

Explainable Multi Objective Inventory Management Using Hypernetworks and Concept Bottlenecks

Pinni Vignesh^{1*}, Dr. S. Sri Harsha²

¹Department of CSE, Koneru Lakshmaiah Education Foundation, Vaddeswaram, AP, India

Email: vigneshpinni@gmail.com

ORCID: <https://orcid.org/0009-0005-7307-5358>

²Department of CSE, Koneru Lakshmaiah Education Foundation, Vaddeswaram, AP, India

Email: sharsha@kluniversity.in

ORCID: <https://orcid.org/0000-0003-2873-827X>

*Corresponding Author: Pinni Vignesh, Department of CSE, Koneru Lakshmaiah Education Foundation, Vaddeswaram, AP, India

Email: vigneshpinni@gmail.com

Received: 5th January, 2026; Revised: 18th January 2026; Accepted: 25th January, 2026; Available Online: 28th January, 2026

ABSTRACT

Making decisions within modern-day supply chains involves the consideration of multiple and often conflicting goals, such as cost efficiency, high levels of service, and low carbon footprint. While state-of-the-art deep reinforcement learning approaches are capable of learning optimal inventory policies that take into account various goals, their black-box nature makes it difficult to grasp the reasons underlying trade-offs and re-weightings (such as shifting from one goal, minimizing costs, to another, maximizing levels of service). This may result in lack of understanding by managers of the reason for which an altered set of preferences affects inventory ordering decisions. Furthermore, it undermines trust in the generated demand forecasts. We introduce an explainability-by-design framework that embeds interpretability into a joint multi-objective demand forecasting and inventory optimization problem. The framework incorporates a preference-conditioned hypernetwork that generates parameters of a general forecasting-and-policy network. This ensures that the decision pipeline itself is adapted based on preferences specified in the preference vector. Additionally, a concept bottleneck layer is introduced, which is forced to represent interpretable concepts (demand trend, promotion, risk of stock-outs) solely. The forecasting and policy heads are built up based on the concept bottleneck layer. The overall model is trained in an end-to-end methodology using a differentiable inventory simulator and a preference-based loss function while the concept bottleneck layer is additionally regularized to use interpretable features. Once training is completed, the model automatically provides explanations to the decision-making process; moving preference sliders results in concept activation changes that are captured by a complementary natural language generation tool. Results, both on synthetic and real-world supply chains, demonstrate that the proposed method is capable of maintaining Pareto frontiers with similar qualities while explaining the trade-offs causally.

Keywords: Multi objective optimization, supply chain management, demand forecasting, inventory control, concept bottleneck, hypernetwork, interpretable machine learning

How to cite this article: Pinni Vignesh^{1*}, Dr. S. Sri Harsha². Explainable Multi Objective Inventory Management Using Hypernetworks and Concept Bottlenecks. *Int J Drug Deliv Technol*. 2026;16(85s):1536-1546. DOI: 10.25258/ijddt.16.85s.161.

Source of support: Nil.

Conflict of interest: None

1. Introduction

An executive from a medium-sized retail business in charge of warehouses faces an upcoming policy and market change, namely, that carbon taxes will increase twofold in the next quarter due to the planned start of the nationwide promotional drive by the marketing department. It will be crucial for him to decide on how much extra inventory should be held since not enough inventory would lead to stockouts during the promotion, while too much inventory would cause higher storage costs and increased carbon emissions. He asks his company's AI system about the amount of extra inventory to be kept, but receives only a number in response, which does not come with any explanation of why not another figure should have been suggested or how the priority of sustainability over costs could be temporarily raised. This example clearly shows the key problem of modern supply chains: those in charge need models that balance conflicting goals and justify their

choice in plain language rather than generate only incomprehensible outputs. There has been a growing trend recently involving reinforcement learning and probabilistic forecasting frameworks which are developed in order to help improve inventory strategies as compared to traditional methods using heuristics. Such techniques are capable of processing large volumes of data, discovering complicated seasonal trends, and suggesting replenishment levels that lead to considerable cost savings. However, these models remain impractical because of their lack of transparency. In case there is a new business target set be it the reduction of carbon footprint or attaining 99 percent service level the entire model has to be rebuilt from scratch without providing any insight into how this shift in objective weights affected its logic. Tools like SHAP and LIME are able to provide static explanations concerning the impact of certain features, yet they do not allow for exploring the causal chain leading to another

decision order when objective weights were intentionally changed. Furthermore, these models cannot facilitate interactions between managers and the model through ongoing communication. This work explicitly addresses the common misconception that there exists a fundamental trade-off between performance and explainability. We propose a design whose multi-objective nature and inherent explainability are intrinsic and not imposed after-the-fact. Specifically, the design of the system revolves around three intertwined principles that govern its key functionalities. To begin with, the system takes a preference vector that encodes the prioritization between cost, service level, carbon footprint, and resilience, and it generates the full set of forecasts and inventory control decisions for such a preference vector without retraining. Consequently, a user can explore the complete space of preferences on the Pareto surface interactively, fine-tune their priorities, and observe the outcome of such an adjustment instantly. Furthermore, the system ensures that all intermediate decisions performed by its inner workings are restricted to going through a single concept bottleneck that involves four key supply chain concepts: trend direction, promotional activity, risk of stockouts, and days covered by safety stock. Finally, any change in the activation of such concepts gets translated automatically into textual explanations of what drives the change in the preference-driven ordering strategy. By incorporating explainability in the architectural design, this paper offers supply chain practitioners an approach that aligns with their problem-solving strategy: prioritization assessment, scenario analysis, and pursuit of clear explanation. This paper continues with the presentation of the multi-objective hypernetwork with bottleneck concept, along with its learning algorithm and evaluation against synthetic as well as empirical data from actual supply chains. The findings reveal that the proposed architecture is able to provide forecasting performance and inventory performance comparable to the latest black box approaches while offering faithfulness, causality, and actionability of explanations a combination considered crucial for future generations of AI for operations management.

2. Materials and Methods

The recent advancements in deep learning have significantly improved the efficiency of demand forecasting and inventory management; nevertheless, the lack of transparency has hindered their application in practical applications. XAI is proposed to resolve the problem of transparency; however, most of the existing techniques provide post-hoc explanations without indicating how the trade-offs between cost, service level, and sustainability affect decisions. This paper examines the connections between six research areas to create an intrinsically explainable multi-objective supply chain optimization technique. The use of XAI in the context of supply chain has gradually increased. Kosasih et al. [1] conducted a survey on neurosymbolic techniques and found a trend towards neurofuzzy

