Bayesian Model-Based Offline Reinforcement Learning for Product Allocation

Porter Jenkins [1], **Hua Wei** [2], **J. Stockton Jenkins** [1], **Zhenhui Li** [3]

[1]Brigham Young University
[2]New Jersey Institute of Technology
[3]Pennsylvania State University

Abstract

Product allocation in retail is the process of placing products throughout a store to connect consumers with relevant products. Discovering a good allocation strategy is challenging due to the scarcity of data and the high cost of experimentation in the physical world. Some work explores Reinforcement learning (RL) as a solution, but these approaches are often limited because of the sim2real problem. Learning policies from logged trajectories of a system is a key step forward for RL in physical systems. Recent work has shown that model-based offline RL can improve the effectiveness of offline policy estimation through uncertainty-penalized exploration. However, existing work assumes a continuous state space and access to a covariance matrix of the environment dynamics, which is not possible in the discrete case. To solve this problem, we propose a Bayesian model-based technique that naturally produces probabilistic estimates of the environment dynamics via the posterior predictive distribution, which we use for uncertainty-penalized exploration. We call our approach Posterior Penalized Offline Policy Optimization (PPOPO). We show that our world model better fits historical data due to informative priors, and that PPOPO outperforms other offline techniques in simulation and against real-world data.

Introduction

The product allocation problem refers to the process of placing the right products in the right place at the right time in a physical retail space (Jenkins et al. 2020). In a typical scenario a manufacturer (e.g., Coca-cola) seeks to allocate product across the store by placing bids on discrete locations. The optimization goal of the manufacturer is to choose locations that maximize revenue subject to a budget constraint. Effective allocation strategies can increase revenue by connecting products with consumers. In many large stores, it can be difficult for consumers to find what they are looking for due to search costs. Proper product placement reduces search costs and improves convenience for the customer (Badgaiyan and Verma 2015)(Mattila and Wirtz 2008). Traditional operations research in this area uses mathematical models and heuristic search algorithms (Heragu et al. 2005) (Guerriero et al. 2012) to solve the problem from a theoretical standpoint. However, these methods are limited in a practical sense because they often make assumptions that deviate from real-world scenarios. Additionally, prior work focuses on logistics and warehouse allocation, which is fundamentally different from the retail scenario.

Recent work has studied the product allocation problem in retail using model-free RL (Jenkins et al. 2020), which requires training in simulation. However, policies trained with a simulator often fail to generalize to the real-world system due to the gap between the simulator and the real world (also called the sim2real problem) (Christiano et al. 2016). Most real-world systems are too complex to effectively simulate because they are only partially observable and high dimensional (Dulac-Arnold, Mankowitz, and Hester 2019). Moreover, training an allocation policy online is challenging because extensive exploration and experimentation is needed to learn a good policy, which is typically cost prohibitive due to potential lost revenue and the physical cost of moving products around the store. Therefore, learning a policy offline from logged trajectories of the system is essential.

Offline reinforcement learning is the task of learning policies from pre-recorded datasets without direct interaction with the environment; this approach represents a promising directions for scaling RL to real-world systems (Fujimoto, Meger, and Precup 2018)(Yu et al. 2020)(Dulac-Arnold, Mankowitz, and Hester 2019). In many domains, such static datasets are commonly accessible and contain sufficient statistical diversity of states for value estimation. While any off-the-shelf RL algorithm, such as DQN (Mnih et al. 2015), can be applied to static datasets, they struggle in the offline setting because without online exploration, they fail to generalize to out-of-distribution states and actions (Fujimoto, Meger, and Precup 2018). Model-based offline RL, in part, has shown to better generalize to out-of-distribution states because the agent's internal world model allows for offline exploration, branched from real data (Janner et al. 2019). To further address the out-of-distribution issue, (Yu et al. 2020) proposed Model-based Offline Policy Optimization (MOPO), which relies on an ensemble world model to produce probabilistic values of states and actions. The advantage of their approach is that the agent can perform offline exploration, but learn using an uncertainty penalized reward function to encourage the policy network to avoid states that the model is unsure about.

While MOPO appears to be a promising direction, it cannot be directly applied to the product allocation problem. MOPO assumes both states and reward follow a Gaussian distribution and computes model uncertainty using the covariance matrix of the model dynamics. However, the product allocation problem is formulated where most of the state space is discrete (e.g., product, i is in region, j). Therefore we cannot construct a covariance matrix to calculate the uncertainty-penalized reward function.

