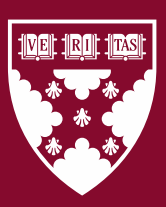
Working Paper 25-037

Dynamic Personalization with Multiple Customer Signals: Multi-Response State Representation in Reinforcement Learning

Liangzong Ma
Ta-Wei Huang
Eva Ascarza
Ayelet Israeli



Harvard Business School

Dynamic Personalization with Multiple Customer Signals: Multi-Response State Representation in Reinforcement Learning

Liangzong Ma
Harvard Business School

Ta-Wei Huang
Harvard Business School

Eva Ascarza
Harvard Business School

Ayelet Israeli
Harvard Business School

Working Paper 25-037

Funding for this research was provided in part by Harvard Business School.

Dynamic Personalization with Multiple Customer Signals: Multi-Response State Representation in Reinforcement Learning

Liangzong Ma
Harvard Business School, lma@hbs.edu

Ta-Wei Huang
Harvard Business School, thuang@hbs.edu

Eva Ascarza
Harvard Business School, eascarza@hbs.edu

Ayelet Israeli
Harvard Business School, aisraeli@hbs.edu

Abstract. Reinforcement learning (RL) offers potential for optimizing sequences of customer interactions by modeling the relationships between customer states, company actions, and long-term value. However, its practical implementation often faces significant challenges. First, while companies collect detailed customer characteristics to represent customer states, these data often contain noise or irrelevant information, obscuring the true customer states. Second, existing state construction techniques focus primarily on summarizing characteristics related to short-term values, rather than capturing the broader behaviors that drive long-term customer value. These limitations hinder RL's ability to effectively learn customer dynamics and maximize long-term value. To address these challenges, we introduce a novel *Multi-Response State Representation (MRSR) Learning* method to enhance existing RL methods. Unlike state construction methods, MRSR utilizes rich customer signals — such as recency, engagement, and spending — to construct low-dimensional state representations that effectively predict behaviors driving long-term customer value. Using data from a free-to-play mobile game with dynamic difficulty adjustments, MRSR demonstrates significant improvements, increasing 30-day in-game currency spending by 37% compared to standard offline RL methods and 24% over advanced state representation techniques. Policy interpretation further highlights MRSR's ability to identify distinct and relevant customer states, enabling precise and targeted interventions to boost long-term engagement and spending.

Key words: Dynamic Policy, Deep Reinforcement Learning, Customer Relationship Management, Representation Learning, Dynamic Difficulty Adjustment, Latent Variable Model

History: This version: February 5, 2025.

1. Introduction

Marketing personalization is widely adopted, with many companies tailoring promotions (Simester et al. 2020, Huang and Ascarza 2024), communications (Ellickson et al. 2022), and product designs (Hauser et al. 2009, Yoganarasimhan et al. 2023) to individual customers. Most existing personalization methods are designed for one-time decision-making: they optimize a single, immediate action based on static customer data. However, customers typically engage with companies over multiple interactions, creating a dynamic relationship that evolves over time. For instance, customers might receive coupons repeatedly from an e-commerce platform (Liu 2023), view multiple ads on a video platform (Rafieian 2023), or interact with

various product recommendations. Each interaction presents an opportunity to personalize the experience, optimizing the customer's future engagement and strengthening their overall relationship with the company.

Reinforcement learning (RL) provides a practical framework for managing sequential customer interactions by focusing on the long-term impact of each decision. Unlike traditional methods that focus solely on a single marketing action, RL dynamically personalizes a sequence of actions to optimize long-term objectives, such as cumulative spending over an extended period. One of RL's key advantages is its ability to leverage offline data — historical interactions and customer responses — allowing organizations to refine strategies without incurring the cost of real-time experimentation. To implement RL with historical data, companies need an initial policy that determines actions (e.g., offering a promotion or presenting a specific product version) using predefined business rules (based on customer characteristics). Additionally, companies must gather detailed customer characteristics, such as demographics and behavioral patterns, prior to the action, as well as record immediate rewards (e.g., spending) following the action. These inputs enable RL models to estimate the long-term value of different actions across customers with different characteristics and dynamically optimize policies to maximize long-term values. RL is increasingly gaining traction in marketing applications, including dynamic coupon targeting (Liu 2023), adaptive ad sequencing (Rafieian 2023), and dynamic pricing (Wang et al. 2023b), offering a promising avenue for data-driven decision-making in customer engagement.

Despite its potential, reinforcement learning (RL) often encounters critical challenges in real-world applications, leading to suboptimal policies. A key challenge lies in the noisy and potentially irrelevant nature of customer data, which can significantly reduce the efficiency of the offline RL process. Specifically, RL's success relies on accurately representing the "customer relationship state" — a representation that reflects the future value of customers as influenced by the company's actions. While companies typically collect extensive customer characteristics (e.g., past interactions, demographics, and behavioral metrics) to account for heterogeneity in customer relationship states, much of this information may be irrelevant or fail to capture the key factors that moderate customer responses to actions. This results in a state representation that inadequately reflects the true customer relationship state before an action is delivered, hindering efficient learning and leading to suboptimal policy decisions. Prior research (Gelada et al. 2019, Zhang et al. 2021, Hong et al. 2023) has documented this challenge and highlights the importance of effective state representation learning as a way to reduce irrelevant information and improve RL performance.

Moreover, maximizing long-term customer value requires RL methods to leverage state representations that accurately capture the dynamics of long-term customer behaviors. However, many existing state representation learning techniques fail to meet this need. These methods often focus on summarizing correlations in high-dimensional customer characteristics (Allen et al. 2021) or predicting immediate rewards (Gelada et al. 2019), neglecting the complex interactions between company actions and the diverse customer behaviors that drive long-term value. This shortcoming prevents companies from fully utilizing RL to improve customer retention and engagement, thereby limiting their ability to maximize long-term customer value.

In this research, we introduce a novel method called *Multi-Response State Representation (MRSR) Learning* to address the challenges of learning state representation in RL. Specifically, we develop a *deep multi-task state representation learning* model that transforms high-dimensional customer characteristics into low-dimensional latent representations. The model is trained with two key objectives: (i) capturing information relevant to multiple short-term behaviors that collectively drive long-term customer value, and (ii) preserving the *Markov property* to ensure the state representation sufficiently summarizes past interactions and customer dynamics. To achieve the first objective, we supervise representation learning by predicting both the immediate reward (e.g., customer spending in the current period) and additional responses (e.g., retention and engagement proxies). These additional responses capture not only short-term rewards but also behaviors critical to driving long-term value, enabling the model to retain information essential for sustained customer engagement. For the second objective, we incorporate auxiliary prediction tasks proposed by Allen et al. (2021) to ensure that the latent state representation adheres to the Markov property and effectively summarizes a customer's historical information.

We evaluate our proposed method using data from a free-to-play mobile puzzle game, where the company seeks to implement a dynamic difficulty adjustment (DDA) policy to maximize players' coin spending (in-game currency) over a 30-day period. Identifying an effective policy to increase spending is crucial for the company's success, as coin purchases are its primary revenue source. However, constructing state representations that capture long-term spending behavior is challenging, given the infrequency of spending events — a common trait of free-to-play games. To address this, we go beyond conventional approaches by incorporating two additional customer responses — time until the next play (a proxy for retention) and rounds played per day (a proxy for engagement) — which complement short-term spending and provide a more robust foundation for capturing long-term customer value.

Using data from a large-scale field experiment with predefined difficulty adjustment rules set by the company, we demonstrate that combining state-of-the-art offline RL with our proposed MRSR method can double 30-day in-game currency spending — the primary outcome of interest — compared to the company's current implementation. Furthermore, offline RL based on the multi-response state representation outperforms standard offline RL using observed player characteristics, achieving a 37% increase in 30-day coin spending. We also evaluate a constrained version of our model, where the state representation is derived solely from immediate coin spending, as proposed by Gelada et al. (2019). Compared to the single-response representation, the proposed MRSR improves outcomes by 24%. These findings highlight the value of integrating multiple behavioral signals into state representation learning to enhance long-term customer value.

Leveraging the learned state representation, we conduct several policy interpretation analyses to showcase how the MRSR policy differs from benchmark policies. First, we analyze variations in the state space across different state representation learning methods by projecting observed player characteristics and state representations into a two-dimensional space for clustering analysis. The results show that both single-response

and multi-response state representations identify significantly fewer latent player segments compared to clustering based on observed characteristics — yet, outperform the latter in maximizing coin spending. This suggests that these methods effectively distill information by filtering out irrelevant details that do not impact player behavior under dynamic difficulty adjustments. Notably, the multi-response state representation identifies a distinct segment of players who were moderately active and performed well over the past week but showed signs of disengagement or performance decline on the prior day. Second, we compare the resulting policy recommendations and find that the MRSR policy specifically adjusts game difficulty to re-engage these recently disengaged players. In contrast, policies based on observed characteristics or single-response representations fail to detect these short-term behavioral shifts, instead making decisions based on aggregated behaviors over the past week. As a result, they do not make targeted adjustments for players at risk of disengagement. These findings showcase the MRSR policy's strength in identifying key states for sustaining long-term player engagement.

Our research makes two key contributions to the literature. Methodologically, we propose a novel approach that enables companies to design dynamic personalization policies optimized for long-term outcomes using high-dimensional customer data. Specifically, our representation learning method captures customer relationship states that drive the long-term customer value. This approach is broadly applicable across industries where dynamic adaptation of product features or sequential marketing interventions is important. Examples include website morphing to enhance user engagement over time (Liberali and Ferecatu 2022), adjusting user experience elements to promote long-term retention (Ko et al. 2024), and dynamically delivering targeted coupons to maximize customer value (Liu 2023).

Substantively, we show that gaming companies can significantly increase cumulative player spending by dynamically adjusting game difficulty at an individualized level (Ascarza et al. 2025). Beyond this, we provide a practical framework for optimizing product adjustments, emphasizing the role of intertemporal dynamics — factors often overlooked in existing product interaction design literature (Hauser et al. 2009, 2014, Li et al. 2021). Our model interpretation analysis sheds light on the mechanisms driving these outcomes. For example, highly engaged players may lose interest if the game becomes too easy, while recently inactive players may benefit from reduced difficulty, which can help re-engage them. Crucially, our approach highlights the importance of distinguishing between two types of recently inactive players: those who are stuck and those who are bored. Reducing game difficulty can re-engage stuck players by alleviating frustration but may further disengage bored players by removing the challenge. This nuanced understanding of player dynamics demonstrates the value of our method in crafting tailored strategies that enhance both engagement and long-term customer value.