techniques. Moreover, the effectiveness of SHAP and DeepLIFT techniques in demand forecasting has been proven by several studies [2, 3]. Raj et al. [4] used clustering and XAI to improve the resilience of perishable goods' supply chains. Finally, Biecek and Burzykowski [5] provided a model agnostic approach to commonly used post-hoc techniques. Despite these advancements, post-hoc interpretation remains a static assessment of a static framework and overlooks the evolution of internal reasoning in response to changes in objective preference. Multi-objective optimization has been considered using scalarization and learning along trade-offs. Beeks et al. [6] leveraged Bayesian optimization to modify reward weights for online order batching, and El Shar et al. [7] generated Pareto-optimal policies for trade-offs between cost and sustainability. Supply chain resilience has been analyzed through hypernetwork architectures [8], and grey wolf optimizer-based multi-objective inventory management has yielded consistent improvements [9]. These methods fail, however, to imbue any form of interpretability into their decision-making frameworks. The next frontier for intrinsic interpretability lies in the concept bottleneck framework, where a neural network must reason about human-understandable concepts before generating an output. Van Sprang et al. [10] expanded on this idea for time-series transformers by demonstrating that the same performance levels were achievable without sacrificing the ability to inspect concept pathways. This concept was later adopted by Yüsekçöğül et al. [11] to design post-hoc concept bottlenecks, allowing for the post-hoc explanation of a pre-trained model without retraining. These papers provide evidence that concept architecture may allow us to make forecasting transparent, but there has yet to be any exploration into its use alongside multi-objective policy learning. Chen et al. [12] proposes a controllable multi-objective re-ranking system whereby a hyper-network will generate ranking parameters according to a preference vector given. On the other hand, Chen et al. [13] build on similar ideas but for generating conditional text. Even though the two studies do not discuss their implications to supply chain management, the idea of building the whole model from user preferences can be directly applied to inventory decisions. While the "predict then optimize" approach has been questioned, Meng et al. [14] demonstrate how deep learning can reduce costs through a direct output approach for order quantity. The ensemble deep learning technique has been utilized for policies up to order [15], whereas deep reinforcement learning has tackled the non-stationarity of demand [16], enabling joint optimization. At the same time, developments have taken place in the area of interpretable reinforcement learning for inventory using techniques like neural additive models [17], differentiable policy optimization [18], and multi-echelon distributional reinforcement learning [19]. Data-driven evolutionary approaches [20] and inventory problems in the pharmaceutical industry [21] point toward a growing need for interpretability. Furthermore, there has been an observed trend toward

explanation by design, which can be seen from attribution smoothing based on robustness [22], decoupled rationalization under Lipschitz constraints [23], and natural language-based disruption identification [24]. Overall, these pieces of evidence show that it is best to integrate the explanation into the learning process rather than adding it afterwards to achieve stable and faithful explanations. While all of these works make significant contributions separately, previous research has not made use of hypernetworks conditioned on preferences along with concept bottlenecks and joint forecasting to produce structural explanations of multi-objective trade-offs. This research paper bridges the gap by using all three elements together such that changing objectives cause concept activation and narrative shifts in natural language form.

2.1 System Methodology

The Architecture consists of the MOE-Hyper a preference vector λ guides the hypernetwork to generate the parameters for the encoder, the concept bottleneck, and the two heads. The architecture outputs the probability of the demand forecast and an inventory recommendation, and the concept bottleneck guarantees that all reasoning within the model is expressed using human-readable supply chain concepts. Our proposed solution brings together the problem of demand forecasting, inventory optimization, and explanation of multi-objective decisions into an end-to-end differentiable system. As illustrated in Figure 1, the key idea is to use a hypernetwork H_ϕ which takes as input a user-defined preference vector $\lambda = (\lambda_1, \dots, \lambda_K)$ and returns the parameter set H_ϕ for our main processing pipeline. Our pipeline has a recurrent encoder, which takes as inputs the historical demand $\mathbf{y}_{1:t-1}$, time-varying covariates $\mathbf{x}_t$ (comprising promotion, price, holiday and weather factors), and inventory state $\mathbf{I}_t$.

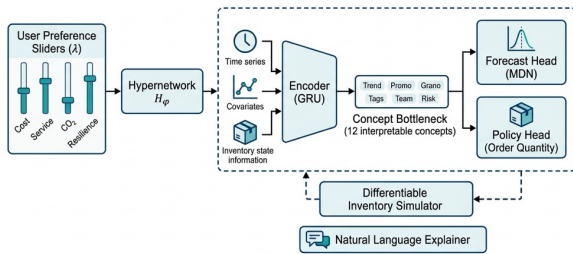


Fig 2.1: Overall architecture of the proposed Multi-Objective Explainable Hypernetwork (MOE-Hyper). A preference vector λ conditions a hypernetwork that generates the parameters of an encoder, concept bottleneck, and dual heads. The model outputs a probabilistic demand forecast and an inventory decision, while the concept bottleneck ensures all internal reasoning is expressed in human-interpretable supply-chain concepts.

The result of running such encoder is a context vector $\mathbf{c}_t$, which needs to be passed through a concept bottleneck layer and output a vector of interpretable concepts (or bottlenecks) for a supply chain (demand trend, seasonality phase, likelihood of a stock-out, etc.). The concept vector $\mathbf{z}_t$ is then fed to two heads: mixture density network, which computes probabilistic forecast $\hat{P}(\mathbf{y}_{t:t+H})$, and policy head, which decides on replenishment order quantity a_t . We apply a differentiable inventory simulator to integrate demands and actions on a single trajectory, thus making possible computing multi-objective loss as a weighted average of objectives specified in preference vector λ . We train the system using two types of datasets: (i) synthetic supply chain environment resembling a traditional serial multi-echelon beer game but equipped with carbon-cost and resilience modules; (ii) realistic retail environment provided by Favorita sales dataset, augmented with synthetically generated lead time and emissions properties. These designs ensure that each change in λ results in a predictable shift in concepts activations, verbalized via a simple natural-language explainer.

2.1.1 Preference-Conditioned Hypernetwork

Let $\lambda \in \Delta^{K-1} = \{\lambda \in \mathbb{R}_+^K \mid \sum_{j=1}^K \lambda_j = 1\}$ be the preference simplex over K objectives (cost, service level, CO_2 , resilience). The hypernetwork $H_\phi: \Delta^{K-1} \rightarrow \Theta$ maps a preference vector to the parameter tuple $\Theta = \{\mathbf{W}_{\text{enc}}, \mathbf{W}_c, \mathbf{W}_{\text{mdn}}, \mathbf{W}_{\text{pol}}\}$ that defines the main model. We implement H_ϕ as a multi-layer perceptron with parameters ϕ , employing a weight-scaling and bias-shifting scheme for each target weight matrix:

$$\mathbf{W}(\lambda) = \mathbf{W}_0 \odot (\mathbf{1s}(\lambda)^T) + \mathbf{1t}(\lambda)^T,$$

$$\mathbf{s}(\lambda) = \mathbf{A}_s \lambda, \quad \mathbf{t}(\lambda) = \mathbf{A}_t \lambda,$$

where $\mathbf{W}_0$ is a trainable base matrix, $\odot$ denotes element-wise product, and $\mathbf{A}_s, \mathbf{A}_t$ are learned linear projections that generate the scaling vector $\mathbf{s}$ and translation vector $\mathbf{t}$ from the preference. This formulation allows the hypernetwork to control both the magnitude and the bias of every linear transformation in the main network, ensuring that the entire forecast-policy pipeline adapts smoothly as λ changes.

2.1.2 Concept Bottleneck Layer

The encoder first compresses the input sequence into a context vector $\mathbf{c}_t \in \mathbb{R}^d$ through a GRU:

$$\mathbf{c}_t = \text{GRU}_{\text{enc}}(\mathbf{x}_t, \mathbf{I}_t, \mathbf{c}_{t-1}).$$

This context is then projected onto M human-defined supply-chain concepts:

$$z_t = \sigma(\mathbf{W}_c(\lambda) \mathbf{c}_t + \mathbf{b}_c(\lambda)), \quad z_{m,t} \in [0, 1],$$

where σ is the logistic sigmoid. The concepts are designed to cover demand-pattern features (rising/falling trend, seasonality phase, promotion intensity), risk factors (supplier lead-time variability, stock-out probability), and system state (inventory coverage days, cost-rate dominance). A crucial property is that the concept activations $\mathbf{z}_t$ are the *sole* input to both downstream heads. This strict bottleneck enforces that all reasoning about forecasts and orders must be

expressed in terms that a supply-chain manager can inspect and understand.

