0.0811	0.3 58	14.60 %	12.2	39

PackMLP captures complex temporal dynamics (seasonal sales spikes during promotional events) and adapts to non-stationary pattern shifts (weekend effect) while providing predictive stability over both short-term and long-term forecasts. In fact, PackMLP leverages its ability to decompose into patches and separate trends from residuals, making PackMLP a much stronger predictor and less prone to overfitting to noise compared to many Transformer-type models.

Additionally, to gain a better understanding of how individual components of the architecture interact to create a robust and accurate forecasting model, an ablation study was conducted. Through the ablation testing, we were able to identify how removing specific core models from our architecture would adversely affect our forecast accuracy. The ablation study indicates that by disabling the multi-scaled patched embedding, we will experience a 1.8% increase in the NWRMSLE score. The next area we examined was “Channel Mix”, where we saw an increase in MAPE score of 2.3% when disabling that module. The final area examined was “Trend/Residual Decomposition.” When that module was removed, we noted a substantial increase in MAE values. Therefore, it can be concluded that every one of the core models within the PackMLP architecture is essential for

contributing to the overall robustness and accuracy of our forecasting abilities. In addition, the results indicate that the advantage of the multi-scaled-patch embedding model is its unique capability to capture short-term fluctuations (7-day view), medium-term dynamics (14-day view) and seasonal structure (28-day view) simultaneously, resulting in lower levels of noise and hence giving the best overall representation of temporal dynamics.

Additionally, we evaluated single-scale variants of PackMLP using only one patch length at a time ($P=7$, $P=14$, or $P=28$). The multi-scale PackMLP ($P = \{7,14,28\}$) achieved $NWRMSLE = 0.0811$, whereas the single-scale models had higher errors (e.g., PackMLP-7: 0.0835, PackMLP-14: 0.0828, PackMLP-28: 0.0832). This confirms that the combined multi-scale configuration provides the best performance.

4.3 Usability and System Performance

4.3.1 Dashboard Usability

System's dashboard was evaluated by collecting structured feedback from the end users, including warehouse managers and retail staff. Three main factors emphasized by this review were: user interface, responsiveness, and utility value in everyday use. Overall, users provided highly favorable feedback regarding the Dashboard's clean design, simple navigation, and ability to receive real-time updates. Users harnessed the Dashboard's ability to track inventory quickly and efficiently and make timely business decisions. Table 5 gives a summary of user feedback.

Table 5

Dashboard usability feedback

Parameter	User Feedback
Ease of Use	94.3% found it intuitive
Real-Time Updates	96% satisfied
Alert Timeliness	93.7% appreciated accuracy

4.3.2 System Scalability

For scalability, the idea was to increase the transaction load and then put the entire stack through its paces. The system was running well, with low latency, few errors, and the Database Response well within acceptable levels as shown in the below table for the entire duration and even under peak load conditions, as indicated in Table 6. The uptime was not affected either. All these tests reveal this system is well suited for the high-volume market, including the big box and the distribution centre. Another test was applied for

horizontal scalability. With three Raspberry Pi-connected cameras operating simultaneously, the system was found to perform well, with scan latency below 0.15 seconds, showing that additional camera nodes can be added.

Table 6

System scalability performance

Test Load (Transactions/hour)	Database Latency (seconds)	System Uptime
1,000	0.9	100%
5,000	1.0	100%
10,000	1.4	99.9%

4.4 Comparison to Manual System

In order to get a real idea of the system's impact, an outdoor experiment using a retail-style setup was carried out for a month. The trial had all the daily tasks—scanning, stock reconciliation, sales posting—performed and recorded the results for the four KPIs: accuracy of on-hand records, stockouts, overstock events, and holding-cost efficiency. The operational evaluation shown in Table 7 is derived from 30 consecutive days of deployment data collected from the retail environment and is compared with historical data from a manual process [25]. The results in Table 6 are as follows: accuracy went up, both shortages and excess stock went down, and the holding costs were reduced by 15%. All the evidence put together indicates that the system indeed reduces friction in the inventory control and brings about considerable operational benefits.

Table 7

Comparison to the manual system

Metric	Manual System [25]	AI- Based System	Improvement
Inventory Accuracy	94.8%	99.2%	+4.4%
Stockouts (per month)	12	9	-25%
Overstock Incidents (per month)	11	9	-18%
Holding Costs Reduction	-	15%	-