The remainder of the paper is organized as follows. Section 2 reviews the relevant literature. Section 3 introduces the company's dynamic product adjustment problem aimed at maximizing long-term customer value and outlines the empirical challenges involved. Section 4 presents the proposed solution. Section 5

discusses the empirical business context and data. Section 6 presents the benchmarks for comparison, the offline policy evaluation method, and the performance of the proposed model relative to these benchmarks. Section 7 interprets the learned policies and explores their managerial implications. Section 8 concludes and suggests directions for future research.

2. Related Literature

Our research relates to multiple streams of literature in marketing, statistics, and computer science.

2.1. Dynamic Policy Learning

Our work expands the literature on designing dynamic personalized marketing policies. Early research in this area developed dynamic policies using structural dynamic programming (e.g., Gönül and Shi 1998, Simester et al. 2006, Khan et al. 2009, Hauser et al. 2009) to optimize long-term outcome in contexts such as catalog mailing, customer relationship management, and website design. However, these models typically rely on a simple, discrete state space and basic model specifications, limiting their ability to capture rich heterogeneity and complex relationships between state variables and long-term outcomes. This constraint can lead to suboptimal policies.

More recent literature applies reinforcement learning to handle high-dimensional state spaces and derive optimal policies more efficiently (e.g., Liu 2023, Wang et al. 2023b, Rafieian 2023, Song and Sun 2024). For example, Rafieian (2023) proposes a backward induction approach for policy learning, applying it to ad sequencing to maximize effectiveness, while Wang et al. (2023b) and Liu (2023) use deep reinforcement learning for sequential promotion targeting, demonstrating superior performance over static, myopic policies. Liu (2023) further leverages batch-constrained Q-learning (Fujimoto et al. 2019) to address the challenges of large extrapolation errors when learning dynamic policies from offline data.

While this stream of research highlights the advantages of flexible RL methods for dynamic marketing policies, it often includes many observed customer characteristics as state variables that may introduce irrelevant information about customer responses to marketing efforts. Our results demonstrate that companies can develop more effective RL policies by focusing on state information that is *predictive of multiple customer behaviors*. Using the proposed state representation learning method, we show that companies can achieve better long-term value compared to directly relying on observed customer characteristics.

2.2. Customer Relationship States

Our research contributes to the literature on identifying customer relationship states. In marketing, Hidden Markov Models (HMMs) are widely used for tracking customer relationship states that evolve over time through company interactions (Netzer et al. 2008, Ascarza and Hardie 2013, Ma et al. 2015, Ascarza et al. 2018, Naumzik et al. 2022). This stream of research incorporates customer responses (e.g., customer choice in Netzer et al. (2008) or service usage and retention in Ascarza and Hardie (2013)) to *supervise* state

construction, ensuring that the learned states accurately reflect dynamic customer behavior. While effective, these methods typically yield a limited set of discrete states (often around three), which may not fully capture the heterogeneity of customer states.

In RL literature, several methods have been developed to simplify state space (known as state abstraction) from observed characteristics, each with different goals: approximating the optimal policy (Abel et al. 2016, 2018), maximizing retained information (Anand et al. 2019, Gelada et al. 2019), or preserving the Markov property (Hausknecht and Stone 2015, Zhu et al. 2020, Allen et al. 2021, Song and Sun 2024, Song et al. 2024). Our research builds on the work of Gelada et al. (2019) and Allen et al. (2021), where the former ensures that simplified state representations are predictive of the immediate reward and the latter preserves the Markov property.

We extend this body of work by introducing a supervised *multi-response* state representation learning method that not only preserves the Markov property but also incorporates key customer responses that collectively drive long-term value. Building on previous research that leverages short-term customer responses, such as retention and engagement levels, to better predict sparse spending outcomes (e.g., Huang and Ascarza 2024) and to infer customer relationship states (e.g., Netzer et al. 2008), we demonstrate that including multiple customer responses yields more effective dynamic policy learning compared to using only a single immediate reward to supervise representation learning, as in Allen et al. (2021). Our empirical application shows that multi-response representation learning captures customer heterogeneity and behavior dynamics relevant to decision-making, outperforming existing state representation learning methods.

2.3. Personalized Product Design

This paper contributes to the growing body of research on personalized product design, particularly through the application of machine learning for tailoring product experiences. Prior studies have demonstrated the effectiveness of personalization using various machine learning techniques. For example, Hauser et al. (2009, 2014) use multi-armed bandit algorithms to personalize website design, while Liberali and Ferecatu (2022) combines the bandit approach with a Hidden Markov Model. Yoganarasimhan et al. (2023) optimizes free trial design to increase user subscriptions, Goli et al. (2024) personalizes ad frequency for music streaming users, and Rafieian et al. (2024) adjusts ad formats to boost engagement on a video platform. Although these studies highlight the value of personalization, they typically focus on a single design decision for each customer, without considering the inter-temporal dynamics of design choices.

Our research is especially relevant to the dynamic design of online games, where game mechanics continuously adapt to player behavior. Much of the existing literature examines player progression and behavioral dynamics (e.g., Rutz et al. 2019, Zhao et al. 2022, Castelo-Branco and Manchanda 2023, Wang et al. 2023a) and investigates how design elements such as matching algorithms (Huang et al. 2019), loot boxes (Chen et al. 2021, Amano and Simonov 2024, Miao and Jain 2024), in-game promotions (Lee et al. 2024), and

difficulty adjustments (Lee et al. 2024, Ascarza et al. 2025) influence engagement. Recently, dynamic difficulty adjustment (DDA) has emerged as a key focus area to enhance player engagement. For instance, Ascarza et al. (2025) demonstrates how adapting game difficulty based on past player engagement can improve retention and monetization, while Xue et al. (2017) uses a probabilistic model to classify players into various states (churn, retry, level-up) and leverages dynamic programming to determine optimal difficulty levels for each state. While effective, these methods often simplify player behavior or overlook the heterogeneity among players. Li et al. (2021) advances DDA by predicting immediate churn based on a given difficulty level and selecting difficulty settings to minimize churn, though this approach still lacks consideration of inter-temporal and long-term effects. Importantly, they do not offer a methodological framework to determine the optimal level of difficulty, as we do in this research.

Our research contributes to this literature by introducing an offline RL approach to optimize product design — in our empirical context, game difficulty — over time, capitalizing on the rich heterogeneity in historical customer interactions. This approach enables dynamic adjustments that capture the interplay between current design choices and their future impacts, while effectively learning from high-dimensional customer characteristics through the proposed representation learning method.

3. Problem Formulation

We consider a company that can dynamically adjust specific actions based on observed customer characteristics at each interaction. For example, a gaming company might modify game difficulty based on a player's past engagement, or an e-commerce platform might offer tailored promotions during each browsing session. At the beginning of each period, the company selects an action from a finite set for each customer, based on the observed characteristics up to that point. By the end of the period, the company observes the immediate reward, which serves as a short-term measure of customer value (e.g., spending during the period) and directly contributes to the customer's overall long-term value — the quantity the company aims to maximize. Along with the short-term measure of customer value, the firm also observes a set of (short-term) customer responses — such as click activity, products explored, and other engagement levels — that may influence or provide insights into to the customer's long-term value.

In this section, we formally model the interactions between the company and its customers using a *Markov Decision Process* (MDP) framework and highlight the practical challenges involved in learning an optimal dynamic policy.

3.1. Theoretical Setup

We assume the dynamic customer response process is a MDP (e.g., Liu 2023, Wang et al. 2023b, Rafieian 2023, Bennouna et al. 2024), represented by the tuple $\mathcal{M} = (\mathcal{A}, \mathcal{S}, P^S, p_0^S, \mathcal{X}, P^X, p_0^x, V(\cdot,\cdot), \gamma)$, where each of these elements is described as follows:

- **Action Space:** The action space $\mathcal{A}$ is a predefined set of possible actions $\{a_1, \cdots, a_D\}$ that the company dynamically adjusts.
- **Customer Relationship State:** The state space $\mathcal{S}$ is a continuous vector space representing the customer's relationship with the company, which is *not directly observable* by the company. This space includes an absorbing state, denoted as Churn, indicating the end of customer interactions. We assume that state transitions follow a Markov process, with the state transition function $P_S : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ describing how a customer relationship state evolves in response to the company's actions. Specifically, $P_S(\mathbf{s}' \mid \mathbf{s}, a)$ represents the probability density of the customer relationship state transitioning from $\mathbf{s}$ to $\mathbf{s}'$ after receiving action a. Once the customer reaches the absorbing state of Churn, they do not transition to any other state; i.e., $P_S(\text{Churn} \mid \text{Churn}, a) = 1$ for any $a \in \mathcal{A}$. The initial state distribution $p_0^S : \mathcal{S} \to [0, 1]$ represents the distribution function of customer states at the start of their relationship with the company.
- **Customer Characteristic Space:** The customer characteristic space $\mathcal{X} \subset \mathbb{R}^P$ includes *observable customer characteristics* used in the company's decision-making. We assume that these characteristics indirectly measure the customer's true state while preserving the Markov property.[1] Furthermore, P^X and p_0^x represent the induced transition density and initial distribution of customer characteristics, respectively.
- **Short-term Customer Value:** The company observes the realized short-term value $v_{i,t}$ (i.e., the immediate reward in RL terminology) for customer i at time t. The function $V(\mathbf{s}, a)$ represents the expected short-term value a customer contributes to the company (e.g., spending during the current period) given the state $\mathbf{s}$ and action a, i.e.,$V(\mathbf{s}_{i,t}, a_{i,t}) = \mathbb{E}[v_{i,t}|\mathbf{s}_{i,t}, a_{i,t}]$. The company's goal is to optimize long-term customer value, defined as the sum of (discounted) short-term values over an extended period.
- **Discount factor ($\gamma_{i,t}$):** The discount factor $\gamma_{i,t}$ represents the relative importance the company assigns to short-term values of customer i in the t-th period compared to those realized in the current period. A discount factor near 1 indicates equal value for future and current values, while a factor close to 0 reflects a preference for a high immediate value.