To this end, we propose a new model-based offline RL technique that allows for uncertainty-penalized exploration. Specifically, we build a Bayesian world model of the allocation problem that provides probabilistic value estimates of the state-action space. We build on the (MOPO) framework to penalize reward estimation with model uncertainty in the discrete state setting using the posterior predictive distribution. The advantage of our approach compared to (Yu et al. 2020), which we call Posterior Penalized Offline Policy Estimation (PPOPO), is 1) it can be used in the discrete state setting; 2) we can compute model uncertainty directly from the posterior without relying on computationally expensive model ensembles; and 3) we can easily incorporate economic knowledge and environment constraints through prior distributions.

We define our Bayesian world model and validate it against a real-world dataset. We show it is more predictive than off-shelf estimators such as neural networks due to informative priors. Using our trained world model, we then train a policy estimator offline and evaluate it in simulation, and against a real-world, static data.

In summary, the key are contributions of our paper are:

- We propose a new, model-based offline reinforcement learning algorithm for product allocation called PPOPO that uses the posterior predictive distribution as the uncertainty penalty.
- We build a world model that is trained via Bayesian inference and incorporates economic priors to constrain value estimation of states and actions.
- We show our world-model is more predictive of the true environment than off-the-shelf estimators.
- We show that our method can outperform other offline RL techniques both in simulation and against real-world, static data.

Preliminaries

In the following section, we provide a formal definition of the optimal allocation problem. Additionally, we define the necessary components of our reinforcement learning agent: the state space, action space, reward function, and state transition function.

Optimal Allocation Problem

In a physical retail environment $\mathcal{R}$ with a set of n spatial regions, we represent the environment with a spatial graph $\mathcal{R} = (\mathcal{V}, \mathcal{E})$, where each region $v_i \in \mathcal{V}$ is a vertex in the graph, the spatial neighboring relation between two regions v_i and v_j are represented as $e_{ij} \in \mathcal{V}$. From $\mathcal{G}$, we can construct the adjacency matrix, $\mathbf{A}$. Additionally, we observe a set of k products, $\mathcal{M} = \{m_j : 0 < j <= k\}$ that are sold. For each product, m_j, we know the retail price, p_j.

The decision process faced by the retailer or manufacturer is to allocate each product in $\mathcal{M}$ across regions in $\mathcal{V}$. We define the allocation policy as a function f:

$$f : \mathcal{V} \times \mathcal{M} \rightarrow \mathcal{Z} \tag{1}$$

$$\mathcal{Z} = \{\langle v_i, p_j \rangle, ...\langle v_w, p_q \rangle\} \tag{2}$$

Where $\mathcal{Z}$ is the set of selected product placements, such that $w <= n$, $q <= k$ and $\mathcal{Z} \subseteq \mathcal{V} \times \mathcal{M}$. This function is typically dynamic over time, which we denote as f^t. To simplify computation, we treat $\mathcal{Z}^t$ as an $(n \times k)$ grid and refer to it as the board configuration at time, t. An optimal retail strategy is to find the allocation policy that maximizes revenue:

$$f^* = \sum_{t}^{T} {}_{f^t} \sum_{i,j \in f^t(\mathcal{R},\mathcal{M})} p_j q_{ij} \tag{3}$$

where p_j is the price for product m_j, and q_{ij} is the quantity of product m_j sold in region v_i and T is the future time horizon of analysis.

Optimal Allocation as a Markov Decision Process

The optimal allocation problem is well suited for reinforcement learning because the RL agent is designed for sequential decision making that maximizes expected discounted reward over time. We frame the inputs as a Markov Decision Process (MDP) in a similar fashion as (Jenkins et al. 2020). An MDP is defined by the tuple $\langle \mathcal{S}, \mathcal{A}, P, r, \delta \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the set of possible actions, P is the (typically unkown) state transition function, r is the reward function and $\delta \in [0, 1]$ is the discount factor.

At each time, t, we observe the state of the retail environment, $\mathcal{R}$. We define the state, $s_t \in \mathcal{S}$, as the tuple of state features, $s_t = \langle \mathcal{Z}^t, d^t, \mathbf{g}^{(t-1)} \rangle$, where $\mathcal{Z}^t$ is the current board configuration, d^t is the current day of the week (e.g., Sunday $\rightarrow$ 0), and $\mathbf{g}^{(t-1)}$ is a vector denoting the discretized revenue at the previous time. We bin previous revenue, $\rho^{(t-1)} = p_j q_i^{(t-1)}$, into quintiles determined from the revenue distribution from logged interactions.

We define the action space $\mathcal{A} = \mathcal{R} \times \mathcal{M} \times \{-1, 1\} \cup \{0\}$, indicating "to place", "take way" or "do nothing" for each product, m_j in each region, v_i.

