Stocktake: A Practical End-to-End Inventory Management Model with Deep Learning

Pointers

- Proposes a one-step end-to-end framework that uses deep-learning models to output suggested replenishment amounts directly from input features without intermediate steps.
  - The model is trained to emulate the behavior of the optimal dynamic programming solution based on historical observations, without assuming specific distributions for demand or vendor lead time (VLT).
- End-to-end (E2E) here refers to training a model to output inventory replenishment decisions directly from input data.
- The approach designs a modular deep learning framework with two independent prediction blocks:
  - A Recurrent Neural Network (RNN) for demand prediction.
  - A Multi-Layer Perceptron (MLP) for VLT prediction.

  These blocks are combined with features such as the review period and initial inventory to produce the final replenishment decision.

Glossary

- Vendor Lead Time (VLT): The time between placing an order and its arrival.
- Predict-then-optimize (PTO): A methodology involving prediction of input parameters followed by optimization based on those predictions.
- End-to-end (E2E): Directly mapping inputs to decisions without intermediate steps.
- Recurrent Neural Network (RNN): Is a deep learning model that processes sequential data to produce a sequential output.
- Multi-quantile RNN (MQRNN): Receives multiple time series (demand time series, promotion time series) as inputs and produces a daily demand prediction over a set of quantiles as outputs.
- Optimal Replenish Decisions (OPT): Obtained by solving the replenishment problem with known demand and VLT.
- Gradient Boosting Machine (GBM): Is a machine learning technique based on boosting a functional space, where the target is pseudo-residuals instead of residuals as in traditional boosting. It combines multiple weak models to create a powerful model.
- D_t: Sequence of random demands for $t = 1, \dots, T$.
- I_t: Inventory level at the beginning of period t.
- h: Unit holding cost.
- b: Unit backorder cost.
- M: Number of orders.

- $a = f(x)$: Optimal order quantity at each review point.
- x: Observed features, i.e. historical demand, VLT, item specs, and temporal information.
- Labelling: For supervised learning, compute a_i^*, the optimal order quantity for each replenishment time point i.

Model

Inventory Level Update

$$I_{t+1} = I_t - D_t + \sum_{m=1}^{M} a_m \nVdash \{t = t_m + L_m\}$$

- This equation describes how inventory evolves over time.
- I_t: Inventory level at the beginning of period t.
- D_t: Demand occurring during period t, depleting the inventory.
- $\sum_{m=1}^{M} a_m \nVdash \{t = t_m + L_m\}$: Total replenishment quantities a_m arriving during period t, determined by lead time L_m.
- $\nVdash \{t = t_m + L_m\}$: Indicator function equal to 1 if t matches the replenishment arrival time $t_m + L_m$, otherwise 0.
- **Purpose**: Tracks how inventory is updated dynamically based on demand and replenishment arrivals.

Cost Incurred at Period t

$$S_t = h \left[I_t - D_t + \sum_{m=1}^{M} a_m \nVdash \{t = t_m + L_m\} \right]^+ + b \left[-I_t + D_t - \sum_{m=1}^{M} a_m \nVdash \{t = t_m + L_m\} \right]^+$$

- This equation calculates the cost incurred during period t.
- h: Holding cost per unit of excess inventory.
- b: Backorder cost per unit of unmet demand (shortage).
- $[\cdot]^+ = \max(\cdot, 0)$: Ensures non-negative values for overstock or shortages.
- $I_t - D_t + \ldots$: Net inventory after meeting demand and accounting for replenishment arrivals.
- The cost has two components:
  - **Holding Cost**: $h\,[\ldots]^+$, incurred if net inventory is positive (excess stock).
  - **Backorder Cost**: $b\,[\ldots]^+$, incurred if net inventory is negative (stockout).
- **Purpose**: Quantifies the trade-off between overstocking and stockouts for cost minimization.

Objective Function

$$\min_{a_1, \ldots, a_M} \mathbb{E} \left[\sum_{t=1}^{T} S_t \right]$$

- The goal is to minimize the **expected total cost** over a finite planning horizon of T periods.
- Decision variables $a_1, \ldots, a_M$: Replenishment quantities.
- S_t: Cost at each period t (from the previous equation).
- $\mathbb{E}[\cdot]$: Expectation accounts for uncertainty in demand and vendor lead time.
- **Purpose**: Provides a framework to optimize replenishment decisions that balance costs under uncertainty.

Supervised Learning Mapping

$$\min_{f:\chi\to R}\sum_{i=1}^{N} L(f(x_i), a_i^*)$$

- Maps the replenishment optimization problem to a supervised learning setting.
- $f(x_i)$: Model predicts the replenishment quantity for input features x_i.
- a_i^*: Optimal replenishment quantity derived from dynamic programming.
- $L(\cdot)$: Loss function measuring deviation between $f(x_i)$ and a_i^*.
- χ: Input feature space (i.e. historical demand, lead time, etc.).
- R: Target output space (replenishment quantities).
- **Purpose**: Trains a machine learning model to approximate the optimal replenishment decision.