The company can implement a dynamic policy, $\pi : \mathcal{X} \to \Delta(\mathcal{A})$, to take different actions based on observed customer characteristics. Here, $\Delta(\mathcal{A})$ denotes the set of probability distributions over $\mathcal{A}$. In other words, the company selects an action $a \in \mathcal{A}$ with varying probabilities based on the observed customer

[1] Instead of framing this problem as a *partially observable Markov decision process* — which would allow customer characteristics to arbitrarily depend on historical data rather than being Markovian — we assume a Markov process where customer characteristics satisfy the Markov property. This assumption is reasonable because companies can incorporate a comprehensive set of observed characteristics that effectively summarize long-term history, rather than relying solely on behaviors from the previous period. For example, in our empirical setting, we account for playing behaviors from *the beginning of the sample period*, behaviors from *the past week*, and behaviors from the previous day.

characteristics. The ultimate goal is to find an optimal policy $\pi^\star$ for a customer base $\mathcal{C}$ that maximizes the expected long-term value:

$$\pi^\star = \arg\max_{\pi} \ \mathbb{E}_{p_0}\left[\sum_{i\in\mathcal{C}}\sum_{t=1}^{\overline{T}}\gamma_{i,t}\sum_{a\in\mathcal{A}}\pi(a\mid \mathbf{x}_{i,t})V(\mathbf{s}_{i,t},a)\right], \tag{1}$$

where $\mathbf{s}_{i,t}$ is the customer relationship state at the beginning of period t, $\mathbf{x}_{i,t}$ denotes the observed customer characteristics, and $\overline{T}$ is the time horizon for evaluating customer values.

3.2. Empirical Challenges

To construct the optimal policy, the company can use historical data from past customer interactions under their standard business-as-usual practices (known as a behavioral policy, π_b). These data are collected by tracking a sequence of interactions for each customer over time. For a given customer i, the company observes a sequence of interactions denoted as $\mathcal{D}_i = \{(\mathbf{x}_{i,t}, a_{i,t}, v_{i,t}, \mathbf{x}_{i,t+1})\}_{t=1}^{T_i}$, where T_i represents the total number of periods the customer interacted with the company. In each period t, the company records the customer's characteristics at the start of the period ($\mathbf{x}_{i,t}$), such as demographic details, engagement metrics, or purchase history. It also tracks the action applied during that period ($a_{i,t}$), such as a promotion or a specific version of the product, and the short-term value contributed by the customer ($v_{i,t}$). At the end of the period, any updates to the customer's characteristics ($\mathbf{x}_{i,t+1}$) are also recorded. The data collected for all customers is then combined into an offline dataset.

While this offline dataset is essential for understanding customer behavior and designing personalization policies, the company encounters several challenges in using it to identify the optimal policy $\pi^\star$ with existing (offline) reinforcement learning methods.

Challenge 1: Indirect and Noisy Measurement of Customer Relationship State. While companies can collect extensive customer data, not all variables are meaningful or informative for accurately capturing the true customer relationship state that drives long-term value. For example, a manager might assume customers respond differently depending on the day of the week — perhaps expecting higher responsiveness on Mondays and lower responsiveness on weekends. However, if customer responses are largely consistent across days (which the manager is unaware of), including the day of the week as a state variable may introduce unnecessary noise, reducing the efficiency of reinforcement learning and leading to suboptimal policies (Zhang et al. 2021). This adverse effect of noise is particularly pronounced when the dataset $\mathcal{D}$ contains limited observations.

To address this, we argue that companies should not assume the observed characteristics $\mathbf{x}_{i,t}$ are equivalent to the true customer relationship state $\mathbf{s}_{i,t}$ and directly apply reinforcement learning. Instead, they should first construct a *state representation map*, $\mathbf{\Phi} : \mathcal{X} \to \mathcal{S}$, to recover the state that is most relevant for modeling the relationship between the company's actions and long-term customer value.

Challenge 2: Sparsity of Short-term Customer Value and Its Limited Information of Long-term Customer Value. Traditional state construction methods in RL typically condense high-dimensional customer characteristics into a more compact representation while preserving relevant historical information and ensuring the state satisfies the Markov property (Anand et al. 2019, Allen et al. 2021). More recent approaches further emphasize extracting features that predict the short-term value, such as immediate spending (Gelada et al. 2019). However, relying solely on the short-term value for state representation presents two key challenges.

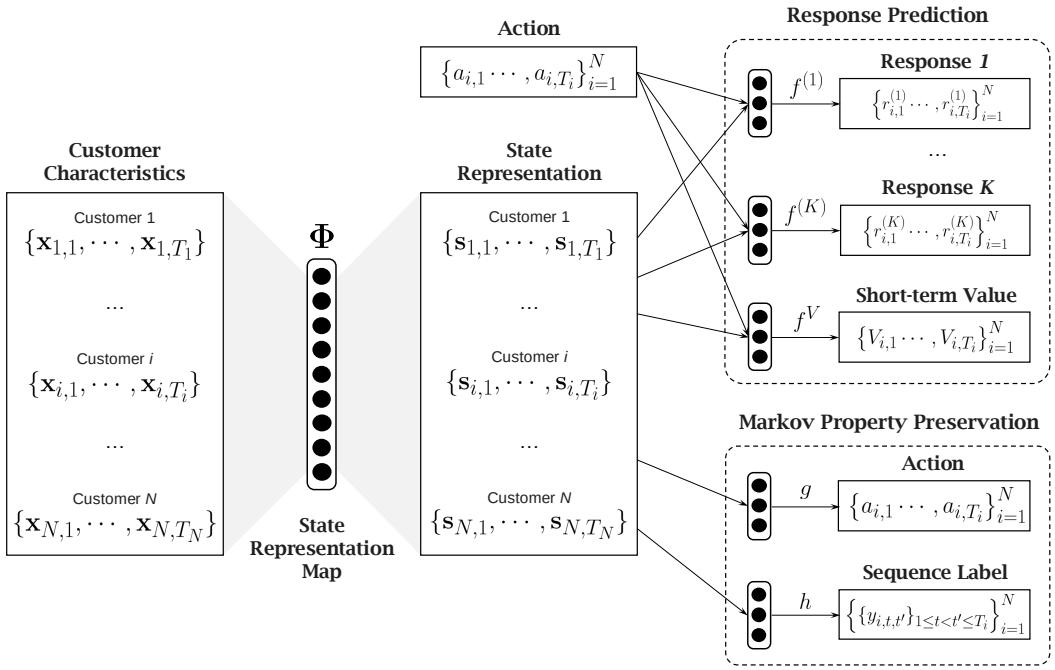
First, short-term values, such as customer spending, are often sparse, with many observations at zero (Zhang et al. 2015, Ascarza et al. 2025). A zero value, though, can signify different long-term outcomes: one customer with zero spending may have lost interest and hold limited future value, while another with the same zero spending might still be highly engaged and have significant potential for future spending. This underscores the need for additional behavioral signals to distinguish between these types of customer, rather than relying solely on the short-term value. Second, short-term value primarily reflects spending behavior during the current period, providing limited insight into future customer value. For example, it does not adequately capture key factors like future retention and engagement, which are critical components of long-term value.

A priori, extracting meaningful information from high-dimensional characteristics — especially when behaviors are sparse and unfold over an extended period — might seem empirically challenging. However, companies often collect additional behavioral signals, such as customer engagements or responses that can serve as valuable proxies for long-term customer value (Yang et al. 2024, Huang and Ascarza 2024). For instance, firms may track when a customer next interacts following a specific action or how much they engage during a given period. We posit that these customer responses, denoted as $\{r_{i,t}^{(1)}, \cdots, r_{i,t}^{(K)}\}$, combined with short-term value $v_{i,t}$, can help firms capture critical information about customer characteristics that is essential for making optimal long-term decisions. In the next section, we discuss how these behavioral signals can be incorporated into state representation learning to more effectively capture this information.

4. Model

To address the challenges discussed above, we propose a novel state construction method called *Multi-Response State Representation* (MRSR) learning. This approach constructs a state representation learning model with two key objectives: (i) capturing information relevant to multiple short-term behaviors that collectively drive long-term customer value, and (ii) preserving the Markov property to ensure the state representation effectively summarizes past interactions and customer relationship dynamics. The resulting state representation is then integrated into a state-of-the-art offline RL method to optimize policy performance.

Figure 1 Model Architecture for Multi-Response State Representation Learning



4.1. Multi-Response State Representation Learning

We first introduce a supervised representation learning method to construct customer relationship states from offline data, ensuring that they (i) are relevant for long-term customer values given different actions, and (ii) preserve the Markov property of the original customer characteristics. To meet these objectives, we set three supervision targets when learning the representation map $\mathbf{\Phi}$: first, the learned state representations should accurately predict short-term customer responses; second, they should be as effective in predicting actions chosen by behavioral policies as the original customer characteristics; third, the state-transition probabilities of the learned states must match those of the original customer characteristics. The first supervised learning target ensures that the learned representations meaningfully capture customer responses, while the second and third targets help preserve the Markov properties of these representations.

Figure 1 presents the proposed model architecture for our state representation learning method. The first model, $\mathbf{\Phi} : \mathcal{X} \to \mathcal{S} \subseteq \mathbb{R}^P$, transforms observed customer characteristics into low-dimensional latent state representations. The second set of models, $\mathbf{f} = \{f^{(1)}, \cdots, f^{(K)}, f^V\} : \mathcal{S} \to \mathbb{R}^{K+1}$, uses these state representations to predict short-term customer responses and the value under various actions. The third set of models, $g : \mathcal{S} \to \mathcal{A}$ and $h : \mathcal{S} \times \mathcal{S} \to \{0, 1\}$, predicts the actions based on the company's behavioral policy and determines whether one state transitioned from another, respectively. Importantly, we jointly train the state representation map $\mathbf{\Phi}$ and all other prediction models to ensure that $\mathbf{\Phi}$ preserves the desired information about short-term customer responses and the Markov property.

4.1.1. Multi-Response Prediction. The first prediction task for supervising representation learning involves predicting both the short-term value $v_{i,t}$ and other short-term responses $r_{i,t}^{(1)}, \cdots, r_{i,t}^{(K)}$, which provide additional information about the customer's long-term value. For example, in a gaming company, these responses could include the time until the next play (a retention proxy) or the number of rounds played (an engagement proxy). By incorporating multi-response predictions, the learned representation captures information relevant to both retention and engagement, which are key behavioral drivers influencing long-term customer value.