The reward function is the total product revenue at time t, constrained by the monetary cost, c, of placing a set of products in each region:

$$r(t) = \sum_{i=1}^{n} \sum_{j=1}^{k} p_j q_{ij}^t - c \sum_{i=1}^{n} 1_{\mathcal{Z}}(v_i) \tag{4}$$

The second term in the reward function accounts for the cost faced by the distributor who typically has to pay for each space in a retail environment.

The state transition, P is defined as $p(s^{t+1}|s^t, a^t) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$, which gives the probability of moving to state, $s^{(t+1)}$ given the current state and action. In the optimal allocation problem the exact transition function, P, is

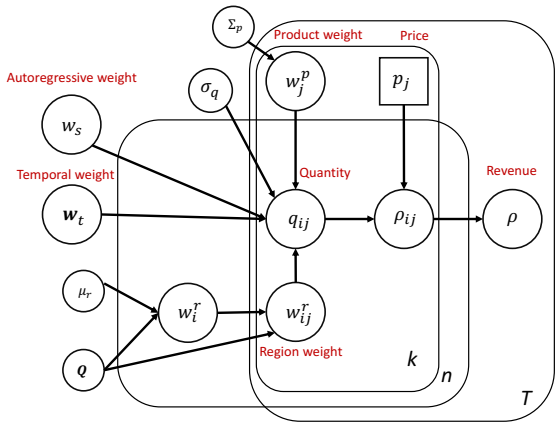

Figure 1: An overview of our Bayesian world model. The boxes are "plates" representing structures in the data. The plates marked by k, n and T represent products, regions, and time, respectively. Circles denote random variables and squares are deterministic quantities. We decompose quantity as a function of region, product, time, and auto-regressive weights. We incorporate prior information in three ways: 1) a hierarchical structure that allows region-level weights to vary by product. 2) a spatial covariance prior for region weights, and 3) structured covariance for product substitution effects.

unknown since the current state, s^t depends on the results of the previous time, $\mathbf{g}^{(t-1)}$. We model this transition as a stochastic process.

Bayesian World Model

In this section, we outline our proposed environment model and discuss the informative priors used to help accurately estimate values of states.

Stochastic Model of Spatial Demand

We propose the following stochastic model of spatial demand in physical retail. See Figure 1 for an overview. The key advantages of using a Bayesian model in the optimal product allocation problem is the ability to use informative priors, and produce probabilistic estimates of reward. We incorporate prior information in three ways: 1) a hierarchical structure that allows region-level weights to vary by product. 2) a spatial covariance prior for region weights, and 3) learned covariance for product substitution effects.

Revenue at each time step, $\rho(t)$ is modeled is as the sum of revenue at each region, $\rho(t) = \sum_{i=1}^{n}\sum_{j=1}^{k} p_j q_{ij}^{(t)}$.

The key variable of interest is, $q_{ij}^{(t)}$, the quantity sold for product, m_j, region, v_i, at time, t. We model $q_{ij}^{(t)}$ as a truncated normal random variable: $q_{ij}^{(t)} \sim \psi(\mu, \sigma, a, b)$.

where, $\psi(\mu, \sigma, a, b)$ is the pdf of the truncated normal distribution. See (Burkardt 2014) for more details. The truncated normal distribution assures that predicted quantity estimates are non-negative. We set $a = 0$ and $b = +\infty$, which forces $\Phi(\mu, \sigma^2; b) = 1$ and constrains the quantity, $q_{ij}^{(t)} \in R^+$. The prior for $q_{ij}^{(t)}$ is characterized by the mean, μ_q, which is a linear function of environment features and learned weights, $\mu_q = \mathbf{x}^\top \mathbf{w} + b$, and the inverse gamma distribution for the variance, $\sigma_q \sim \text{IG}(\alpha_q, \beta_q)$.

In our environment, we observe temporal features, $\mathbf{x}_t$, region features, $\mathbf{x}_r$, product features, $\mathbf{x}_p$, and autoregressive features, $\mathbf{x}_s$: $\mathbf{x} = [\mathbf{x}_t, \mathbf{x}_r, \mathbf{x}_p, \mathbf{x}_s]^\top$. We discuss our feature extraction approach in more detail below.

Region-level Weights One major advantage of Bayesian modeling is the natural ability to exploit the structure inherent in many datasets for parameter sharing. It can also allow for discovery of more fine-grained spatial effects of the environment.