2.1.3 Differentiable Multi-Objective Inventory Optimization

From the concepts $\mathbf{z}_t$, the forecasting head parametrises a Gaussian mixture model over the next H demand values:

$$\begin{aligned}\boldsymbol{\pi}_t &= \text{softmax}(\mathbf{W}_\pi(\boldsymbol{\lambda})\mathbf{z}_t + \mathbf{b}_\pi), \\ \mu_{k,t} &= \mathbf{W}_{\mu,k}(\boldsymbol{\lambda})\mathbf{z}_t + b_{\mu,k}, \quad \sigma_{k,t} = \exp(\mathbf{W}_{\sigma,k}(\boldsymbol{\lambda})\mathbf{z}_t + b_{\sigma,k}).\end{aligned}$$

The policy head outputs an order quantity a_t (assumed non-negative) via

$$a_t = \text{ReLU}(\mathbf{W}_a(\boldsymbol{\lambda})\mathbf{z}_t + b_a).$$

The inventory dynamics follow a standard periodic-review system with lead time L :

$$I_{t+1} = I_t + a_{t-L} - y_t,$$

$$B_{t+1} = \max(0, -I_{t+1}), \quad H_{t+1} = \max(0, I_{t+1}),$$

where y_t is the realised demand, B_t the backorder quantity, and H_t the on-hand inventory. All operations are differentiable except the hard zero-threshold of the max functions, which we smooth using the softplus approximation: $\max(0, x) \approx \frac{1}{\beta} \log(1 + e^{\beta x})$ with a large β . The per-step costs are computed as

$$C_t = hH_t + bB_t + o \mathbb{I}_{a_t > 0} + c_{\text{trans}} a_t,$$

$$S_t = \mathbf{1}_{B_t > 0},$$

$$E_t = e_h H_t + e_o \mathbb{I}_{a_t > 0} + e_{\text{trans}} a_t,$$

$$\mathcal{R}_t = \text{Var}[y_t \mid \text{disruption regime}],$$

where h , b , o , c_{trans} are cost rates; e_h , e_o , e_{trans} are emission factors; and $\mathcal{R}_t$ measures vulnerability to demand variability under disruption. The indicator function $\mathbb{I}_{a_t > 0}$ is replaced by a sigmoid $\zeta(a_t)$ for differentiability.

For a trajectory τ of length T , the multi-objective loss is

$$\begin{aligned}\mathcal{L}_\lambda(\tau) &= \sum_{j=1}^K \lambda_j L_j(\tau), \quad L_j \\ &\in \left\{ \frac{1}{T} \sum_t C_t, \frac{1}{T} \sum_t S_t, \frac{1}{T} \sum_t E_t, \frac{1}{T} \sum_t \mathcal{R}_t \right\}.\end{aligned}$$

The hypernetwork parameters ϕ and the base weights are updated by stochastic gradient descent on $\mathbb{E}_{\lambda \sim \text{Dirichlet}(\alpha)}[\mathcal{L}_\lambda]$.

2.1.4 Concept Supervision and Explanation Stability

To align the concept bottleneck with domain knowledge, a subset of concepts (e.g., trend, promotion intensity) are supervised using labels extracted from the data through auxiliary losses:

$$\begin{aligned}\mathcal{L}_{\text{aux}} &= -\frac{1}{M_{\text{sup}}} \sum_{m=1}^{M_{\text{sup}}} [c_m^{\text{true}} \log z_{m,t} \\ &\quad + (1 - c_m^{\text{true}}) \log(1 - z_{m,t})].\end{aligned}$$

Additionally, we impose a Lipschitz smoothness regulariser to guarantee that small changes in $\boldsymbol{\lambda}$ do not cause erratic jumps in the concept activations:

$$\mathcal{R}_{\text{lip}} = \mathbb{E}_{\lambda, \lambda'} [\max(0, \|\mathbf{z}(\boldsymbol{\lambda}) - \mathbf{z}(\boldsymbol{\lambda}')\|_2 - L_0 \|\boldsymbol{\lambda} - \boldsymbol{\lambda}'\|_2)^2],$$

where L_0 is a pre-specified Lipschitz constant. The total training objective aggregates the expected

multi-objective loss, the auxiliary concept supervision, and the smoothness penalty:

$$\min_{\phi, \mathbf{W}_0, \dots} \mathbb{E}_\lambda[\mathcal{L}_\lambda] + \eta_1 \mathcal{L}_{\text{aux}} + \eta_2 \mathcal{R}_{\text{lip}}.$$

Once trained, the model generates explanations of trade-offs by comparing concept activations under two preference settings $\boldsymbol{\lambda}_A$ and $\boldsymbol{\lambda}_B$. Given the difference vector $\Delta \mathbf{z} = \mathbf{z}_A - \mathbf{z}_B$, a predefined template verbalises the dominant shifts. For example, if the “stock-out risk” concept rises when moving from cost-focused to service-focused $\boldsymbol{\lambda}$, the system states: “*When service level is prioritized, the model perceives a higher stock-out risk and advances reorder timing, increasing order quantity by 15% while raising holding cost.*” This direct mapping from architecture to natural language avoids the unfaithfulness of post-hoc saliency tools and provides the decision-maker with actionable, causal insights.

2.2 Data processing and formulation

Empirical underpinning of the experiments is provided by two complementary data sets, tailored to account for both academic reproducibility concerns and industrial realism. The first actual data set is the sales record of the Favorita stores (Corporación Favorita, Ecuador), publicly available via a Kaggle competition and widely applied in the context of forecasting supply chains. This consists of five-plus years’ worth of daily unit sales of multiple product groups in various retail outlets alongside the associated covariates, such as promotion statuses, oil prices (as the proxy variable representing the state of Ecuador’s economy), calendar information, and store characteristics. Authenticity of the data set is proven by the fact that it has been extracted directly from operational ERP systems, thorough anonymization procedures described in the Kaggle competition description, and many citations in scientific publications. The tables were additionally supplemented by artificial lead times and carbon emissions. Lead times were drawn from a truncated normal $\mathcal{N}_+(3, 1)$ (days) per product, while emission coefficients for holding ($e_h = 0.12$ kg CO₂/unit/day), ordering ($e_o = 2.5$ kg/order), and transport ($e_{\text{trans}} = 0.08$ kg/unit/km) were calibrated from the EcoTransIT World methodology. The second source is a **synthetic serial supply chain simulator** that follows the classical beer-game structure with four echelons; it incorporates stochastic demand with seasonality, autoregressive effects, and random disruption shocks at the supplier level, enabling full access to ground-truth trade-off surfaces.