The multi-response task is implemented by minimizing the mean squared error (MSE) across all response predictions:

$$\mathcal{L}_{\text{Response}}(\mathbf{\Phi}, \mathbf{f}) = \frac{1}{|\mathcal{D}|} \sum_{i=1}^{N} \sum_{t=1}^{T_i} \left\{ \sum_{k=1}^{K} \delta^{(k)} \left[f^{(k)} \left(\mathbf{\Phi}(\mathbf{x}_{i,t}), a_{i,t} \right) - r_{i,t}^{(k)} \right]^2 + \left[f^{V} \left(\mathbf{\Phi}(\mathbf{x}_{i,t}), a_{i,t} \right) - v_{i,t} \right]^2 \right\}, \quad (2)$$

where $\delta^{(k)}$ are hyper-parameters that determine the relative importance of the prediction accuracy for each short-term responses. Conceptually, one might think that solely supervising the prediction of $v_{i,t}$ (i.e., setting $\delta^{(k)} = 0$ for all K) would be sufficient to retain the relevant information about customer long-term values, given that the long-term value is the cumulative sum of $V(\mathbf{s}, a)$ across each period. However, incorporating other customer responses in the supervision provides several advantages.

First, a major challenge in capturing long-term customer value lies in the high level of sparsity commonly observed in spending data (Zhang et al. 2015, Ascarza et al. 2025). For most customers and time periods, spending is zero, resulting in $v_{i,t} = 0$ for the majority of observations. Yet, zero spending can indicate distinct customer states that require different responses. Consider two customers who both received the same action and did not spend any money afterward. The first customer, characterized by $\mathbf{x}_{i,t}$, has lost interest and exhibits low engagement both now and in the future. Repeating the same action would likely further diminish engagement or even result in churn. In contrast, the second customer, characterized by $\mathbf{x}_{i',t}$, remains highly engaged despite not making a purchase this time. For this customer, applying the same action could still be effective, as they are likely to spend in the future. To distinguish between these cases, short-term engagement levels must be incorporated alongside spending as supervision targets. This approach enables $\mathbf{\Phi}$ to assign distinct state representations to $\mathbf{x}_{i,t}$ and $\mathbf{x}_{i',t}$, accurately capturing their differing relationships with the company. Ignoring engagement and relying solely on short-term spending risks conflating the two scenarios, leading to suboptimal policy decisions.

Second, short-term spending ($v_{i,t}$) reflects only immediate behavior and fails to capture future customer value. For example, while short-term spending reflects the immediate willingness to pay given the current action, it does not predict whether the customer will remain with the company — a key driver of long-term value. To address this, we incorporate customer retention and engagement signals, which capture the actions shape future behavior. Using these proxies as prediction targets allows the model to learn state

representations that more accurately reflect long-term customer responses, enhancing the RL policy's ability to optimize for long-term value.

In our empirical application, we utilize RFM-related metrics (recency, frequency, and monetary value), which are widely regarded as reliable predictors of customer lifetime value (Reinartz and Kumar 2003, Fader et al. 2005). Specifically, recency is measured as the number of days until the customer's next interaction with the company, frequency as the number of interactions within the current period, and monetary value as the amount spent by the customer during that period. These metrics are well-established in the marketing literature and are documented to offer valuable insight into the depth and nature of a customer's relationship with the company (Netzer et al. 2008). In Section 5.4 we show that these metrics are relevant to customer value in our context.

4.1.2. Markov Property Preservation. The second and third prediction tasks ensure that the learned state representation map, $\mathbf{\Phi}$, preserves the Markov property. Standard representation learning models often lack a guarantee against using future information when constructing state representations, as parameters are typically calibrated using the complete trajectory for each customer. Consequently, the learned state representations may not maintain the Markov property of the original observed customer characteristics. To preserve the Markov property, we adopt the approach proposed by Allen et al. (2021), which incorporates two prediction targets to supervise the state representation learning process.

Action Prediction. The first prediction target ensures that the transition dynamics of the learned representation, induced by an action, align with the transition dynamics of the original observed characteristics caused by the same action. This can be framed as a prediction requirement (Allen et al. 2021), where two consecutive learned state representations must accurately predict the action responsible for the transition between two consecutive original customer characteristics. The empirical training objective of this task involves minimizing the following cross-entropy loss:

$$\mathcal{L}_{\text{Inv}}(\mathbf{\Phi}, g) = -\frac{1}{|\mathcal{D}|}\sum_{i=1}^{N}\sum_{t=1}^{T_i}\log g\left(a_{i,t}|\mathbf{\Phi}(\mathbf{x}_{i,t}), \mathbf{\Phi}(\mathbf{x}_{i,t+1})\right) \tag{3}$$

where the prediction model g outputs a distribution over actions, and $|\mathcal{D}|$ denotes the number of $(\mathbf{x}_{i,t}, a_{i,t}, \mathbf{x}_{i,t+1})$ pairs observed in $\mathcal{D}$. When the two consecutive learned representations $\mathbf{\Phi}(\mathbf{x}_{i,t}), \mathbf{\Phi}(\mathbf{x}_{i,t+1})$ can predict the action taken between two consecutive observed customer characteristics $\mathbf{x}_{i,t}, \mathbf{x}_{i,t+1}$ with high accuracy, it is reasonable to conclude that the dynamics of the original characteristics under the behavioral policy are equivalent to the dynamics in the learned representation space.

Sequence Order Prediction. The second prediction target ensures that the relative likelihood of observing two consecutive characteristics in the original data matches the likelihood of their corresponding transformed representations forming consecutive pairs. This can be formulated as the following prediction task (Allen et al. 2021). First, we generate $|\mathcal{D}|$ sequential customer characteristic transition pairs from the

original data, denoted as $\mathcal{D}_i^{\text{seq}} = \{(\mathbf{x}_{i,t}, \mathbf{x}_{i,t+1})\}_{t=1}^{T_i}$, for each customer. Each pair in this sequence is labeled as $y(\mathbf{x}_{i,t}, \mathbf{x}_{i,t+1}) = 1$, indicating it is sequential. Next, we create $|\mathcal{D}|$ non-sequential customer characteristic transition pairs by randomly shuffling the $\mathbf{x}_{i,t+1}$ values within each customer's trajectory, ensuring that the resulting shuffled pairs are not consecutive in the original sequence order. These shuffled pairs are compiled into $\mathcal{D}_i^{\text{non-seq}}$, with each pair $(\mathbf{x}_1, \mathbf{x}_2)$ being labeled as $y(\mathbf{x}_1, \mathbf{x}_2) = 0$. Finally, a prediction model $\widehat{h}$ is constructed to estimate the probability distribution of the sequence labels. This is achieved by minimizing the following loss function:

$$\mathcal{L}_{\text{Ratio}}(\mathbf{\Phi}, h) = -\frac{1}{2|\mathcal{D}|} \sum_{i=1}^{N} \sum_{(\mathbf{x}_1, \mathbf{x}_2) \in \mathcal{D}_i^{\text{ratio}}} \log h\left(y_{\mathbf{x}_1, \mathbf{x}_2} \mid \mathbf{\Phi}(\mathbf{x}_1), \mathbf{\Phi}(\mathbf{x}_2)\right), \tag{4}$$

where $\mathcal{D}_i^{\text{ratio}} = \mathcal{D}_i^{\text{seq}} \bigcup \mathcal{D}_i^{\text{non-seq}}$ for each customer i, $y_{\mathbf{x}_1, \mathbf{x}_2} \in \{0, 1\}$ is a label that takes the value 1 when $(\mathbf{x}_1, \mathbf{x}_2)$ is from $\mathcal{D}_i^{\text{seq}}$ and 0 when it is from $\mathcal{D}_i^{\text{non-seq}}$. Following similar logic as in the action prediction task, if the learned representations can distinguish between sequential and non-sequential pairs with accuracy comparable to that constructed using observed customer characteristics, the company can reasonably conclude that the density ratios of the original and the represented next states are approximately equal. In that case, the density ratio condition would be satisfied.

Putting it all together, we construct the state representation map $\hat{\mathbf{\Phi}}$ by jointly minimizing the loss functions from three distinct prediction tasks:

$$\underset{\mathbf{\Phi}, \mathbf{f}, g, h}{\arg\min} \;\; \mathcal{L}_{\text{Response}}(\mathbf{\Phi}, \mathbf{f}) + \alpha \mathcal{L}_{\text{Inv}}(\mathbf{\Phi}, g) + \beta \mathcal{L}_{\text{Ratio}}(\mathbf{\Phi}, h), \tag{5}$$

where α and β determine the relative importance of the two Markov preservation tasks. This approach addresses the challenges of noisy and irrelevant state information, ensures the representation retains key predictive signals for long-term value, and preserves the Markov property essential for effective reinforcement learning.

4.1.3. Practical Consideration. The multi-response state representation effectively captures long-term customer value by embedding critical information for policy learning, achieved through predictions of multiple customer responses and short-term value. Its success, however, depends on the quality of the additional responses, with two key factors determining whether the state representation is policy-relevant.

First, including customer responses that are not truly relevant to long-term value can make the learned state representation less informative than one based solely on short-term customer value, potentially leading to a suboptimal or even less profitable policy — commonly known as the negative transfer problem (Wang et al. 2019, Wu et al. 2020). In the context of optimizing long-term customer value, retention and engagement proxies have proven valuable for capturing customer relationship dynamics over time (e.g., Ascarza and Hardie 2013, Yang et al. 2024, Huang and Ascarza 2024). Our empirical results show that "time to next

play" (a retention proxy) and "rounds played in a day" (an engagement proxy) can help companies enhance long-term value. However, depending on their objectives, managers should choose short-term responses that (i) are predictive of long-term value and (ii) provide more information about the long-term value than merely reflecting short-term customer value.

Second, incorporating numerous short-term responses as supervised targets in state representation learning can make it challenging to balance accurate response predictions with preserving the Markov property. To achieve this balance, the weight of each objective should be adjusted. Companies can tune hyperparameters and evaluate short-term response prediction errors alongside Markov losses. If the Markov losses are not significantly higher than those based on observed customer characteristics, the Markov property is considered preserved, thereby allowing companies to prioritize improvement in prediction accuracy for various short-term responses.

4.2. Deep Conservative Q-Learning

Once the state representation map $\hat{\mathbf{\Phi}}$ is constructed, we use the derived state representations instead of the original customer characteristics for reinforcement learning. To ensure reliable policy learning and evaluation with offline data collected under the company's original policy π_b, we adopt *conservative Q-learning* (CQL) (Kumar et al. 2020) instead of the traditional Q-learning.