We allow for heterogeneous region weights, w_{ij}^r, which denotes the impact of region, r_i, given product, m_j, on $q_{ij}^{(t)}$.

$$\mathbf{w}_{ij}^r \sim \mathcal{N}(\mathbf{w}_r, \mathbf{Q}_r), \mathbf{w}_r \sim \mathcal{N}(\boldsymbol{\mu}_r, \mathbf{Q}_r) \tag{5}$$

$$\mathbf{Q}_r = \gamma \tilde{L} = \gamma(\mathbf{I} - \hat{\mathbf{D}}^{-\frac{1}{2}} \hat{\mathbf{A}} \hat{\mathbf{D}}^{-\frac{1}{2}}) \tag{6}$$

We are able to encode graph information into our prior for $\mathbf{w}_r$ and $\mathbf{w}_{ij}^r$ by computing the adjacency matrix, $\mathbf{A}$ from $\mathcal{G}$ and computing the normalized graph Laplacian, $\tilde{L}$. Doing so allows us to put a spatial prior over the variance of the region weights. We use this as our precision matrix (Dong et al. 2016) times a scale factor, γ. The normalized graph Laplacian ensures that the precision matrix, $\mathbf{Q}$ is symmetric and positive semidefinite.

Note that both $\mathbf{w}_r$ and $\mathbf{w}_{ij}^r$ share the same same covariance structure: the precision matrix defined by the graph laplacian (Equation 6). Thus, the region weights are only hierarchical in their means. Additionally, we treat the upper-level mean vector, $\boldsymbol{\mu}_r$ as a hyperparameter.

Product-level Weights We also define weights for each product, m_j, as follows:

$$\mathbf{w}_p \sim \mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p) \tag{7}$$

We assume the product weights have mean, $\boldsymbol{\mu}_p$ with a structured covariance prior $\boldsymbol{\Sigma}_p$ that models product substitution effects. Learning a covariance matrix for substitution effects are discussed more below.

Temporal weights The temporal features capture the long-term and short-term seasonality of the environment. The temporal weights are defined similar to the product weights. Namely, the temporal weights, $\mathbf{w}_t$, follow a multivariate normal distribution: $\mathbf{w}_t \sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$.

We put an LKJ prior over the covariance matrix, $\boldsymbol{\Sigma}_t = \mathbf{L}\mathbf{L}^\top \sim \text{LKJ}(\sigma_t)$, and reparameterize $\boldsymbol{\Sigma}_t$ as its cholesky decomposition, $\mathbf{L}\mathbf{L}^\top$, so that the underlying correlation matrices follows an LKJ distribution (Lewandowski, Kurowicka, and Joe 2009). The standard deviations, σ_p, follow a half-cauchy distribution. The advantage of the LKJ prior is that it is more computationally tractable than other covariance priors (Lewandowski, Kurowicka, and Joe 2009).

Autoregressive weight Finally, we specify the weight of previously observed revenue values on $q_{ij}^{(t)}$. The feature, $\mathbf{x}_s$ is an autoregressive feature denoting the previous value of product-level revenue. We specify a truncated normal distribution for $w_s \sim \psi(\mu_s, \sigma_s, a, b)$, and half cauchy priors for the location, μ_s and scale, σ_s.

Learning Product Substitution Effects

One important challenge in the allocation problem is accurately capturing product relationships, or substitution effects. Both theoretical and empirical economics demonstrates that products tend to exhibit either complementary or supplementary effects (Mas-Colell and Whinston 1995). In the case of complements, two goods "go together" if the presence of one increases the demand for the second. For supplements, the presence of one good decreases the demand for the other. Many common retail datasets are aggregated and do not reveal individual transactions. Consequently, learning substitution effects from aggregated data can be challenging.

To solve this problem, we propose to incorporate information from both offline retail data, and publicly available online review data. From the online data we observe product reviews within a user. We use this information to learn general product substitution effects, which is then used to seed the covariance matrix for the product weights, $\mathbf{\Sigma}_p$. Given a set of of users $\mathcal{U} = \{u_1, ...u_U\}$, we define our measure of product correlation:

$$\alpha_{ij} = \frac{\sum_{u=1}^{U} s_i s_j 1_u(m_i, m_j)}{\sqrt{\sum_{i=1}^{k} s_i^2}\sqrt{\sum_{i=j}^{k} s_j^2}} \tag{8}$$

where $1_u(m_i, m_j)$ is the indicator that m_i and m_j are both reviewed by user u, and s_i and s_j are the review scores for m_i and m_j from user u. If two products, m_i and m_j are both reviewed by user u, then we multiply the review scores, s_i and s_j together to get a weighted co-occurence. We normalize the scores so that each score falls between 0 and 1. The normalization constant is the product of the square root of the sum of rating for m_i and m_j. Intuitively, when two items have high scores and co-occur often, α_{ij} approaches 1.