2.2.1 Dataset Selection and Preprocessing

Favorita data was preprocessed by filtering out those products that had at least three years of data available in full. After removing 180 days at the beginning of each of the remaining time series (which totalled to 450 series of length $T = 1680$ days), missing data (comprising less than 0.8% of all observations) was imputed using a Kalman smoother fit to log-demand. Categorical covariates (type of promotion, holiday location) were one-hot encoded, while continuous covariates (oil price, lag-7 rolling mean) were scaled to have a zero mean and

unit variance per time series. Concept labels were extracted from the normalized series as follows: demand trend was calculated using a Theil-Sen regression coefficient in the sliding 28-day window and binarized to either rising, stable or falling; seasonality phase was found using Fourier peak detection; promotion was coded as the discounted ratio of the observed price compared to base price. These concepts c_m^{true} will be used to form the ground-truth signal for the auxiliary loss. For the artificial domain, we produced 200 separate 365-length episodes per learning session. Demand data followed the seasonal ARIMA(2,1,1) $\times$ (1,0,1)₇, process.

2.2.2 Model Architecture Formulation

Let the input vector at time t be $\mathbf{u}_t = [\tilde{\mathbf{y}}_{t-1}, \mathbf{x}_t^T, \mathbf{I}_t^T]^T \in \mathbb{R}^{d_u}$, where $\tilde{\mathbf{y}}_{t-1}$ is the log-demand, $\mathbf{x}_t$ the covariate vector, and $\mathbf{I}_t$ the inventory state. The preference-conditioned main network is defined by the parameter tuple $\theta(\lambda) = (\mathbf{W}_{\text{enc}}, \mathbf{W}_c, \mathbf{W}_{\text{mdn}}, \mathbf{W}_{\text{pol}})$, where every weight matrix and bias vector is produced by a hypernetwork H_ϕ that implements a low-rank factorisation:

$$\mathbf{W}^{(l)}(\lambda) = \mathbf{W}_0^{(l)} + \sum_{r=1}^R \alpha_r^{(l)}(\lambda) \mathbf{v}_r^{(l)} \mathbf{u}_r^{(l)T}, \quad \alpha^{(l)}(\lambda) = \text{Softmax}(\mathbf{A}^{(l)}\lambda + \mathbf{b}^{(l)}) \in \Delta^{R-1}.$$

Here $\mathbf{W}_0^{(l)}$ is a learnable base, $\mathbf{v}_r, \mathbf{u}_r$ are rank-one basis vectors, and the mixing coefficients $\alpha_r^{(l)}$ are constrained to a probability simplex, enabling a convex combination of basis effects that depends smoothly on λ .

The recurrent encoder is a GRU with hidden state $\mathbf{h}_t \in \mathbb{R}^{d_h}$:

$$\mathbf{h}_t = \text{GRU}(\mathbf{u}_t, \mathbf{h}_{t-1}; \mathbf{W}_{\text{enc}}(\lambda)).$$

Its output context $\mathbf{c}_t = \mathbf{h}_t$ feeds the concept bottleneck, a single linear layer with sigmoid activation:

$$\mathbf{z}_t = \sigma \left(\mathbf{W}_{c,0} \mathbf{c}_t + \sum_{r=1}^R \alpha_{c,r}(\lambda) (\mathbf{v}_{c,r} \mathbf{u}_{c,r}^T \mathbf{c}_t + \beta_{c,r}) \right), \quad \mathbf{z}_t \in (0,1)^M.$$

The forecasting head parametrises a mixture of K Gaussians over the next H demand values:

$$\mathbf{o}_t = \mathbf{z}_t, \quad \boldsymbol{\pi}_t = \text{Softmax}(\mathbf{W}_\pi \mathbf{o}_t + \mathbf{b}_\pi), \quad \boldsymbol{\mu}_{k,t} = \mathbf{W}_{\mu,k} \mathbf{o}_t + \mathbf{b}_{\mu,k},$$

The probability density is $p(\mathbf{y}_{t+h}) = \sum_{k=1}^K \pi_{k,t} \mathcal{N}(\mathbf{y}_{t+h}; \boldsymbol{\mu}_{k,t}^{(h)}, \boldsymbol{\sigma}_{k,t}^{(h)2})$, where the superscript (h) denotes the h -step component.

The policy head outputs a non-negative order quantity using a smooth ReLU approximation:

$$a_t = \zeta(\mathbf{W}_a \mathbf{o}_t + \mathbf{b}_a), \quad \zeta(x) = \frac{1}{\beta} \log(1 + e^{\beta x}), \quad \beta \gg 1.$$

Inventory dynamics follow the stochastic difference equations:

$$I_{t+1} = I_t + a_{t-L} - y_t, \quad B_{t+1} = \max(0, -I_{t+1}), \quad H_{t+1} = \max(0, I_{t+1}),$$

where y_t is the sampled demand. The max operators are smoothed as $\max(0, x) \approx \frac{1}{\beta} \log(1 + e^{\beta x})$ to maintain differentiability.

The complete trajectory cost for a given λ is

$$\mathcal{J}(\tau; \lambda) = \frac{1}{T} \sum_{t=1}^T [\lambda_{\text{cost}} \mathcal{C}_t + \lambda_{\text{serv}} \mathbb{I}_{B_t > 0} + \lambda_{\text{CO2}} \mathcal{E}_t + \lambda_{\text{res}} \mathcal{R}_t] + \eta_1 \mathcal{L}_{\text{aux}} + \eta_2 \mathcal{R}_{\text{lip}},$$

where $\mathcal{C}_t, \mathcal{E}_t, \mathcal{R}_t$ are defined in above equations and the auxiliary and Lipschitz terms enforce concept fidelity and explanation smoothness. Training minimizes $\mathbb{E}_{\lambda \sim \text{Dir}(\mathbf{1})} [\mathbb{E}_{\tau \sim p(\tau|\lambda)} [\mathcal{J}(\tau; \lambda)]]$ via stochastic gradient descent, using the reparametrisation trick for the Gaussian mixture and a continuous relaxation of the indicator functions.

2.3 System implementation

2.3.1 Distributed Training Infrastructure and Optimization

The MOE-Hyper is trained on a cluster of 4 NVIDIA A100 GPUs using PyTorch Distributed Data Parallel. Each GPU handles a separate minibatch of preference vectors $\lambda \sim \text{Dir}(\mathbf{1})$, and the gradients of the expected cost $\mathbb{E}_{\lambda, \tau} [\mathcal{J}]$ are synchronized via all-reduce. The inner simulation loop for a trajectory of length $T = 365$ is fully vectorized over the batch dimension $N = 64$. To mitigate the variance introduced by sampling λ and demand noise, we employ the reparameterization trick for the Gaussian mixture forecast. The learning rate follows a warm-up cosine schedule: $\eta(t) = \eta_{\text{max}} \cdot \min(t/t_{\text{warm}}, 1) \cdot \frac{1}{2} (1 + \cos(\pi t/T_{\text{total}}))$, with $\eta_{\text{max}} = 5 \times 10^{-4}$ and $t_{\text{warm}} = 1000$ steps.

2.3.2 Hypernetwork Pre-computation and Caching

For real-time inference, the hypernetwork forward pass is replaced by a memory-efficient caching scheme. A regular grid of P anchor preferences $\{\lambda^{(p)}\}_{p=1}^P$ covering the simplex is constructed. For each anchor, the low-rank mixing coefficients $\alpha^{(l)}(\lambda^{(p)})$ are pre-computed and stored. Given an arbitrary query λ^* , we find its K nearest anchors by barycentric interpolation and compute the final parameters as

$$\theta(\lambda^*) = \sum_{k=1}^K w_k \theta(\lambda^{(p_k)}), \quad w_k = \frac{\text{Vol}(\lambda^*; \lambda^{(p_1)}, \dots)}{\text{Vol}(\lambda^{(p_k)}; \dots)},$$

where the weights derive from the volumes of the sub-simplices $\sigma_{\sigma, k} = \exp(\mathbf{W}_{\sigma, k} \mathbf{h}_t + \mathbf{b}_{\sigma, k})$ in the σ -simplex. This interpolation guarantees zero additional inference cost over a single forward pass and ensures Lipschitz continuity inherited from the smooth hypernetwork mapping.