CQL consists of two components for learning. The first is the standard Q-learning process, which iteratively estimates the long-term value of taking specific actions at the current time (referred to as the action-value function or Q-value) while adhering to the optimal policy for future actions. The second component penalizes the learned action-value function when it deviates too far from the one induced by the behavioral policy π_b. This ensures that the learned policy remains conservative, mitigating overestimation error for state-action pairs that are underrepresented in the offline data, thereby improving stability and robustness in offline reinforcement learning.

4.2.1. Optimal Action-Value Function Approximation. In traditional Q-learning, the primary goal is to estimate an action-value function, q_π, which is then used to identify the optimal action. For a given policy π, this function represents the expected discounted long-term value of taking action a in state $\mathbf{s}$ at the current period t, while adhering to policy π for all future decisions. Formally, it is defined as:

$$q_\pi(\mathbf{s}, a) = \mathbb{E}_\pi \left[\sum_{j=0}^{\infty} \gamma_{t+j} V(\mathbf{s}_{i,t+j}, a_{t+j}) \middle| \mathbf{s}_{i,t} = \mathbf{s}, a_{i,t} = a \right].$$

Then, the action-value function of the optimal policy, $\pi^\star$, satisfies the *Bellman optimality equation* (Andrew and Richard 2018):

$$q_{\pi^\star}(\mathbf{s}, a) = V(\mathbf{s}, a) + \mathbb{E} \left[\gamma_{t+1} \max_{a'} q_{\pi^\star}(\mathbf{s}_{i,t+1}, a') \middle| \mathbf{s}_{i,t} = \mathbf{s}, a_{i,t} = a \right]. \tag{6}$$

This equation forms the basis for learning the action-value function of the optimal policy through an iterative approximation procedure known as *Q-learning*. During each iteration τ, the Q-learning algorithm minimizes the *Bellman error*, defined as:

$$\mathcal{L}_{\text{Bellman}}(\hat{q}^{(\tau)}, \hat{q}^{(\tau-1)}) = \frac{1}{|\mathcal{D}|} \sum_{i=1}^{N} \sum_{t=1}^{T_i} \left[v_{i,t} + \gamma_t \max_{a'} \hat{q}^{(\tau-1)} \left(\hat{\mathbf{\Phi}}(\mathbf{x}_{i,t+1}), a' \right) - \hat{q}^{(\tau)} \left(\hat{\mathbf{\Phi}}(\mathbf{x}_{i,t}), a_{i,t} \right) \right]^2. \quad (7)$$

After $\mathcal{T}$ iterations, when the Bellman error converges, we can approximate the optimal action-value function as closely as possible. In deep Q-learning (Mnih et al. 2015), a deep neural network is used as the functional form of $\hat{q}$, providing greater flexibility for action-value approximation.

4.2.2. Conservative Penalization. Prior research highlights the challenges of learning from offline data, particularly in reinfocement learning, where state-action pairs are often underrepresented in the data (Fujimoto et al. 2018, 2019, Levine et al. 2020). Specifically, policies that deviate significantly from the behavioral policy π_b are prone to extrapolation errors, a common issue in deep Q-learning. This challenge is especially relevant in real-world marketing settings, where historical customer interaction data is typically collected under simple business rule-based policies. Such policies often limit the exploration of alternative actions, resulting in incomplete coverage of the potential state-action space in the dataset.

To overcome the extrapolation issue, we utilize *Conservative Q-learning* (CQL) (Kumar et al. 2020) to ensure safe policy learning. Specifically, CQL not only minimizes the Bellman error but also includes an explicit term designed to penalize (i) deviations of the learned action-value function from the action-value function estimated under the behavioral policy, and (ii) deviations of the learned policy from the behavioral policy itself. To achieve this, Kumar et al. (2020) suggests minimizing the following loss function:

$$\mathcal{L}_{\text{Dev}}(\hat{q}^{(\tau)}, \pi_b) = \log \sum_{a \in \mathcal{A}} \exp \left(\hat{q}^{(\tau)} \left(\hat{\mathbf{\Phi}}(\mathbf{x}_{i,t}), a_{i,t} \right) \right) - \sum_{a \in \mathcal{A}} \pi_b \left(a | \mathbf{x}_{i,t} \right) \hat{q}^{(\tau)} \left(\hat{\mathbf{\Phi}}(\mathbf{x}_{i,t}), a_{i,t} \right), \quad (8)$$

where $\pi_b \left(a | \mathbf{x}_{i,t} \right)$ is the probability of assigning action a given the characteristics $\mathbf{x}_{i,t}$ under the behavioral policy.

Together, deep conservative Q-learning minimizes the following combined loss function in each iteration:

$$\hat{q}^{(\tau)} = \arg\min_{q} \ \ \mathcal{L}_{\text{Bellman}}(q, \hat{q}^{(\tau-1)}) + \eta \mathcal{L}_{\text{Dev}}(q, \pi_b), \quad (9)$$

where η is the hyper-parameter that controls the degree of deviation from the behavioral policy. This approach ensures that the expected value of a Softmax policy under the learned Q-function serves as a lower bound to its true value, offering a conservative estimate of the policy's performance (Kumar et al. 2020). This conservatism helps mitigate the risk of overestimating the policy's value due to extrapolating errors for rarely observed characteristic-action pairs. When the combined CQL loss shows no further improvement after sufficient iterations, the estimated conservative Q-function $\hat{q}$ can be derived.

4.2.3. Policy Deployment Based on Conservative Q-function. Finally, once the conservative Q-function $\hat{q}$ is estimated, we can determine the action assignment based on the Q-value corresponding to specific $\mathbf{x}$. In our research, we implement the following constrained Softmax policy:

$$\hat{\pi}(a|\mathbf{x}) = \frac{\mathcal{I}\left(\pi_b(a|\mathbf{x}) > 0\right) \exp\left(\hat{q}(\hat{\mathbf{\Phi}}(\mathbf{x}), a)\right)}{\sum_{a' \in \mathcal{A}} \mathcal{I}\left(\pi_b(a|\mathbf{x}) > 0\right) \exp\left(\hat{q}(\hat{\mathbf{\Phi}}(\mathbf{x}), a')\right)}, \tag{10}$$

where $\mathcal{I}(\cdot)$ is the indicator function. We propose this policy for two main reasons. First, the Softmax component satisfies the exponential proportion condition outlined in Kumar et al. (2020), ensuring that CQL achieves the desired conservative policy value and mitigates large extrapolation errors.[2] Second, we restrict the policy from making decisions outside the region covered by the offline collected data. This ensures valid offline policy evaluation and prevents reliance on extrapolated Q-values for decision-making.

4.3. Summary

Algorithm 1 summarizes the proposed multi-response state representations (MRSR) method with CQL.

Algorithm 1 Multi-Response State Representation with Conservative Q-Learning

Input: $\delta^{(1)}, \cdots, \delta^{(K)}$, α, β, η, $\mathcal{T}$: Number of iterations for CQL, $\hat{\pi}_b(a|\mathbf{x})$: (Estimated) behavioral policy

Data: Offline Interaction Data $\mathcal{D} = \bigcup_{i=1}^{N} \{(\mathbf{x}_{i,t}, a_{i,t}, \mathbf{r}_{i,t}, \mathbf{x}_{i,t+1})\}_{t=1}^{T_i}$

Output: State Representation Map $\hat{\mathbf{\Phi}}$, Action-Value Function $\hat{q}$, Constrained Softmax Policy $\hat{\pi}(a|\mathbf{x})$

Step 1: Multi-Response State Representation Learning.

Initialize the encoding network ($\hat{\mathbf{\Phi}}$), short-term response prediction networks ($\hat{f}^{(1)}, \cdots, \hat{f}^{(k)}, \hat{f}^{V}$), action prediction network ($\hat{g}$), and sequence order prediction network ($\hat{h}$).

Update $\hat{\mathbf{\Phi}}, \hat{\mathbf{f}}, \hat{g}, \hat{f}$ by performing mini-batch gradient descent on the joint loss function for representation learning: $\mathcal{L}_{\text{Response}}(\hat{\mathbf{\Phi}}, \hat{\mathbf{f}}) + \alpha \mathcal{L}_{\text{Inv}}(\hat{\mathbf{\Phi}}, \hat{g}) + \beta \mathcal{L}_{\text{Ratio}}(\hat{\mathbf{\Phi}}, \hat{h})$.

Step 2: Conservative Q-learning.

Initialize conservative Q-network, $\hat{q}^{(0)}$.

for $\tau = 1, \cdots, \mathcal{T}$ **do**

  Set $\hat{q}^{(\tau)} \leftarrow \hat{q}^{(\tau-1)}$.

  If $\mathbf{x}_{i,t}$ corresponds to a churn state, set $\hat{q}^{(\tau-1)}\left(\hat{\mathbf{\Phi}}(\mathbf{x}_{i,t}), a\right) = 0$ for all $a \in \mathcal{A}$.

  Update $\hat{q}^{(\tau)}$ by performing mini-batch gradient descent on $\mathcal{L}_{\text{Bellman}}(\hat{q}^{(\tau)}, \hat{q}^{(\tau-1)}) + \eta \mathcal{L}_{\text{Dev}}(\hat{q}^{(\tau)}, \hat{\pi}_b)$.

end

Set $\hat{q} = \hat{q}^{(\tau)}$.

Step 3: Policy Deployment.

Deploy the constrained Softmax policy: $\hat{\pi}(a|\mathbf{x}) = \frac{\mathcal{I}(\hat{\pi}_b(a|\mathbf{x})>0) \exp\left(\hat{q}(\hat{\mathbf{\Phi}}(\mathbf{x}),a)\right)}{\sum_{a' \in \mathcal{A}} \mathcal{I}(\hat{\pi}_b(a|\mathbf{x})>0) \exp\left(\hat{q}(\hat{\mathbf{\Phi}}(\mathbf{x}),a')\right)}$.

[2] For additional theoretical details, refer to the Theorem 3.3 in Kumar et al. (2020).

First, we learn a lower-dimensional state representation map by minimizing the loss function in Equation (5). Second, we train the deep conservative Q-network by minimizing the loss function in Equation (9) using the learned representations. To improve learning stability and accurately reflect long-term customer value, we set the Q-network value to zero for all actions when the observation corresponds to a churn state.