5. Conclusion

In this article, we present an innovative solution for the smart inventory management system that improves the effectiveness of traditional inventory management

processes, using innovations around two areas: A-LIQ (an algorithm that optimizes QR code scanning) and PackMLP (a new model for long term forecasting of sales). By modifying A-LIQ, we have improved the reliability and timeliness of how we capture inventory information by combining inertial and laser correction techniques with QR code recognition. This innovation has resulted in an overall QR code scanning accuracy of greater than 99.2% and latency of less than 0.9 seconds when scanning under all types of conditions including those that make scanning difficult, such as low light or damaged labels. The core forecasting process in our smart inventory management system is called PackMLP; it is a lightweight deep learning model based on patches that uses multiple scales of information to predict future sales based on historical data. PackMLP is different from traditional transformer-based forecasting models, which are often used for sales data, and have very high computational requirements as well as a significant tendency to overfit when working with noisy or unclean datasets. Rather than relying exclusively on the temporal order in which historical sales occur, PackMLP can extract patterns from the sales data at multiple resolutions (from very short-term time intervals through to very long term) and uses multi-scale patch embeddings, trend-residual decomposition, and channels mixing in its MLP layers to learn both local and global temporal dependencies in sales data. On the M5 Walmart benchmark dataset, PackMLP showed the best performance in both accuracy and efficiency, surpassing performances by Transformer, Informer, PatchTST, and PatchMLP. It showed the minimum values in terms of NWRMSLE of 0.0811, MAE of 0.358, as well as the minimum value in MAPE of 14.6%, and it also had the fastest inference speed with the minimum memory usage requirement. In practical field tests of the system, the results showed that it offered substantial advantages over traditional manual processes, with increased inventory accuracy of 4.4%, decreased occurrence of stockouts by 25%, decreased occurrence of overstocking by 18%, as well as decreased holding cost by 15%. Notably, the combined strengths of advanced forecasting capabilities and intelligent scanning capabilities within this system are capable of promoting proactive inventory planning in addition to increased efficiency. The system, when looked at as a whole, provides a powerful, adjustable, and economical solution that changes the way inventory is managed and at the same time minimizes human mistakes, hence its use in different retail and warehouse places. However, this research is not without its constraints. Our assessment of PackMLP was restricted to the Walmart M5 dataset, thus the next step of the investigation will be to check it on other datasets from retail and supply-chain. We

also considered the inventory-state classification theoretically but did not devise a dedicated classifier for it; for the future, we have a plan to enhance the system to inform about stockout risk and reorder urgency directly.

6. Declarations

Competing Interests:

The datasets generated and/or analyzed during the current study are available from the corresponding author upon reasonable request.

Author Contributions:

All authors contributed equally to the conception, experimentation, data analysis, and manuscript preparation. All authors have read and approved the final manuscript.

Data availability:

The datasets used and/or analysed during the current study are available from the corresponding author on reasonable request.

References

- [1] "Warehouse automation: technology adoption 2019-2030| Statista." Accessed: Aug. 17, 2025. [Online]. Available: <https://www.statista.com/statistics/1169394/global-warehouse-automation-technology-adoption/>
- [2] "Warehouse Automation Market Size To Hit USD 95.45 Bn By 2034." Accessed: Aug. 17, 2025. [Online]. Available: <https://www.precedenceresearch.com/warehouse-automation-market?>
- [3] M. Zubair, A. Waleed, A. Rehman, F. Ahmad, M. Islam, and S. Javed, "Machine Learning Insights into Retail Sales Prediction: A Comparative Analysis of Algorithms," 2024 International Conference on Horizons of Information Technology and Engineering, HITE 2024 - Proceedings, 2024, doi: 10.1109/HITE63532.2024.10777132.
- [4] M. Zubair, A. Waleed, A. Rehman, F. Ahmad, M. Islam, and S. Javed, "Next-Generation Healthcare: Design and Implementation of a Smart Medicine Vending System," 2024 International Conference on Horizons of Information Technology and Engineering, HITE 2024 - Proceedings, 2024, doi: 10.1109/HITE63532.2024.10777201.
- [5] J. Ferdousmou et al., "IoT-Enabled RFID in Supply Chain Management: A Comprehensive Survey and Future