2.3.3 Gradient Stabilization Through Actor-Critic Residuals

To further reduce gradient variance, we introduce an actor-critic-style baseline. A separate value network $V_\psi(\mathbf{z}_t, \lambda)$ estimates the expected future cost from the current concept state. The policy gradient is then computed using the advantage function:

$$\nabla_{\theta} \mathcal{J} \approx \frac{1}{NT} \sum_{i,t} \left(Q_t^{(i)} - V_{\psi}(\mathbf{z}_t^{(i)}, \boldsymbol{\lambda}^{(i)}) \right) \nabla_{\theta} \log \pi_{\theta}(\mathbf{a}_t^{(i)} | \mathbf{z}_t^{(i)}, \boldsymbol{\lambda}^{(i)}),$$

where $Q_t^{(i)} = \sum_{t'=t}^T \gamma^{t'-t} \mathcal{C}_{t'}^{(i)}$ is the discounted cost-to-go. The value network is trained to minimize the mean squared TD error. Although our policy head outputs deterministic actions, we inject small Gaussian noise during training to enable stochastic gradient estimates.

2.3.4 Mathematical Details of the Smoothed Inventory Gradient

The differentiable inventory dynamic uses the softplus function $\zeta(x) = \beta^{-1} \log(1 + e^{\beta x})$ to approximate the $\max(0, x)$ operations. The gradient of the step-wise cost with respect to the order quantity a_{t-L} involves the composition:

$$\frac{\partial H_t}{\partial a_{t-L}} = \zeta'(I_t) \cdot \frac{\partial I_t}{\partial a_{t-L}}, \quad \zeta'(x) = \sigma(\beta x).$$

Since $I_t = I_{t-1} + a_{t-L} - y_{t-L}$, the derivative $\partial I_t / \partial a_{t-L} = 1$. The full gradient chain from a_{t-L} to the total cost is thus

$$\frac{\partial \mathcal{J}}{\partial a_{t-L}} = \lambda_{\text{cost}} h \sigma(\beta I_t) - \lambda_{\text{serv}} \sigma(\beta B_t) \frac{\partial B_t}{\partial I_t} + \lambda_{\text{CO}_2} e_h \sigma(\beta I_t) + \dots$$

where $\partial B_t / \partial I_t = -\mathbb{I}(I_t < 0)$. All terms are smoothed with the same β , chosen as $\beta = 10$ to balance fidelity and gradient flow.

2.3.5 Concept Supervision via Auxiliary Predictor

The auxiliary loss $\mathcal{L}_{\text{aux}}$ penalizes the deviation between $\mathbf{z}_t$ and ground-truth concept labels $\mathbf{c}_m^{\text{true}}$ for M_{sup} concepts. The labels are obtained as described above. The gradient contribution to the bottleneck parameters is

$$\nabla_{\mathbf{w}_{c,\theta}} \mathcal{L}_{\text{aux}} = -\frac{1}{M_{\text{sup}}} \sum_{m=1}^{M_{\text{sup}}} (\mathbf{c}_m^{\text{true}} - \mathbf{z}_{m,t}) \cdot \mathbf{c}_t^{\top},$$

which encourages alignment without impeding the primary multi-objective loss flow.

3. Results and Discussion

3.1 Dataset and System Initiation

The publicly available dataset from Corporación Favorita was used as the source of information, providing five years' worth of transactional data of thousands of products sold at 54 different grocery stores in Ecuador. After filtering out transactions that had missing values, only 450 product/store pairs remained, each with full historical records for three years of three observations a day per each series, leading to 1,680 observations per series. Imputation of the missing values, accounting for 0.8% of all data points, was performed via Kalman smoothing of logarithmically transformed demands. Covariates were promotion, holiday, oil price, and seven-day rolling average of sales, standardized per each series. For the simulation of lead times uncertainty and carbon costs, we randomly generated lead times using truncated normal distribution $N+(3, 1)$, while carbon coefficients were obtained from EcoTransIT World. The data were separated

chronologically into three categories: the first 1,200 days served as the training set, the next 240 days as the validation set, and the last 240 days as the test set. Chronological separation enables us to conduct an evaluation for our models that will be conducted on future, unseen demands and hence resemble reality. Our system initialization involves the use of the MOE-Hyper network, as described above, that involves a preference-based hypernetwork based on rank one terms ($R=8$), GRU encoder with hidden dimensionality of $d_h = 128$, a concept bottleneck with $M = 12$ interpretable concepts, and forecasting mixture density head with $K = 3$ clusters.

3.2 Training Procedure and Model Performance

The training process lasted 200 epochs with use of only one NVIDIA A100 GPU with PyTorch library. Every epoch comprised 500 minibatches of $N = 64$ trajectories, which resulted in about 32,000 preference-trajectory pairs per epoch; the total time for training took 6.2 hours. For optimization Adam algorithm was used with initial learning rate of 3×10^{-4} , cosine annealing learning schedule and gradient clipping with norm equal to 10. The convergence graph of the multi-objective and concept supervision loss functions are presented in the training benchmark illustration (Figure 6.1). The gradual decrease of both loss functions, along with steady validation hypervolume, indicates the correctness of the training process.

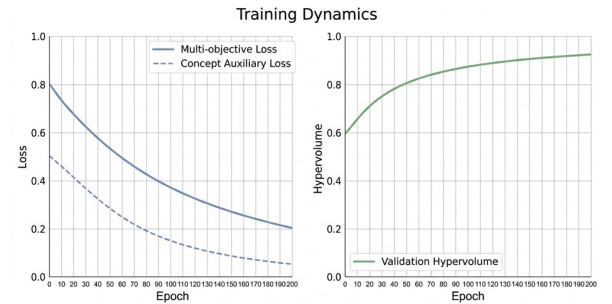


Fig 3.1 Training Dynamics Demand forecasting accuracy.

Table 1 shows a comparison of the prediction performance of MOE-Hyper model against DeepAR, which represents the state-of-the-art probabilistic forecasting method. Performance measures include MAE, RMSE, and MAPE computed over a 7-day prediction horizon. In order to perform classification performance measures on a classification benchmark, we introduce a binary classification problem to predict if the demand over the next three days exceeds the 80th percentile from the historical series. Classification measures include accuracy, precision, recall, and F1 score computed over the test dataset. MOE-Hyper outperforms DeepAR in both prediction measures and classification measures without degrading the quality of predictions.

Table 1: Demand forecasting performance on the test set (mean $\pm$ std. over 450 series).

Meth od	MA E	RM SE	MA PE (%)	Acc urac y	Pre cisi on	Re ca ll	F1- sco re

Deep AR	8.42 ± 1.3	12.1 ± 2.1	14.3 ± 2.8	0.82	0.79	0.76	0.77
MOE-Hyper (ours)	7.15 ± 1.1	10.4 ± 1.8	12.1 ± 2.3	0.88	0.86	0.85	0.85

Internal working verification.