Training

We train the world model using the No U-Turn Sampler (NUTS) algorithm (Hoffman and Gelman 2011). This allows us to draw samples from the posterior distribution of model weights, $\mathbf{W}$, as well as the posterior predictive distribution of quantity, $q_{ij}^{(t)}$, and revenue $\rho^{(t)}$. We use Automatic Differention Variational Inference (ADVI) (Kucukelbir 2017) as an initialization point for the sampling procedure. All models are implemented in PyMC3.

We initialize with ADVI using 200,000 iterations. Once initialized we sample the posterior using NUTS with a tuning period of 5,000 draws followed by 5,000 samples across four chains.

Feature Extraction

In order to train our world model, we extract environment-level features, $\mathbf{x}$, which is composed of temporal features, $\mathbf{x}_t$, region features, $\mathbf{x}_r$, product features, $\mathbf{x}_p$, previous sales features and $\mathbf{x}_s$.

For temporal features, we use a one-hot vector denoting the day of the week for, $\mathbf{x}_t$. We again use a one-hot vector to encode spatial regions, $\mathbf{x}_r$. Additionally, we expect each product to vary in popularity. To capture this effect we also construct a one-hot vector for products, $\mathbf{x}_p$. Finally, we construct an autoregressive sales feature that represents the sales at time, $t-1$. We use the previous sales for product m_j, summed across all regions, $w_s = \rho_j^{(t-1)} = \sum_{i=1}^{k} p_j q_{ij}^{(t-1)}$. This feature captures micro-fluctuations in demand for each product.

PPOPO

Algorithm 1: PPOPO with Bayesian World Model

World model, $f(s, a)$, trained on batch data, $\mathcal{D}_{env}$, epoch length, N, reward penalty parameter, λ, rollout horizon, h, and rollout batch size b trained policy, π
Initialize policy π, and empty replay buffer, $\mathcal{D}_{model} \leftarrow \emptyset$
for *Epoch* $1, 2, ...N$ **do**
  for *1, 2, ..., b* **do**
    sample state, $s_j \sim \mathcal{D}_{env}$ to initialize rollout
    for *j = 1, 2, ..., h* **do**
      sample action from policy: $a_j \sim \pi(s_j)$
      Compute posterior predictive distribution and next state: $p(\hat{\rho}), s_{j+1} = f(s_j, a_j)$
      Compute penalized reward: $\tilde{r}_j = \overline{p(\hat{r})} - \lambda\sigma_{p(\hat{r})}$
      Add tuple $(s_j, a_j, \tilde{r}_j, s_{j+1})$ to $\mathcal{D}_{model}$
    end
    Draw samples from $\mathcal{D}_{model} \cup \mathcal{D}_{env}$ to update π
  end
end

We model the environment dynamics using the Bayesian world model described above. The advantage of model-based RL for offline learning is, given an accurate world model, the agent can perform rollouts and potentially discover states not seen in the static dataset. Additionally, the uncertainty penalty encourages the agent to avoid states that the model is unsure about.

The key differences between PPOPO and prior work is in our construction of a single, probabilistic world model and our uncertainty penalty. Rather than build an ensemble of neural networks, our Bayesian model is capable of probabilistic uncertainty quantification via the posterior predictive distribution, $p(\hat{r})$. The uncertainty penalized reward estimate of PPOPO is:

$$\tilde{r} = \overline{p(\hat{r})} - \lambda\sigma_{p(\hat{r})} \tag{9}$$

where $\overline{p(\hat{r})}$ is the mean of the posterior predictive distribution of reward, r. Thus, we penalize the mean reward by

Model	Store 1 MAE	Store 1 MAPE	Store 2 MAE	Store 2 MAPE
OLS	29.10	0.3932	33.16	0.3546
KNN	26.53	0.3709	33.32	0.3361
SVR	29.00	0.3637	35.97	0.3446
RF	24.33	0.3265	31.06	0.2986
GBRT	27.24	0.3641	31.66	0.3133
MLP	25.10	0.3350	31.60	0.2962
World Model	**19.39**	**0.3218**	**19.92**	**0.2862**

Table 1: Store-level revenue prediction against test data (April 1, 2019 to August 31, 2019). Our Bayesian world model is able to better forecast revenue than off-the-shelf estimators in both stores. In addition to being more accurate, our world model provides uncertainty quantification of state-action values.

the standard deviation, or uncertainty, of the estimate. We treat λ as a hyperparameter.