5. Empirical Context and Data

5.1. Business Context

We apply our proposed methodology to a free-to-play (F2P) mobile puzzle game to develop a dynamic rule for adjusting game difficulty based on observed player characteristics. The game is similar to popular titles like Candy Crush Saga, where players swap adjacent blocks on a grid to align three or more of the same type in a row or column to earn points. Players progress through levels with specific objectives, such as achieving a target score, clearing all blocks, or collecting specific block types. Progression in the game is sequential, requiring players to complete each level to unlock the next, with objectives becoming progressively more challenging. After each round, players receive feedback indicating whether they met the level's objectives (win or lose) and their total points earned. This setup provides an ideal environment for testing our approach to dynamic difficulty adjustment, as it combines player performance data with sequential decision-making.

Like many F2P games, this game includes an in-game currency which players can spend to help them advance in the game. Players start with a set number of coins and have the option to purchase additional coins using real money. The game employs various strategies to manage player progression and encourage spending. Each player begins with five lives and loses one each time they fail to meet the objective of a play round. Lives regenerate at a rate of one every 30 minutes, up to a maximum of five, but players can use coins to purchase additional lives and skip the waiting period. The game also includes a "gate" mechanism that imposes a mandatory five-day waiting period every 20 levels. During this time, players can only replay previously completed levels unless they use coins to bypass the gate. Additionally, players can spend coins on "power-ups" that help them complete objectives more efficiently and earn higher scores. These design elements are carefully crafted to incentivize coin purchases, making coin spending the primary metric for the company to evaluate player value and optimize their monetization strategies.

5.2. Dynamic Difficulty Adjustment Experiment

An important factor influencing player activity — and ultimately their long-term value — is the *game difficulty* set for each round (e.g., Xue et al. 2017). If the game is too difficult and players fail to progress after several attempts, there is a risk of churn, leading to a loss of long-term value for the company. Conversely, if the game is too easy, players might continue playing but feel less inclined to purchase extras with coins or might churn due to a lack of challenge, negatively impacting both retention and monetization. To address these trade-offs, rather than relying on a static, pre-defined difficulty curve as players progress through levels, the company implemented a *dynamic difficulty adjustment* (DDA) policy to modify game difficulty on

a daily basis. Specifically, this policy adjusted the probability of color co-occurrence between consecutive blocks on the game board — higher probabilities made it easier for players to achieve the level's objective. This dynamic approach allows the company to tailor difficulty levels in real time, balancing player engagement and monetization Ascarza et al. (2025).

The company conducted an experiment to test two different DDA policies, both designed to lower game difficulty when players exhibited reduced engagement, measured by the number of rounds played during the week prior. The new difficulty levels were defined as $\mathcal{A} = \{1, 2, 3, 4, 5\}$, where level 1 is the easiest, and level 5, corresponding to the original difficulty of each round before any adjustments, is the most challenging. Difficulty adjustments were made daily: every day at midnight (00:00 UTC), the company evaluated each player's performance over the previous week and, based on the given mapping rule (Table 1), assigned a difficulty level for all rounds to be played that day. These adjustments were only applied if the player logged in and started the game.[3]

Table 1 shows the number of rounds played and the corresponding difficulty level mappings for the two DDA policies tested in the experiment. The company conducted a player-level randomized experiment in three phases. In the first wave, 30% of players were randomly selected and assigned Setting 1 from Table 1. The second wave began one month later, targeting another random sample of 30% of players and applying Setting 2. The third wave, launched one month after the second wave, sampled an additional 30% of players and again applied Setting 1. The remaining 10% of eligible players did not receive any difficulty adjustments and consistently experienced gameplay at difficulty level 5. For each wave, the designated mapping rule was applied only after a player became eligible for difficulty adjustment according to the mapping rule. Once assigned, the assigned DDA policy remained fixed for that player and was applied in all subsequent gameplay throughout the experimental period.

Table 1 Difficulty Mapping Rules

Setting 1		Setting 2	
# Rounds Played in Last 7 Days	**Difficulty Level**	**# Rounds Played in Last 7 Days**	**Difficulty Level**
< 5	1	< 10	1
≥ 5 and < 10	2	≥ 10 and < 17	2
≥ 10 and < 15	3	≥ 17 and < 28	3
≥ 15 and < 20	4	≥ 28 and < 40	4
≥ 20	5	≥ 40	5

The inclusion of different policies in the experimental design introduces exogenous variations in difficulty assignments. These variations enable us to quantify the impact of different difficulty levels on players' coin spending behavior and to optimize the DDA policy to enhance players' long-term value.

[3] The experiment was conducted only on players who had previously passed level 20.

5.3. Data

We observe each player's gameplay history, including every round they played. Using these data, we construct a dataset of active-day-player gameplay transitions (966,910 observations) spanning from the start of the first experimental wave to one month after the third wave's launch. This dataset represents a random sample of 100,000 players who met the experiment qualifications. For each player i, we include data from their first active day after meeting the experiment criteria up to their last activity within the dataset's timeframe.[4] Each observation, $(\mathbf{x}_{i,t}, a_{i,t}, v_{i,t}, \mathbf{r}_{i,t}, \mathbf{x}_{i,t+1})$, captures a transition between two consecutive active days. If a player did not play on a given day, no observation was recorded for that day. Thus, the subscript t in our setting refers to one *active day* for player i, with $t+1$ denoting the next active day.

Figure 2 illustrates the process of constructing each observation during the company's DDA experiments. Each observation includes the following information:

1. **Player Characteristics** ($\mathbf{x}_{i,t}$ and $\mathbf{x}_{i,t+1}$): We collect 32 player characteristics for each active play day, including outcome metrics such as points per round, number of active rounds, and coin spending, as well as historical metrics such as the number of days since the last play day and coins used in the previous day or week. These metrics are used to construct the state representation. We also create an indicator variable to identify whether a player has churned (i.e., no observations during the sampling period following that day), marking an absorbing state in the player relationship state space.
2. **Game Difficulty**: There are five difficulty levels, $a_{i,t} \in \{1, 2, 3, 4, 5\}$.
3. **Short-term Player Value**: We define the short-term player value $v_{i,t}$ as the total coins spent by player i on day t, which is the primary outcome of interest for the focal company.
4. **Additional Player Responses**: In addition to the short-term value, we also collect two behavioral outcomes for our representation learning: the number of days until the next active day, $r_{i,t}^{\text{Days}}$ (short-term recency, serving as a proxy for player retention), and the number of rounds played on that day, $r_{i,t}^{\text{Rounds}}$ (short-term frequency, serving as a proxy for player engagement).

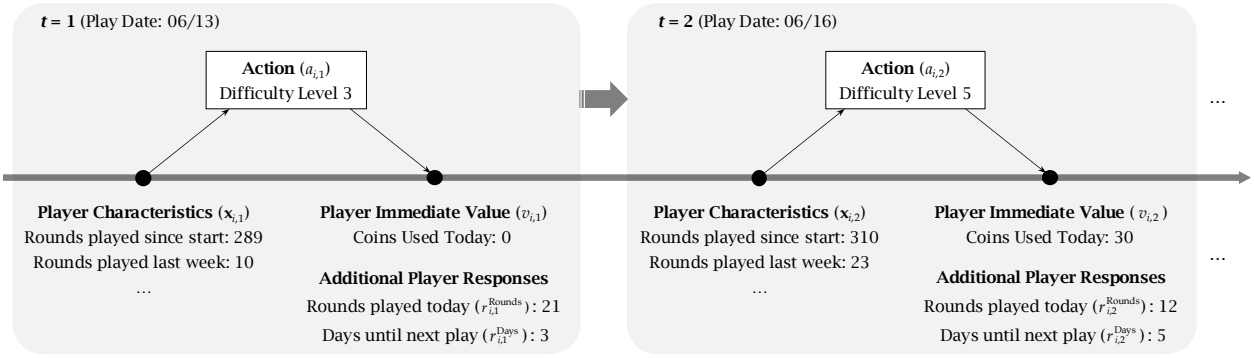
Summary statistics for all these variables can be found in Online Appendix A.

5.4. Descriptive Evidence

Before applying the proposed framework to the empirical setting, we explore the data by analyzing the impact of the DDA experiment on subsequent player behavior and presenting empirical evidence that short-term recency (r^{Days}) and engagement metrics (r^{Rounds}) provide valuable information for long-term value, specifically cumulative long-term coin spending.

[4] The 30-day horizon aligns with the company's standard measurement practices for measuring long-term behavior.

Figure 2 Data Construction Procedure



Note. The figure illustrates how detailed gameplay data is transformed into inputs for the proposed model. In this example, the player belongs to the treatment group in Wave 1. Starting on June 11 (the beginning of the experiment), the player played the game on June 13, 16, and 21, with potential additional playdays thereafter. Before $t = 1$ and $t = 2$, player characteristics were calculated based on past gameplay behaviors. On the first playday, the player was assigned a game difficulty level of 3, having played only 10 rounds in the previous week. By the second playday, the assigned difficulty level increased to 5, as rounds played in the prior week rose to 23. At the end of each playday, the company recorded the coins used and rounds played on that day. When the next playday occurred, the company additionally recorded the days until the next play. If the player stopped playing altogether, $r_{i,t}^{\text{Days}}$ was set to 31 days.

5.4.1. Impact of DDA on Player Behaviors. We begin by examining the impact of the company's DDA experiment on three key dimensions of player behavior over the 30 days following their qualification for difficulty adjustment: engagement intensity (average number of rounds played per active player on day t), play consistency (average of the cumulative days played per player up to day t), and monetization (average of the cumulative coins spent per player up to day t). Since each wave tests a distinct DDA setting, we present the results separately for each wave. For each experimental wave, we estimate the average treatment effect on players who qualified for difficulty adjustment by comparing treated players to control players. In Setting 1,[5], we analyze players who played fewer than 20 rounds in the past seven days, as this threshold triggered the DDA policy. Similarly, in Setting 2, we compare treated and control players who played fewer than 40 rounds during the same period.