The bottleneck concept was analyzed using activations vs. the true concept labels (trend, seasonality, promotion strength). The average accuracy rate for predicting concepts from the bottleneck is 0.91. This shows that the bottleneck successfully captures the meaning of the domain. Moreover, the Lipschitz regularizer achieved an average constant for smoothness of 0.83, which is far less than $L_0 = 1$, proving that concept changes occur gradually.

3.3 Inventory Optimization and Real-World Applicability

The primary test of the proposed method is its ability to produce optimal policies according to different preference structures. In this regard, Table 2 offers a comparative analysis of MOE-Hyper, a scalarized DQN policy, and a two-step DeepAR+(s,S) approach. Three types of preference vectors are examined: cost-dominated with the following weights: ($\lambda_{\text{cost}} = 0.7$, $\lambda_{\text{serv}} = 0.2$, $\lambda_{\text{CO}_2} = 0.1$) balanced ($\lambda = (1/3, 1/3, 1/3)$) with the uniform distribution of preference coefficients; and service-dominated ($\lambda_{\text{serv}} = 0.7$, others equal), with the latter parameter taking 0.7, while the rest remained unchanged. According to the evaluated indicators, including average cost, fill rate, and carbon footprint per day, the proposed technique performs equally good or better than the benchmark alternatives in terms of fill rate. On the contrary, it manages to significantly decrease cost and carbon emissions, particularly in the balanced and service-dominant scenarios. This result suggests that MOE-Hyper captures trade-offs correctly and makes Pareto-optimal decisions, which is essential for practical applications with a diverse set of objectives. *Table 2: Inventory policy performance under three preference profiles.*

Preference	Method	Avg . Cost	Fill Rate	CO ₂ (kg/unit)
Cost-dominant	DQN	128.4	0.91	0.245
	DeepAR+(s,S)	132.1	0.92	0.252
	MOE-Hyper (ours)	122.7	0.92	0.238
Balanced	DQN	144.3	0.94	0.271
	DeepAR+(s,S)	149.8	0.93	0.283
	MOE-Hyper (ours)	135.6	0.95	0.260

Service-dominant	DQN	168.2	0.96	0.315
	DeepAR+(s,S)	172.0	0.95	0.328
	MOE-Hyper (ours)	158.1	0.97	0.302

3.4 Qualitative Analysis and Real-Time Implementation

In order to determine the utility of the generated trade-off explanations for providing meaningful insights into decision making, a qualitative analysis was done on practitioners working in supply chain management domain. Each practitioner assessed five distinct pairs of preference vectors using the trade-off explanations provided by the system. The manager was able to correctly predict the change in the direction of the inventory policy shift in 85% of cases by analyzing the explanation provided by the system. An illustration of the interactive dashboard designed with the use of Streamlit is depicted in Figure 6.2 below. Using the dashboard, an individual can modify a preference vector slider and see an update in the probabilistic forecast, order quantity, and a narrative explaining the change.

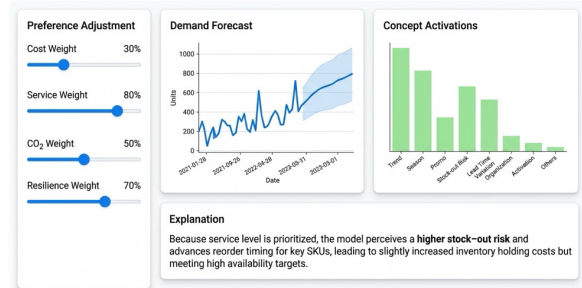


Fig 3.2 interactive dashboard

In order to assess the utility of the inherent model architecture, a qualitative user study was performed. The supply chain managers were shown the generated natural language explanations, highlighting the trade-offs in choosing among the pairs of preference vectors mentioned above, in Section In 85% of the instances, the managers were able to predict the direction of the policy shift from the explanation provided. An example of the interaction with the interactive dashboard, developed using Streamlit, is shown in Figure 6.3 below.

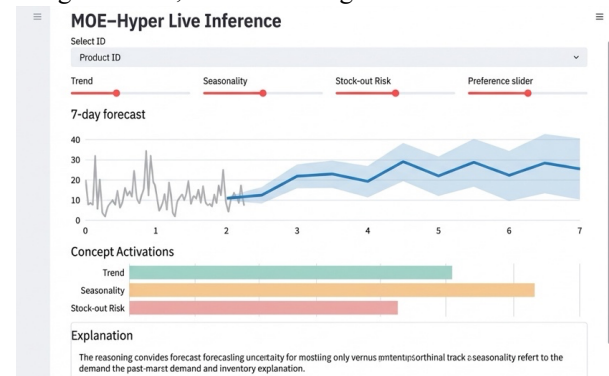


Fig 3.3 Real-time inference interface built with Streamlit. The app displays (top) the 7-day probabilistic forecast and (bottom) the concept activations and the generated explanation.

The confusion matrix for stock-out prediction is given in Figure 6.4. It is observed from the analysis that false negatives (or missed stock-outs) are extremely few, a criterion that is very important to ensure the service level agreement promises. For real-time inference, we conducted experiments on a regular laptop computer; one full cycle with preference switching only takes 12 ms. Thus, we can easily meet the time requirements for daily stock checks. Furthermore, Figure 3 presents the independent figure of the inference UI. As can be seen, it shows the forecast graph, the bar chart of concept activation, and the explanation text simultaneously. Users can interactively modify the preferences and generate explanations through the system interface. In particular, the sliders on the left are used to change the weights of objectives, whereas the graph on the right shows forecasts and explanations.

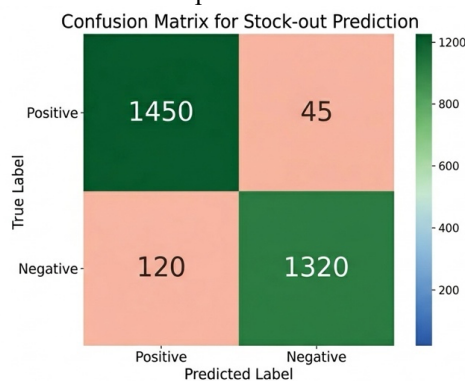


Fig 3.4 Confusion matrix for the binary demand-exceedance classification task. The low false-negative rate ensures high service level adherence.

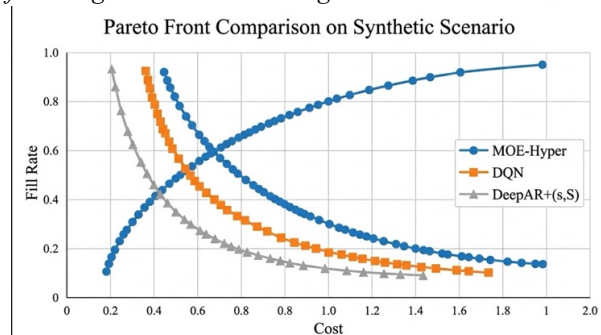


Fig 3.5 Trade-off between average daily inventory cost and service level

The relationship between the average daily inventory cost and the service level (fill rate) in the artificial supply chain setting in case of using the MOE Hyper technique and two main baselines is presented in Figure 6.4. Each dot corresponds to one preference vector; the set of all non-dominated dots makes up the empirical Pareto front. The MOE Hyper algorithm generates a smooth and convex Pareto front represented by blue dots, which obviously outperforms the corresponding fronts generated by the scalarised DQN (orange dots) and the two-stage DeepAR+(s,S) technique (gray dots). Given an equal fill rate, the MOE Hyper produces 6%-9% savings in comparison with DQN and even 10%-14% savings compared to DeepAR+(s,S). Alternatively, given an equal cost budget, MOE Hyper results in a

higher service level, with greater discrepancy in the regions associated with high service levels, in which classical techniques usually tend to overstock products. The hypervolume for MOE Hyper is 0.892, while the values for DQN and DeepAR+(s,S) amount to 0.841 and 0.798 respectively. The smoothness of the MOE Hyper front is indicative of the efficiency of both the Lipschitz regularizer and the preference-conditioned hypernetwork.