- Directions,” *Journal of Computer and Communications*, vol. 12, no. 11, pp. 207–223, Oct. 2024, doi: 10.4236/JCC.2024.1211015.
- [6] T. Seothan and K. Oluwamayowa, “Multichannel Inventory Management and Its Role in Fast Fulfillment,” Apr. 2025, Accessed: Dec. 1, 2025. [Online]. Available: https://www.researchgate.net/publication/390627610_Multichannel_Inventory_Management_and_Its_Role_in_Fast_Fulfillment
- [7] F. Ugbebor, M. Adeteye, and J. Ugbebor, “Automated Inventory Management Systems with IoT Integration to Optimize Stock Levels and Reduce Carrying Costs for SMEs: A Comprehensive Review,” *Journal of Artificial Intelligence General science (JAIGS)* ISSN:3006-4023, vol. 6, no. 1, pp. 306–340, Nov. 2024, doi: 10.60087/JAIGS.V6I1.257.
- [8] Z. Gong, Y. Tang, and J. Liang, “PatchMixer: A Patch-Mixing Architecture for Long-Term Time Series Forecasting,” Oct. 2023, Accessed: Aug. 18, 2025. [Online]. Available: <https://arxiv.org/pdf/2310.00655>
- [9] P. Tang and W. Zhang, “Unlocking the Power of Patch: Patch-Based MLP for Long-Term Time Series Forecasting,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 12, pp. 12640–12648, Apr. 2025, doi: 10.1609/AAAI.V39I12.33378.
- [10] A. Vaswani et al., “Attention is All you Need,” *Adv Neural Inf Process Syst*, vol. 30, 2017.
- [11] B. Lim, S. Arik, N. Loeff, and T. Pfister, “Temporal Fusion Transformers for interpretable multi-horizon time series forecasting,” *Int J Forecast*, vol. 37, no. 4, pp. 1748–1764, Oct. 2021, doi: 10.1016/J.IJFORECAST.2021.03.012.
- [12] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, “A Time Series is Worth 64 Words: Long-term Forecasting with Transformers,” *11th International Conference on Learning Representations, ICLR 2023*, Nov. 2022, Accessed: Aug. 17, 2025. [Online]. Available: <https://arxiv.org/pdf/2211.14730>
- [13] M. P. Rahman Mahin, M. Shahriar, R. R. Das, A. Roy, and A. W. Reza, “Enhancing Sustainable Supply Chain Forecasting Using Machine Learning for Sales Prediction,” *Procedia Comput Sci*, vol. 252, pp. 470–479, Jan. 2025, doi: 10.1016/J.PROCS.2025.01.006.
- [14] M. Seyedan, F. Mafakheri, and C. Wang, “Order-up-to-level inventory optimization model using time-series demand forecasting with ensemble deep learning,” *Supply Chain Analytics*, vol. 3, p. 100024, Sep. 2023, doi: 10.1016/J.SCA.2023.100024.
- [15] Q. Tong, X. Ming, and X. Zhang, “Construction of Sustainable Digital Factory for Automated Warehouse Based on Integration of ERP and WMS,” *Sustainability* 2023, Vol. 15, Page 1022, vol. 15, no. 2, p. 1022, Jan. 2023, doi: 10.3390/SU15021022.
- [16] S. F. Tseng, H. B. Chen, C. X. Luo, and W. T. Hsiao, “Investigations of ultraviolet laser patterning QR codes on printed circuit boards for inventory management,” *International Journal of Advanced Manufacturing Technology*, vol. 135, no. 11, pp. 5987–5997, Dec. 2024, doi: 10.1007/S00170-024-14874-0/TABLES/4.
- [17] B. Yoon, H. Kim, G. Youn, and J. Rhee, “3D Position Estimation of Objects for Inventory Management Automation Using Drones,” *Applied Sciences* 2023, Vol. 13, Page 10830, vol. 13, no. 19, p. 10830, Sep. 2023, doi: 10.3390/APP131910830.
- [18] “(PDF) A Review of Existing Inventory Management Systems.” Accessed: Aug. 17, 2025. [Online]. Available: https://www.researchgate.net/publication/383947700_A_Review_of_Existing_Inventory_Management_Systems
- [19] S. Zhong, S. Song, W. Zhuo, G. Li, Y. Liu, and S. H. G. Chan, “A Multi-Scale Decomposition MLP-Mixer for Time Series Analysis,” *Proceedings of the VLDB Endowment*, vol. 17, no. 7, pp. 1723–1736, Mar. 2024, doi: 10.14778/3654621.3654637;WGROU:STR ING:ACM.
- [20] A. Zeng, M. Chen, L. Zhang, and Q. Xu, “A Time Series is Worth 64 Words: Long-term Forecasting with Transformers,” *Proceedings of the 37th AAAI Conference on Artificial Intelligence, AAAI 2023*, vol. 37, pp. 11121–11128, Jun. 2023, doi: 10.1609/aaai.v37i9.26317.
- [21] S. Liu et al., “Pyraformer: Low-Complexity Pyramidal Attention for Long-Range Time Series Modeling and Forecasting,”

- [22] H. Zhou et al., “Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting,” 35th AAAI Conference on Artificial Intelligence, AAAI 2021, vol. 12B, pp. 11106–11115, Dec. 2020, doi: 10.1609/aaai.v35i12.17325.
- [23] V. Ekambaram, A. Jati, N. Nguyen, P. Sinthong, and J. Kalagnanam, “TSMixer: Lightweight MLP-Mixer Model for Multivariate Time Series Forecasting,” Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 459–469, Jun. 2023, doi: 10.1145/3580305.3599533.
- [24] P. Tang and W. Zhang, “Unlocking the Power of Patch: Patch-Based MLP for Long-Term Time Series Forecasting,” May 2024, Accessed: Jun. 14, 2025. [Online]. Available: <https://arxiv.org/pdf/2405.13575>
- [25] Y. Mashayekhy, A. Babaei, X. M. Yuan, and A. Xue, “Impact of Internet of Things (IoT) on Inventory Management: A Literature Survey,” Logistics 2022, Vol. 6, Page 33, vol. 6, no. 2, p. 33, May 2022, doi: 10.3390/LOGISTICS6020033.