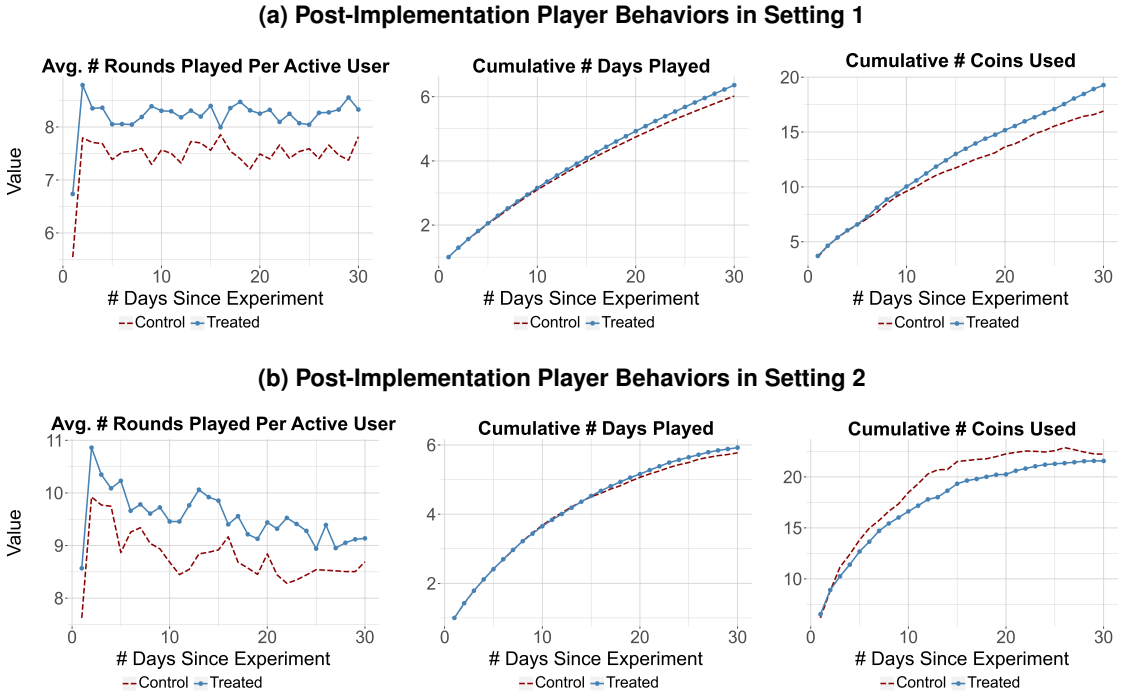
Figure 3 shows the average values of the three key metrics across treatment and control conditions. The first row shows the results for Setting 1, while the second row presents the results for Setting 2. The findings suggest that both DDA settings increase player engagement intensity over the 30-day period following eligibility for difficulty adjustment. This indicates that adjusting difficulty effectively maintains players' interest and deepens their involvement in the game. As shown in the middle column, treated players in both settings also exhibit an increase in the number of days they played. This improvement in play consistency suggests that difficulty adjustments not only enhance engagement but also encourage sustained gameplay, potentially improving player retention over time.

[5] For Setting 1, we focus on the first wave of the experiment as the complete 30-day behaviors of some players in wave 3 are not observed.

The right-most column of Figure 3 illustrates the contrasting effects of difficulty reduction on players' 30-day coin spending between the first and second settings. In the first setting, where difficulty was reduced exclusively for players with low engagement (fewer than 20 rounds in the previous week), the DDA adjustment positively impacted 30-day coin spending. However, in the second setting, where the reduction was extended to moderately engaged players (between 20 to 40 rounds in the 7 days prior) and was implemented more aggressively, the impact became negative, as treated players used fewer coins than the control players.

Together, these results highlight the inherent trade-offs associated with difficulty reduction. On the one hand, lowering game difficulty helps retain players by encouraging sustained gameplay, thereby increasing opportunities for future coin spending. On the other hand, reducing the challenge for engaged players may reduce their incentive to spend coins on extras and negatively impact long-term monetization. This complexity in response highlights the importance of implementing a more nuanced dynamic difficulty adjustment strategy that considers both variations in player characteristics and engagement levels.

Figure 3 Post-Implementation Player Behaviors Across Two Difficulty Adjustment Settings



Note. Panel (a) shows post-implementation player behavior in the experiment for Setting 1, while Panel (b) shows post-implementation player behavior for Setting 2. Both panels compare the averages between treated and control players. The left column shows the average number of rounds played by active players, with each point representing the average number of rounds played on day t (x-axis) by players who were active on that day. The middle column summarizes the cumulative days played by players up to day t, and the right column presents the cumulative coins used by players up to day t.

5.4.2. Relationship Between Player Short-Term Responses and Long-Term Value. To validate the use of RFM-related metrics — recency (days until next play), engagement intensity (number

of rounds played), and monetization (coins spent) — as short-term responses for capturing players' long-term value in our representation learning method, we estimate a regression model linking these metrics to long-term value (30-day coin spend). This analysis is conducted exclusively on control group observations to avoid confounding effects from difficulty adjustments, ensuring that the results reflect intrinsic player behavior. Specifically, we estimate the following fixed-effects regression model:

$$\log(\text{30-Day Coins}_{i,t}) = \alpha + \beta_v \log(v_{i,t}) + \beta_{\text{Days}} \log(r_{i,t}^{\text{Days}}) + \beta_{\text{Rounds}} \log(r_{i,t}^{\text{Rounds}}) + \mathbf{x}_{i,t}^{\top}\boldsymbol{\eta} + \zeta_i + \psi_t + \varepsilon_{i,t},$$

where $\log(\text{30-Day Coins}_{i,t})$ is the logarithm of the total coins spent by player i over the subsequent 30 days (beginning on day t), $\log(v_{i,t})$ is the logarithm of the short-term value (coin spending) of player i on day t, $\log(r_{i,t}^{\text{Days}})$ is the logarithm of the short term recency (number of days until next play), $\log(r_{i,t}^{\text{Rounds}})$ is the logarithm of the short term frequency (number of rounds played by player i on day t), and $\mathbf{x}_{i,t}$ includes the player characteristics described in Section 5.3 used as control variables. We also include player fixed effects (ζ_i) and date fixed effects (ψ_t) in the regression specification to account for unobserved heterogeneity and time-specific factors.[6] The coefficients β_v, β_{Days}, and β_{Rounds} capture the marginal relationships between the short-term signals and long-term value, while holding the other variables constant.

Table 2 presents the results of three regression models with different fixed effect specifications. As expected, players' daily coin spending is significantly and positively associated with their 30-day total coin spending. The analysis also shows a significant negative relationship between the time until a player's next session and 30-day spending, along with a significant positive relationship between short-term engagement intensity and long-term value. Notably, variations in recency and engagement intensity are associated with statistically significant differences in future spending, even after controlling for immediate coin spending. These findings provide empirical support for incorporating not only immediate spending but also retention and engagement metrics to better capture and predict long-term player value.

6. Empirical Performance

We now evaluate the effectiveness of our proposed method using the gaming data described in Section 5. First, we outline the model implementation process, followed by a description of the evaluation approach and the models used for comparison. Finally, we demonstrate how our proposed solution outperforms existing benchmarks in maximizing long-term value.

6.1. Model Implementation

We begin by summarizing the implementation details of the proposed model, which comprises six distinct network sets: the encoding network $\mathbf{\Phi}$, which transforms observed player characteristics into state representations; three short-term response prediction networks, f^{Days}, f^{Rounds}, and f^{Coins}; the action prediction

[6] We add 1 to both 30-Day Coin$_{i,t}$ and $v_{i,t}$ to handle zero values in the logarithmic calculations.

Table 2 Relationship Between Short-term Responses and 30-day Coin Spending

	log(30-Day Coins)		
	(1)	(2)	(3)
log(# Coins Used on the Day)	$0.6744^{***}(0.0184)$	$0.3117^{***}(0.0159)$	$0.3115^{***}(0.0159)$
log(# Days Until Next Play)	$-0.1731^{***}(0.0059)$	$-0.0450^{***}(0.0048)$	$-0.0442^{***}(0.0048)$
log(# Rounds Played on the Day)	$0.0610^{***}(0.0082)$	$0.0152^{***}(0.0055)$	$0.0149^{***}(0.0055)$
Control Variables	Yes	Yes	Yes
Player Fixed Effects	No	Yes	Yes
Date Fixed Effects	No	No	Yes
# Observations	80,442	80,442	80,442
R^2	0.3207	0.7912	0.7915
Adjusted R^2	0.3204	0.7594	0.7595

Note. $^{*}p < 0.1$, $^{**}p < 0.05$, $^{***}p < 0.01$. Standard errors are clustered at the player level.

network g; the sequence order prediction network h; and the conservative Q-network q. Each of these networks is structured as a five-layer fully-connected neural network with 16 hidden units per layer.[7] All hidden units use the Rectified Linear Unit (ReLU) activation function ($\sigma(x) = \max(0, x)$). Further details for each network set are provided below:

- **Encoding Network**: It takes 32 player characteristics as input and outputs a 15-dimensional vector as the state representation.[8]
- **Short-Term Response Prediction Networks**: Each short-term response prediction network uses a 15-dimensional learned representation as input. The output layer applies a linear activation function to predict the corresponding short-term response.
- **Action Prediction Network and Sequence Order Prediction Network**: Both networks generate predicted probabilities using the Softmax activation function in the output layer. The action prediction network generates five probabilities corresponding to the five difficulty levels, while the sequence order prediction network predicts the probability that two input player characteristics represent the same transition observation.
- **Conservative Q-Network**: The conservative Q-network takes as input the concatenation of the 15-dimensional state representation and the churn indicator, producing five action values through its output layer with the Softmax activation function.

To train the network on the training set, we begin by learning the player feature representations by optimizing the loss function described in Equation (5). Next, we generate state representations for each observation in the training set and train the conservative Q-network by minimizing the loss function specified in Equation (9). We use the adaptive moment estimation (Adam) algorithm (Kingma 2014), which combines momentum with adaptive learning rates to enable faster convergence. Similar to Liu (2023), we

[7] Alternative neural network structures are tested in Online Appendix D.1.

[8] Alternative state representation dimensions are explored in Online Appendix D.2.

employ mini-batch training to enhance efficiency and stability. We apply Z-score normalization to the short-term responses to balance objective scales and accelerate convergence. Further implementation details are available in Online Appendix B.2.

6.2. Benchmark Policies for Comparison

We evaluate the performance of the proposed MRSR model against the following benchmark polices:

- **Behavioral policy** (`Behavioral`): We estimate the difficulty adjustment rule based on offline data collected by the company. Further details are provided in Online Appendix B.1.
- **CQL with observed player characteristics** (`Standard CQL`): We construct a conservative Q-network that utilizes the observed player characteristics directly and then apply the softmax policy. This benchmark allows us to quantify the added value of state representation learning.
- **CQL with single-response state representation** (`SRSR`): Instead of learning a Markov state representation that captures short-term recency, engagement, and coin spending, this method constructs the state representation using only the immediate coin spending, $v_{i,t}$. The single-response representation is then used as input for conservative Q-learning. This benchmark enables us to quantify the value of incorporating multiple player behaviors when supervising state representation learning.
- **Myopic policy** (`Myopic`): We construct a short-term policy by setting the discount factor to zero when performing conservative Q-learning with the original player characteristics. This approach focuses solely on short-term coin spending and serves to assess the benefit of incorporating long-term value considerations in reinforcement learning.