Dynamic Programming Recursion

$$V_m(I_{v_m}) = \min_{a_m\geq 0}\left\{\sum_{s=v_m}^{v_{m+1}-1}\left[h\left(I_{v_m}+a_m-d_{[v_m,s]}\right)^+ + b\left(d_{[v_m,s]}-I_{v_m}-a_m\right)^+\right] + V_{m+1}\left(I_{v_m}+a_m-d_{[v_m,v_{m+1}-1]}\right)\right\}$$

- This recursion calculates the **optimal cost** $V_m(I_{v_m})$ over the interval $[v_m, v_{m+1}-1]$.
- I_{v_m}: Inventory level at the start of the interval.
- a_m: Decision variable (replenishment quantity at time v_m).
- $d_{[v_m,s]}$: Cumulative demand between periods v_m and s.
- The expression has two components:
  - **Immediate Costs**: Holding and backorder costs within the interval.
  - **Future Costs**: Continuation value $V_{m+1}(\ldots)$, which depends on future states.
    * The term $V_{m+1}\left(I_{v_m}+a_m-d_{[v_m,v_{m+1}-1]}\right)$ represents the future cost starting at V_{m+1} based on the inventory level at the end of the current interval. Accounting for the expected future cost in the dynamic programming recursion is essential because it ensures that the current decision is optimal not just for the immediate time period but also in the context of the entire planning horizon.
- **Purpose**: Decomposes the multi-period optimization problem into smaller, solvable subproblems.

Theorem of the Dynamic Programming Recursion

$$a_m^{**} = \arg\min_{a_m\geq 0}\left\{\sum_{s=v_m}^{v_{m+1}-1}\left[h\left(I_{v_m}+a_m-d_{[v_m,s]}\right)^+ + b\left(d_{[v_m,s]}-I_{v_m}-a_m\right)^+\right] + V_{m+1}\left(I_{v_m}+a_m-d_{[v_m,v_{m+1}-1]}\right)\right\}$$

Explanation: This equation describes the optimal inventory replenishment strategy by minimizing:

- **Immediate Costs:** Holding cost h for excess inventory and backorder cost b for unmet demand.
- **Future Value Function:** V_{m+1} represents the expected future cost at the next decision point.

Evaluations:

- a_m: Decision variable representing the replenishment quantity for period m.
- I_{v_m}: Initial inventory level at the beginning of period m.
- $d_{[v_m,s]}$: Cumulative demand from v_m to s.

- h: Per-unit holding cost for excess inventory.
- b: Per-unit backorder cost for unmet demand.
- $(x)^+$: Positive part of x, defined as $\max(x, 0)$.
- $V_{m+1}(I)$: Future value function parameterized by ending inventory level.

Closed-Form Optimal Solution

$$a_m^{**} = \max\left\{d_{[v_m, s^*]} - I_{v_m}, 0\right\}$$
$$s^* = \left\lfloor \frac{b(v_{m+1} - v_m)}{h + b} \right\rfloor + v_m$$

Explanation: This is the closed-form solution that calculates the replenishment a_m^{**} required to meet demand up to a critical point s^*. It balances holding cost h and backorder cost b.

Evaluations:

- $d_{[v_m, s^*]}$: Cumulative demand forecast up to s^*.
- I_{v_m}: Current inventory level.
- s^*: Critical point determined by the ratio of b to $h + b$, rounded down.

MQRNN Training Objective Function

$$\min_{\theta} \sum_{i=1}^{N} \left[L(\text{Out1}_i; a_i^*) + \lambda_1 \hat{L_1}(\text{Out2}_i; a_{DF,i}^*) + \lambda_2 \hat{L_2}(\text{Out3}_i; a_{VLT,i}^*) \right]$$

Explanation: This function trains the MQRNN model by minimizing:

- L: Replenishment decision loss based on Out1.
- $\hat{L_1}$: Demand forecast loss based on Out2.
- $\hat{L_2}$: VLT (Vendor Lead Time) forecast loss based on Out3.

The weights λ_1 and λ_2 balance the contributions of forecast losses.

Evaluations:

- N: Total number of training samples.
- Out1_i: Output for replenishment decisions.
- Out2_i: Output for demand forecasts.
- Out3_i: Output for VLT forecasts.
- a_i^*: Optimal replenishment decision.
- $a_{DF,i}^*$: Optimal demand forecast.
- $a_{VLT,i}^*$: Optimal VLT forecast.
- λ_1, λ_2: Hyperparameters weighting the forecast losses.

Base-Stock Levels

For Normally Distributed Demand:

$$BM_{\text{normal}} = \mu_D(R + \mu_{\text{VLT}}) + \phi^{-1}\left(\frac{b}{b+h}\right)\sqrt{(R+\mu_{\text{VLT}})\sigma_D^2 + \mu_D^2\sigma_{\text{VLT}}^2}$$

- Computes the optimal base-stock level accounting for uncertainty in demand (μ_D, σ_D) and VLT $(\mu_{\text{VLT}}, \sigma_{\text{VLT}})$.
- ϕ^{-1}: Quantile of the standard normal distribution, representing the service level (based on b and h).