Experiments

In this section, we first evaluate the effectiveness of the world model at predicting revenue on a static, test dataset. We then perform policy evaluation experiments where we examine the effectiveness of a policy trained with PPOPO from static data. To this end, we test the learned policy in simulation (Jenkins et al. 2020) and against observed trajectories using the Per-State Rejection Sampling (PSRS) evaluator (Mandel et al. 2016). Training details and hyperparameters are discussed in the supplementary materials.

Dataset Decription

In the following section we describe in detail our offline retail dataset, and the online, product review dataset.

Retail Data Our retail dataset is comprised of three primary entities: stores, products, and regions. We collect data from Swire Coca-Cola, a large Coca-Cola distributor in the western United States. The data is comprised of two large supermarket and retail stores in Salt Lake City, UT, USA. Each store primarily sells groceries, common household goods and clothing. We observe quantities sold for a set of 15 products, as well as each product's average price over the year. All of the products in our dataset are Coca-cola brand beverage products. The product set includes items such as "Coca-Cola 8 oz - 12 pack", and "Sprite 2 Liter bottle".

The data provides daily counts of quantities at the region-product level. Additionally, the locations of the products are varied in product "displays". Store 1 is comprised of 17 regions, and store 2 has 12. Each region represents a section of the store. In general regions tend to be constructed based on the function of each space (e.g., pharmacy, deli, etc.). We construct a spatial graph of these regions.

We partition the time series into a training period from August 1, 2017 - March 31, 2019 , and a test period of April 1, 2019 to August 31, 2019. We will use these to train and evaluate our world model. Additionally, we use the same test data for our policy experiment using the PSRS evaluator.

Online Review Data In addition to the retail dataset, we use the publicly available Amazon 18 dataset (McAuley et al. 2015) to learn product substitution effects and structure our covariance matrix. Because all of the products in the retail data are Coca-Cola products, we filter the dataset to three categories for our experiments: grocery and gourmet food, prime pantry, and home and kitchen.

World Model Evaluation

We report the results in Table 1. Our Bayesian world model is overall more accurate at predicting future states than baselines (using posterior predictive mean). We observe that error is minimized across both metrics and datasets. In both stores, the reduction in MAE offered by the Bayesian world model is significant. In store 1, we see a 20% decrease in MAE from the next best model; for store 2 we similarly see a 36% decrease. Additionally, in both stores the average error falls within approximately $19 of the true value. It appears that a combination of the informative priors from spatial and product similarities, in addition to the hierarchical parameter sharing structure, allows the model to better learn the underlying demand (see appendix for ablation study exploring this result).

Policy Evaluation

We compare PPOPO to existing offline policy estimators using two strategies. First, we train each policy offline using a static dataset and then evaluate using a spatial demand simulator (Jenkins et al. 2020). Second, because simulation-based experiments may not exactly reflect the real-world we also evaluate the trained policy using a static dataset and the Per-state Rejection Sampling Evaluator (Mandel et al. 2016).

Baseline Policy Estimators We compare the policy learned using PPOPO to DQN trained offline and Batch Constrained Q-Learning (BCQ) (Fujimoto, Meger, and Precup 2018). BCQ is an offline algorithm that constrains the action space towards the actions selected in a batch of logged data.

Simulation Experiment Our first policy experiment seeks to test how well a policy trained offline will generalize to the online setting. We train each policy with the offline training dataset. For each store, we vary the episode length (30, 60, and 90 days) to assess the effectiveness of short and long term strategies. We perform each episode 10 times and report the mean and standard deviation of cumulative reward in Table 2.

In general, we see that PPOPO outperforms both Off-policy DQN and BCQ under all settings. Interestingly, BCQ seems to perform better than Off-policy DQN in store 1, but not store 2. The standard deviations indicate that the cumulative reward distributions do not overlap and are statistically significant. Additionally, as the episode length increases so does the relative improvement of PPOPO over the next best

	Store 1			Store 2		
	$l = 30$	$l = 60$	$l = 90$	$l = 30$	$l = 60$	$l = 90$
Off-policy DQN	23.4 (.11)	43.2 (.14)	68.9 (.39)	13.5 (.44)	22.6 (.15)	27.5 (.20)
BCQ	23.7 (.29)	49.4 (2.26)	74.5 (3.87)	8.27 (.26)	15.41 (.32)	23.50 (1.08)
PPOPO (ours)	**35.8** (.20)	**70.3** (.23)	**105.0** (.41)	**16.14** (.13)	**32.00** (.23)	**45.23** (.11)

Table 2: A comparison of offline RL algorithms across store 1 and store 2 by total cumulative reward (in thousands of $). We vary the episode length, l, in 30 day increments (i.e., $l = 30$, $l = 60$, and $l = 90$ days in the future). PPOPO algorithm is superior in all cases.

policy. For example, in store 2 going from 30 to 90 days increases the relative improvement in reward from 19.5% to 41.6%.