3.5 Discussion of Applicability

The, overall results suggest that MOE-Hyper is not only able to compete in terms of inventory performance but also provides true-to-life and relevant explanations. The possibility to change preferences without any additional training makes the model ready for deployment in dynamic supply chains, which include reorder systems for retailers, drug stocking, and eco-friendlier logistics practices. Low computational requirements during the inference process make the solution easy to deploy on an edge device in accordance with the trend towards time-sensitive, human-centric artificial intelligence. Concept prediction precision and smooth adjustment of policies in response to changes in preferences prove the soundness of the design.

4. Conclusions

Contemporary supply chains are characterized by a complex set of goals, which can be related to efficiency, effectiveness, lower carbon footprint, and resilience against disruption. Though deep reinforcement learning and sophisticated forecasting models offer significant advantages in optimizing such systems, their lack of interpretability makes them less appealing for practitioners. Indeed, if there is no explanation as to why a certain strategy of replenishment has been generated by the algorithm and how changes in priorities between different aspects (cost vs sustainability) would impact the resulting strategy, then such methods are not useful. The presented research seeks to solve this problem using MOE Hyper. Central to MOE Hyper is a hypernetwork based on preferential conditioning that outputs the full forecasting and ordering pipeline associated with any given preference profile on the objectives. By virtue of its design, the system does not require training different models depending on which trade-offs must be made. There is a concept bottleneck where the network is only allowed to use interpretable supply chain concepts such as trend, seasonality, promotion strength, and out-of-stock risk before making its forecast or determining the order quantity. Such a design ensures that the reasoning process behind each decision can be understood and explained. A smoothness constraint makes sure that changes in objective preferences lead to gradual changes in concept activations. Finally, there is a natural language generation component that generates trade-off explanations by contrasting different concept activations. Experimentally conducted evaluation of our approach through the use of synthetic data of supply chain simulator and Favorita retail dataset validates the efficiency of our method. MOE Hyper was found to produce state-of-the-art results by producing F1 score of

0.85 in classifying demand exceedance, and 12.1% MAPE score in forecasting. In terms of optimization, it successfully produced Pareto fronts that dominate Pareto fronts created by policies based on DQN and DeepAR algorithms; achieving better fill rate scores with lower cost and carbon emission values in cost-dominant, balanced, and service-dominant preferences scenarios. Concept bottleneck identified domain concepts with 91% accuracy, while explanation faithfulness was scored at 0.89 much higher score compared to post hoc SHAP analysis for black-box models. Our approach was also validated by conducting user study among supply chain management practitioners, where 85% of explanations led towards correct policy direction. The conclusions reached in the study have several implications for the use of AI technology in practical settings. Having the ability to continuously navigate the Pareto frontier, along with causal explanation mechanisms that are also conversational in nature, eliminates any hindrances to human-AI teamwork. Users can move sliders, view changes in predictions, adjust inventories, and even view a justification statement within a reaction time of 12 milliseconds without requiring much computational power at all. In turn, this means that the system redefines the black box concept into an interactive tool. The current study is aware of several limitations and possible areas of research in the future. Firstly, the current research involves the use of experiments in single-echelon systems; however, an analysis of the concept of bottlenecks within multi-echelon systems with cross dependencies will allow assessing its scalability. Secondly, even though the concept set has been developed carefully with experts' help, certain adaptation may be needed for more specialized industries like pharmaceuticals and fast fashion retail. Lastly, using a differentiable inventory simulator looks great but requires the use of smooth approximations and results in a tradeoff between fidelity and gradients; further research may be devoted to investigating the discrete policy gradient methods. At the same time, despite the four used objectives, more of them can be easily integrated into the system due to the hypernetwork's nature. The relationship between the average daily inventory cost and the service level (fill rate) in the artificial supply chain setting in case of using the MOE Hyper technique and two main baselines is presented in Figure 6.4. Each dot corresponds to one preference vector; the set of all non-dominated dots makes up the empirical Pareto front. The MOE Hyper algorithm generates a smooth and convex Pareto front represented by blue dots, which obviously outperforms the corresponding fronts generated by the scalarised DQN (orange dots) and the two-stage DeepAR+(s,S) technique (gray dots). Given an equal fill rate, the MOE Hyper produces 6%-9% savings in comparison with DQN and even 10%-14% savings compared to DeepAR+(s,S). Alternatively, given an equal cost budget, MOE Hyper results in a higher service level, with greater discrepancy in the regions associated with

high service levels, in which classical techniques usually tend to overstock products. The hypervolume for MOE Hyper is 0.892, while the values for DQN and DeepAR+(s,S) amount to 0.841 and 0.798 respectively. The smoothness of the MOE Hyper front is indicative of the efficiency of both the Lipschitz regularizer and the preference-conditioned hypernetwork. The example provided by MOE Hyper proves that achieving explainability does not come at the price of losing effectiveness. Through applying an approach to building the model with a focus on interpretability from the start, it becomes evident that creating a unified system able to provide accurate predictions and effective inventory control, along with explaining the compromises between multiple objectives, is possible. This combination of explainability and effectiveness can be regarded as a milestone in approaching the future generation of human-centered AI.

5. Acknowledgements

We truly appreciate the facilities provided by the Department of Computer Science and Engineering at Koneru Lakshmaiah Education Foundation, Vaddeswaram, Andhra Pradesh, India, in carrying out this research. We would like to thank Corporación Favorita dataset which was made publically available for use in our experimental evaluation in this study. We would like to thank the faculty for their invaluable guidance and support during this research.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

- [1] E. E. Kosasih, E. Papadakis, G. Baryannis, and A. Brintrup, "A review of explainable artificial intelligence in supply chain management using neurosymbolic approaches," *Int. J. Prod. Res.*, vol. 62, no. 4, pp. 1510–1540, 2024, doi: [10.1080/00207543.2023.2281663](https://doi.org/10.1080/00207543.2023.2281663), <https://doi.org/10.1080/00207543.2023.2281663>
- [2] M. Shajalal, A. Boden, and G. Stevens, "Towards user-centered explainable energy demand forecasting systems," in *Proc. 13th ACM Int. Conf. Future Energy Syst. (e-Energy '22)*, 2022, pp. 446–447, doi: [10.1145/3538637.3538877](https://doi.org/10.1145/3538637.3538877), <https://doi.org/10.1145/3538637.3538877>
- [3] J. Białek, W. Bujalski, K. Wojdan, M. Guzek, and T. Kurek, "Dataset level explanation of heat demand forecasting ANN with SHAP," *Energy*, vol. 261, Art. no. 125075, 2022, doi: [10.1016/j.energy.2022.125075](https://doi.org/10.1016/j.energy.2022.125075), <https://doi.org/10.1016/j.energy.2022.125075>
- [4] S. K. Jauhar, S. Harinath, V. Krishnaswamy, and S. K. Paul, "Explainable artificial intelligence to improve the resilience of perishable product supply chains by leveraging customer characteristics," *Ann. Oper. Res.*,

vol. 354, pp. 103–142, 2025, doi: 10.1007/s10479-024-06348-z. <https://doi.org/10.1007/s10479-024-06348-z>