For Gamma Distributed Demand:

$$BM_{\text{gamma}} = Q_{\bar{D},\text{gamma}}\left(\frac{b}{b+h}\right)$$

- Computes the base-stock level for gamma-distributed demand using the quantile function $Q_{\bar{D},\text{gamma}}$ evaluated at the desired service level.

BM1

$$\min_{a_m \geq 0} \mathbb{E}_{\hat{d_m}}\left[b(\hat{d_m} - a_m + I_{t_m})^+ + h(a_m + I_{t_m} - \hat{d_m})^+\right]$$

Explanation: BM1 minimizes the expected inventory cost based on forecasted demand $\hat{d_m}$, balancing backorder (b) and holding costs (h).

Evaluations:

- a_m: Replenishment decision for period m.
- $\hat{d_m}$: Forecasted demand for period m.
- I_{t_m}: Inventory at the start of period t_m.
- b, h: Backorder and holding cost parameters.

BM2

$$\min_{a_m \geq 0} \sum_{t=v_m}^{v_{m+1}-1} \left[h\left(I_{t_m} + a_m - d_{[t_m,t]}\right)^+ + b\left(d_{[t_m,t]} - I_{t_m} - a_m\right)^+\right]$$

- Sum of holding and backorder costs over the planning interval $[v_m, v_{m+1} - 1]$.

Explanation: BM2 minimizes the total inventory cost (holding and backorder) over the multi-period horizon $[v_m, v_{m+1} - 1]$.

Evaluations:

- a_m: Replenishment decision for period m.
- I_{t_m}: Initial inventory at the start of period t_m.
- $d_{[t_m,t]}$: Cumulative demand between t_m and t.
- h, b: Cost parameters.

Results

T-Test

- E2E algorithm significantly reduces all five performance metrics (holding cost, stockout cost, total inventory cost, turnover rate, stockout ratio).
- Ideal observation: Low p-value, statistically signficant results
  - E2E algorithm can reduce the average holding cost by 26.1%, average stockout cost by 51.7%, total cost is reduced by 40.4%, average turnover rate reduced by 8.8% and stockout ratio reduced by 34.6%

Linear Regression

$$\text{Outcome } = \theta_0 + \theta_1 \text{Is-E2E} + \theta_2 \text{Ave-Demand} + \theta_3 \text{Ave-VLT}$$

- Is-E2E: Is a binary independent variable that represents if a SKU orders using E2E algorithm
- Ave-Demand: Is an independent variable that represents the average demand of a SKU
- Ave-VLT: Represents the average VLT of a SKU
- The linear regression model aims to provide a better comparison of the treatment and control groups' average outcome by considering covariates that may affect the outcome of the metrics.
- Ideal observation: E2E algorithm leads to significant reductions on all five performance metrics.

Difference-in-differences (DID) Estimation

- Compare the performance of pre-experiment period with that of the post-experiment test period
- Ideal observation: Negative DID for the five performance metrics indicates improves between pre and post-experiment period.

Advantages of an End-to-End Model

1. **Direct Method:** The E2E model skips intermediate steps, directly mapping input features to output.
2. **No Prior Assumptioons Required:** The E2E model does not rely on assumptions about the distribution of demand or lead times. (Adaptable to a diverse range of scenarios.)
3. **Scalability:** The E2E model is suitable for large datasets with numerous input features, making this approach suitable to handle higher-dimentional data.

Disadvantages of an End-to-End Model

1. **Oversimplification:** While bypassing intermediate steps simplies decision-making, it may potentially neglect important domain-specific naunces. (Variability)
2. **Covariate:** Dependencis among variables might not be effectively addressed by th model. Without explicity addressing covariates, decisions can be suboptimal.
3. **Data Sentivitiy:** Noise or outliers in data can effect the model, resulting in inaccurate predictions and suboptimal results.
4. **Uncertainty:** An E2E model is less suited for handling uncertainties (variabilities).
5. **Black Box Nature:** Potential lack of transparency in understanding how E2E models process inputs to product outputs. (Layered Transformation, HIdden Feature Interactions, Overfitting, Lack Intermediate Steps)
6. **Adaptability:** Parameters may shift over time. Since E2E models are trained on historical data, it may not adapt well to changes unless retrained frequently.
7. **Biasness:** If the training data has biases, the E2E model can amplify these issues in its output.
8.

Utilizing Bayesian Framework to Mitigate Uncertainty in E2E Model

Potential hybrid approach by implementing some form of Bayesian framework to mitigate some uncertainty within an E2E model? Complexity? Cost?

Comments/Questions

- The relationship between feature-based newsvendor problems and quantile regression of demand?
- Challenges of replicating the results?
- From the paper, noted possibility of scaling and generalization where the proposed E2E model enables higher inventory-management accuracy with a lower operational cost and fewer labor efforts.
- Other papers to consider?
- What specifically is this end-to-end deep learning model? Ideal and objective of the function is well-defined within the paper, but how about the implementation part?