Rejection-sampling Experiment Lastly, we compare the learned policy to a real-world policy used to generate our test dataset. In order to study how the learned policies might perform in the real environment, we compare them using the Per-State Rejection Sampling (PSRS) (Mandel et al. 2016) evaluator, which is shown to be an unbiased estimator of the true environment. PSRS requires both a learned policy, π_b and an estimate of the online sampling distribution, π_e. We are more likely to accept an output tuple (a, s') if the two policies are similar. If a tuple is accepted, the learned policy, π_b, receives the observed reward from the test data, times a discount factor, γ. A policy that closely approximates true decisions made in the dataset will receive high reward. In the product allocation problem, we don't have direct access to π_e, but we can approximate it from the test data using a neural net, $\pi_e(a|s)$ trained to predict the probability of the next action given the state. We run ten episodes of the PSRS evaluator with a discount factor, $\gamma = .8$.

We report the mean and standard deviations of discounted, cumulative reward in Table 3. We see that PPOPO outperforms both baselines in each store. The off-policy DQN struggles to mirror the real-world policy from the test set as none of the action proposals are accepted by the sampler. This indicates large deviations between the two policies. We conclude that PPOPO more closely approximates decisions made in the true environment.

Discussion of Real-world Implications

Reinforcement Learning has shown the ability to learn superhuman policies in simulated environments where data is essentially unlimited and consequences of poor actions are non-existent; getting RL to work in the real-world is much more challenging (Dulac-Arnold, Mankowitz, and Hester 2019). One primary reason is the difficulty in building accurate simulators for complex, real-world environments. We acknowledge that the methods proposed in the current work may still be difficult to implement in a real-world, retail setting. However, we believe PPOPO tackles a key challenge in real-world RL: training offline from fixed logs of real-world behavior.

	Store 1	Store 2
Off-policy DQN	0.0	0.0
BCQ	1.43 (.32)	1.13 (.43)
PPOPO	**5.71** (.60)	**1.96** (.77)

Table 3: Per-state Rejection Sampling (PSRS) evaluation

Related Work

Shelf Space Allocation: Some classical work seek to optimize shelf space allocation via dynamic programming algorithm (Zufryden 1986) and simulated annealing (Borin, Farris, and Freeland 1994). More recent work includes product orientation within a shelf (Murray, Talukdar, and Gosavi 2010) as a decision variable. Frequent pattern mining algorithms have also been proposed to allocate product shelf space (Aloysius and Binu 2011). These existing studies all focus on micro-regions (shelves) within the retail environment and not macro-level patterns across the store.

Offline RL Two general strategies have been proposed for offline RL: 1) imposing constraints such that the learned policy is closer to the policy observed in the logged dataset (Fujimoto et al. 2019)(Fujimoto, Meger, and Precup 2018), and 2) using uncertainty quantification to stabilize Q-Functions (Yu et al. 2020)(Agarwal, Schuurmans, and Norouzi 2019). PPOPO builds on uncertainty quantification methods and extends them to discrete states using the posterior predictive distribution.

Model-based RL A body of prior work exists (Wang et al. 2019) proposing model-based online learning methods including linear models (Levine and Koltun 2013), Gaussian processes (Deisenroth and Rasmussen 2011), and neural nets (Kaiser et al. 2019). Our work adapts this line of research to the offline case for discrete state problems such as product allocation.

Conclusion

In this paper, we study the problem of product allocation in physical retail. In contrast to previous work that seeks to learn model-free allocation policies trained in simulation, we propose a novel training technique called Posterior Penalized Offline Policy Optimization (PPOPO). We construct a world model that incorporates informative economic priors and constraints to enable better generalization to out-of-distribution states.

References

Agarwal, R.; Schuurmans, D.; and Norouzi, M. 2019. An Optimistic Perspective on Offline Reinforcement Learning. arXiv:1907.04543.

Aloysius, G.; and Binu, D. 2011. An approach to products placement in supermarkets using PrefixSpan algorithm. *Journal of King Saud University - Computer and Information Sciences*.

Badgaiyan, A.; and Verma, A. 2015. Does urge to buy impulsively differ from impulsive buying behaviour? Assessing the impact of situational factors. *Journal of Retailing and Consumer Services*.

Borin, N.; Farris, P. W.; and Freeland, J. R. 1994. A model for determining retail product category assortment and shelf space allocation. *Decision Sciences*.