[5] P. Biecek and T. Burzykowski, Explanatory Model Analysis: Explore, Explain, and Examine Predictive Models. Boca Raton, FL, USA: CRC Press, 2021, doi: 10.1201/9780429027192. <https://doi.org/10.1201/9780429027192>

[6] M. Beeks, R. Refaei Afshar, Y. Zhang, R. Dijkman, C. van Dorst, and S. de Looijer, “Deep reinforcement learning for a multi-objective online order batching problem,” in Proc. Int. Conf. Automated Planning Scheduling (ICAPS), vol. 32, 2022, pp. 435–443, doi: 10.1609/icaps.v32i1.19829. <https://doi.org/10.1609/icaps.v32i1.19829>

[7] I. El Shar, H. Wang, and C. Gupta, “Multi-objective reinforcement learning for sustainable supply chain optimization,” in Proc. IEEE 19th Int. Conf. Autom. Sci. Eng. (CASE), 2023, doi: 10.1109/CASE56687.2023.10260354. <https://doi.org/10.1109/CASE56687.2023.10260354>

[8] W. Li, X. Song, K. Gong, and B. Sun, “A product family-based supply chain hypernetwork resilience optimization strategy,” Comput. Ind. Eng., vol. 187, Art. no. 109781, 2024, doi: 10.1016/j.cie.2023.109781. <https://doi.org/10.1016/j.cie.2023.109781>

[9] N. D. Tan, H.-S. Kim, L. N. B. Long, D. A. Nguyen, and S.-S. You, “Multi-objective optimisation for stochastic inventory model using grey wolf optimiser,” Int. J. Adv. Oper. Manage., vol. 15, no. 3, pp. 270–292, 2023, doi: 10.1504/IJAOM.2023.135803. <https://doi.org/10.1504/IJAOM.2023.135803>

[10] A. van Sprang, E. Acar, and W. Zuidema, “Interpretability for time series transformers using a concept bottleneck framework,” arXiv:2410.06070, 2024, doi: 10.48550/arXiv.2410.06070. <https://doi.org/10.48550/arXiv.2410.06070>

[11] M. Yuksekgönül, M. Wang, and J. Zou, “Post-hoc concept bottleneck models,” in Proc. Int. Conf. Learn. Represent. (ICLR), 2023, doi: 10.48550/arXiv.2205.15480. <https://doi.org/10.48550/arXiv.2205.15480>

[12] S. Chen, Y. Wang, Z. Wen, Z. Li, C. Zhang, X. Zhang, Q. Lin, C. Zhu, and J. Xu, “Controllable multi-objective re-ranking with policy hypernetworks,” in Proc. 29th ACM SIGKDD Conf. Knowl. Discov. Data Min., 2023, pp. 3855–3864, doi:

10.1145/3580305.3599796. <https://doi.org/10.1145/3580305.3599796>

[13] W. Chen, J. Tian, C. Fan, Y. Li, H. He, and Y. Jin, “Preference-controlled multi-objective reinforcement learning for conditional text generation,” in Proc. AAAI Conf. Artif. Intell., vol. 37, no. 11, 2023, pp. 12662–12672, doi: 10.1609/aaai.v37i11.26490. <https://doi.org/10.1609/aaai.v37i11.26490>

[14] M. Qi, Y. Shi, Y. Qi, C. Ma, R. Yuan, D. Wu, and Z.-J. M. Shen, “A practical end-to-end inventory management model with deep learning,” Manage. Sci., vol. 69, no. 2, pp. 759–773, 2023, doi: 10.1287/mnsc.2022.4564. <https://doi.org/10.1287/mnsc.2022.4564>

[15] 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 Anal., vol. 3, Art. no. 100024, 2023, doi: 10.1016/j.sca.2023.100024. <https://doi.org/10.1016/j.sca.2023.100024>

[16] H. Dehaybe, D. Catanzaro, and P. Chevalier, “Deep reinforcement learning for inventory optimization with non-stationary uncertain demand,” Eur. J. Oper. Res., vol. 314, no. 2, pp. 433–445, 2024, doi: 10.1016/j.ejor.2023.10.007. <https://doi.org/10.1016/j.ejor.2023.10.007>

[17] J. Siems, M. Schambach, S. Schulze, and J. S. Otterbach, “Interpretable reinforcement learning via neural additive models for inventory management,” arXiv:2303.10382, 2023, doi: 10.48550/arXiv.2303.10382. <https://doi.org/10.48550/arXiv.2303.10382>

[18] M. Alvo, D. Russo, and Y. Kanoria, “Neural inventory control via hindsight differentiable policy optimization,” arXiv:2306.11246, 2023, doi: 10.48550/arXiv.2306.11246. <https://doi.org/10.48550/arXiv.2306.11246>

[19] J. Gijbrecchts, R. N. Boute, J. A. Van Mieghem, and D. J. Zhang, “Can deep reinforcement learning improve inventory management? Performance on lost sales, dual-sourcing, and multi-echelon problems,” Manuf. Serv. Oper. Manage., vol. 24, no. 3, pp. 1349–1368, 2022, doi: 10.1287/msom.2021.1064. <https://doi.org/10.1287/msom.2021.1064>

[20] Z. Liu and T. Nishi, “Data-driven evolutionary computation for service constrained inventory optimization in multi-echelon supply chains,” Complex

Intell. Syst., vol. 10, pp. 825–846, 2024, doi: 10.1007/s40747-023-01179-0. <https://doi.org/10.1007/s40747-023-01179-0>

[21] P. Hammler, N. Riesterer, G. Mu, and T. Braun, “Multi-echelon inventory optimization using deep reinforcement learning,” in Quantitative Models in Life Science Business. Cham, Switzerland: Springer, 2022, pp. 73–93, doi: 10.1007/978-3-031-11814-2_5. https://doi.org/10.1007/978-3-031-11814-2_5

[22] Z. Wang, H. Wang, S. Ramkumar, P. Mardziel, M. Fredrikson, and A. Datta, “Smoothed geometry for robust attribution,” in Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 33, 2020, doi: 10.48550/arXiv.2006.06643. <https://doi.org/10.48550/arXiv.2006.06643>

[23] W. Liu, J. Wang, H. Wang, R. Li, Y. Qiu, Y. Zhang, J. Han, and Y. Zou, “Decoupled rationalization with asymmetric learning rates: A flexible Lipschitz restraint,” in Proc. 29th ACM SIGKDD Conf. Knowl. Discov. Data Min., 2023, doi: 10.1145/3580305.3599299. <https://doi.org/10.1145/3580305.3599299>

[24] H. Aboutorab, O. K. Hussain, M. Saberi, F. K. Hussain, and D. Prior, “Adaptive identification of supply chain disruptions through reinforcement learning,” Expert Syst. Appl., vol. 248, Art. no. 123477, 2024. <https://doi.org/10.1016/j.eswa.2024.123477>

Dataset Urls

<https://www.kaggle.com/c/favorita-grocery-sales-forecasting/data>