Burkardt, J. 2014. The truncated normal distribution. *Department of Scientific Computing Website, Florida State University*, 1–35.

Christiano, P.; Shah, Z.; Mordatch, I.; Schneider, J.; Blackwell, T.; Tobin, J.; Abbeel, P.; and Zaremba, W. 2016. Transfer from Simulation to Real World through Learning Deep Inverse Dynamics Model. arXiv:1610.03518.

Deisenroth, M. P.; and Rasmussen, C. E. 2011. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. In *In Proceedings of the International Conference on Machine Learning*.

Dong, X.; Thanou, D.; Frossard, P.; and Vandergheynst, P. 2016. Learning Laplacian Matrix in Smooth Graph Signal Representations. In *IEEE Transactions on Signal Processing*.

Dulac-Arnold, G.; Mankowitz, D.; and Hester, T. 2019. Challenges of Real-World Reinforcement Learning. arXiv:1904.12901.

Fujimoto, S.; Conti, E.; Ghavamzadeh, M.; and Pineau, J. 2019. Benchmarking Batch Deep Reinforcement Learning Algorithms. arXiv:1910.01708.

Fujimoto, S.; Meger, D.; and Precup, D. 2018. Off-Policy Deep Reinforcement Learning without Exploration. arXiv:1812.02900.

Guerriero, F.; Musmanno, R.; Pisacane, O.; and Rende, F. 2012. A mathematical model for the Multi-Levels Product Allocation Problem in a warehouse with compatibility constraints. *Applied Mathematical Modelling*.

Heragu, S. S.; Du, L.; Mantel, R. J.; and Schuur, P. C. 2005. Mathematical model for warehouse design and product allocation. *International Journal of Production Research*, 43(2): 327–338.

Hoffman, M. D.; and Gelman, A. 2011. The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*.

Janner, M.; Fu, J.; Zhang, M.; and Levine, S. 2019. When to Trust Your Model: Model-Based Policy Optimization. arXiv:1906.08253.

Jenkins, P.; Wei, H.; Jenkins, J. S.; and Li, Z. 2020. A Probabilistic Simulator of Spatial Demand for Product Allocation. arXiv:2001.03210.

Kaiser, L.; Babaeizadeh, M.; Milos, P.; Osinski, B.; Campbell, R. H.; Czechowski, K.; Erhan, D.; Finn, C.; Kozakowski, P.; Levine, S.; Mohiuddin, A.; Sepassi, R.; Tucker, G.; and Michalewski, H. 2019. Model-Based Reinforcement Learning for Atari. arXiv:1903.00374.

Kucukelbir, e. a. 2017. Automatic Differentiation Variational Inference. *Journal of Machine Learning Research*.

Levine, S.; and Koltun, V. 2013. Guided policy search. In *International conference on machine learning*, 1–9. PMLR.

Lewandowski, D.; Kurowicka, D.; and Joe, H. 2009. Generating random correlation matrices based on vines and extended onion method. *Journal of Multivariate Analysis*.

Mandel, T.; Liu, Y. e.; Brunskill, E.; and Popovic, Z. 2016. Offline Evaluation of Online Reinforcement Learning Algorithms. In *Proceedings of the 2016 conference of Association for the Advancement of Artificial Intelligence (AAAI'16)*.

Mas-Colell, A.; and Whinston, M. D. 1995. *Microeconomic Theory*. Oxford University Press.

Mattila, A.; and Wirtz, J. 2008. The role of store environmental stimulation and social factors on impulse purchasing. *Journal of Services Marketing*.

McAuley, J.; Targett, C.; Shi, Q.; and Van Den Hengel, A. 2015. Image-based Recommendation on Styles and Substitutes. In *SIGIR*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533.

Murray, C. C.; Talukdar, D.; and Gosavi, A. 2010. Joint Optimization of Product Price, Display Orientation and Shelf-Space Allocation in Retail Category Management. *Journal of Retailing*.

Wang, T.; Bao, X.; Clavera, I.; Hoang, J.; Wen, Y.; Langlois, E.; Zhang, S.; Zhang, G.; Abbeel, P.; and Ba, J. 2019. Benchmarking Model-Based Reinforcement Learning. arXiv:1907.02057.

Yu, T.; Thomas, G.; Yu, L.; Ermon, S.; Zou, J.; Levine, S.; Finn, C.; and Ma, T. 2020. MOPO: Model-based Offline Policy Optimization. arXiv:2005.13239.

Zufryden, F. S. 1986. A Dynamic Programming Approach for Product Selection and Supermarket Shelf-Space Allocation. *The Journal of Operational Research Society*.