For all the benchmarks above, the neural network structures are comparable to the corresponding components of the proposed method. Additional details on model specifications are provided in Online Appendix B.2. We also perform validity checks in Online Appendix C to ensure that both the multi-response and single-response representations preserve the Markov property from the original player characteristics and demonstrate the prediction accuracy of short-term coin spending across methods.

6.3. Empirical Performance

6.3.1. Off-Policy Evaluation Method. We evaluate the performance of various policies using the stepwise importance sampling estimator (Andrew and Richard 2018). Specifically, the stepwise importance sampling estimator for the long-term value of a target policy π on a dataset collected under a behavioral policy π_b is:

$$\widehat{\mathrm{LV}}_{\pi} = \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T_i}\left(\prod_{j=1}^{t}\gamma_0^{d_{i,j}}\right)\left[\prod_{j=1}^{t}\frac{\pi\left(a_{i,j}|\mathbf{x}_{i,j}\right)}{\pi_b\left(a_{i,j}|\mathbf{x}_{i,j}\right)}\right]V_{i,t}. \tag{11}$$

where γ_0 is the baseline discount factor and $d_{i,j}$ represents the number of days between the $j-1$-th and the j-th observations (i.e., number of days between two active days in the company's context) when $j > 1$

and 0 otherwise. In our empirical analysis, we estimate the action distribution $\hat{\pi}_b(a|\mathbf{x})$ as a function of the player characteristics $\mathbf{x}$ using logistic regression with 5-fold cross-fitting to approximate the behavioral policy empirically.[9]

The step-wise importance sampling estimator provides an unbiased estimate of the counterfactual policy value of π if (i) there are no hidden confounders influencing the choice of actions beyond the observed player characteristics, and (ii) the probability of observing an action determined by the target policy in the offline dataset is non-zero, i.e., $\pi_b(a|\mathbf{x}) > 0$ whenever $\pi(a|\mathbf{x}) > 0$ (Hanna et al. 2021). Both these conditions are satisfied in our context. First, the unconfoundedness assumption holds as the company's adjustments on any given day were solely based on the number of rounds played in the preceding week, which is included in the observed player characteristics. Second, since our policy is constrained to operate within the action space covered by the offline dataset, as defined in Equation (10), the non-zero probability condition is inherently satisfied. Together, these factors validate the use of the company's collected data for policy evaluation.

6.3.2. Policy Comparison Results. We now evaluate the long-term player values (i.e., the 30-day cumulative coin spending) for each policy learning method. Here, we implement a bootstrap validation framework similar to Ascarza (2018) to ensure that results are not driven by a single split. Specifically, we generate 20 random train-test splits, with 70% of players assigned to the training set and the remaining 30% to the test set in each split. We then report the bootstrap mean and standard deviation of the long-term value value (defined in Equation (11)) for each policy.

For each split, we use the training set to estimate the model and derive the recommended policy, then evaluate the policy's performance on the test set using stepwise importance sampling, assuming the policy operates for up to 30 days. To ensure valid policy evaluation, we exclude two types of players from the test set. First, we exclude players who joined late in the experimental data and do not have complete observations for the full 30-day evaluation period. This exclusion is *exogenous*, determined solely by the player observation window rather than the company's DDA intervention. Second, following existing literature (e.g., Imbens and Rubin 2015, Gordon et al. 2019, Ellickson et al. 2023), we exclude players who are highly unlikely to be observed, as their inclusion could result in unrealistic policy values due to adjustments in stepwise importance sampling. Specifically, we exclude the 5% of players whose cumulative importance sampling weight, $\prod_{t=1}^{T_i} \hat{\pi}_b\left(a_{i,j}|\mathbf{x}_{i,j}\right)$, is below 0.01%, as these extreme cases could lead to excessively high and unrealistic policy values.

Table 3 presents the policy evaluation results, showing the total value of each policy over different time horizons, ranging from 1 to 30 days. As expected, the `Myopic` policy performs best on the first day but is quickly outperformed. When looking beyond the first day, the proposed `MRSR` method (column 2) consistently achieves the highest performance, highlighting the benefits of leveraging multiple responses to

[9] Further details regarding the estimation procedure are provided in Online Appendix B.1.

construct a comprehensive player state representation. Second, both CQL models trained with multi- and single-response state representations (column 2 and 3) outperform the `Standard CQL` model (column 4) trained on the original player characteristics in the long term. This result underscores the advantage of eliminating irrelevant information from the original characteristics to enhance policy learning.

Third, while the `Myopic` policy (column 5) achieves the highest performance on the first day after implementation, the other long-term policies consistently result in higher cumulative discounted coin spending over time. This highlights the importance of considering inter-temporal dynamics and the long-term impact of adjusting game difficulty when aiming to optimize sustained value.

Collectively, these findings demonstrate that the proposed MRSR method enables the company to enhance long-term player value more effectively by extracting critical information in the player characteristics related to long-term value. This approach leads to a 124% increase in 30-day value compared to the `Behavioral` policy (column 6), a 37% improvement over the state-of-the-art `Standard CQL` baseline method (column 4), and a 24% improvement over the `SRSR` policy (column 3).

Table 3 Cumulative Coin Spending Under Different DDA Policies

# Days	MRSR	SRSR	Standard CQL	Myopic	Behavioral
1	5.06 (0.04)	4.99 (0.04)	5.00 (0.04)	**5.20 (0.04)**	4.88 (0.03)
5	**12.11 (0.82)**	11.66 (0.80)	11.05 (0.64)	10.65 (0.29)	9.20 (0.07)
10	**18.09 (1.37)**	16.38 (1.23)	15.61 (1.02)	14.27 (0.42)	11.88 (0.09)
15	**21.96 (1.58)**	19.16 (1.38)	18.30 (1.16)	16.40 (0.48)	13.45 (0.10)
20	**27.23 (2.21)**	23.25 (1.98)	22.21 (1.79)	18.23 (0.66)	14.42 (0.11)
25	**31.23 (2.53)**	26.47 (2.28)	24.16 (2.00)	19.31 (0.71)	15.12 (0.12)
30	**35.08 (2.69)**	28.38 (2.51)	25.57 (2.13)	20.21 (0.74)	15.60 (0.12)

Note. We report the mean policy value across 20 fully random train-test splits, with standard deviations shown in parentheses. Results are presented with $\gamma_0 = 0.95$. Results of other discount factors are provided in Online Appendix D.3, and robustness checks for comprehensive results including additional benchmarks are available in Online Appendix E.

6.3.3. Discussion. While our offline evaluation results demonstrate the advantages of the proposed MRSR method in optimizing players' long-term coin spending, there are potential limitations due to the inherent challenges of policy evaluation with offline data. First, all the policies we constructed are restricted to assigning actions with non-zero observed probabilities in the offline dataset. While exploring regions with zero observations could potentially yield better outcomes, our primary objective is not to derive the absolute optimal policy but to show that the proposed state representation learning method enhances the efficacy of offline RL. Therefore, our results highlight the practical value and effectiveness of MRSR in real-world applications despite the imposed conservative constraints. Second, we exclude player trajectories

with extremely low cumulative importance sampling weights during policy evaluation. However, this is unlikely to pose a significant issue for the company, as such trajectories correspond to rare scenarios.[10]

Third, deploying the `MRSR` policy may alter the player state distribution, leading to changes in long-term player value. However, our empirical results provide a conservative estimate of the policy's long-term value. This is because dynamic difficulty adjustment under MRSR is likely to steer players into more valuable states over time, beyond the adjustments captured in the current data. As a result, the observed benefits likely represent a lower bound on the potential gains from fully implementing the MRSR policy in a live environment.

Finally, one might question whether cumulative coin spending is the most appropriate metric to optimize, given that many free-to-play games also generate revenue from advertising, which depends on player engagement rather than in-game spending. We focus on cumulative coin spending as it aligns most closely with the focal company's primary objective — maximizing spending behavior. Further, we avoid using the actual monetary values in our analysis to protect the company's proprietary information. In practice, companies can easily incorporate advertising revenue or other relevant metrics into the calculation of immediate customer value, $v_{i,t}$ (as described in Section 3.1), and use offline RL to optimize their specific objectives. This flexibility ensures that our framework can be adapted to suit different business models and goals.

7. Policy Explanation and Managerial Implications

In this section, we offer deeper insights into the benefits of the proposed representation learning method by (i) characterizing the state representations learned from different state construction methods and (ii) comparing the `MRSR` policy with the two reinforcement learning benchmarks, `SRSR` and `Standard CQL`. This comparison highlights the additional player behaviors captured by the proposed method, providing valuable managerial implications for decision-makers.

7.1. Insights from State Representations

We begin by analyzing the type of information captured by each representation learning approach. To do this, we randomly select one train-test split from the policy evaluation procedure. Next, we randomly sample 5,000 player characteristic observations. Using the encoding networks trained on the corresponding training set, we generate the associated state representations. This process is repeated for both the single-response and multi-response state representation learning methods to facilitate comparison.

[10] In addition, we observed that these players consistently exhibited low engagement over time and were more likely to receive lower difficulty levels under the company's DDA policy, yet their behaviors remained unchanged. As a result, a DDA policy — whether the company's original design or an RL-based policy—may have minimal to no impact on them. This suggests that our value estimates using the trimmed data might be slightly optimistic (albeit marginally, given the rarity of these customers) compared to the actual value across all customers. However, since these trajectories are uniformly excluded across all policies in the offline evaluation, our comparisons remain valid and fair.

To summarize the information contained in different state representations, we apply t-distributed stochastic neighbor embedding (t-SNE, Van der Maaten and Hinton (2008)) to project the original player characteristics, single-response representation, and multi-response representation into a two-dimensional space. Next, we apply the DBSCAN clustering algorithm (Ester et al. 1996) to identify player latent segments.

Figure 4 presents the t-SNE plots, where each cluster represents a distinct segment of players *with similar statuses at a given time*, based on either the original player characteristics (Panel a) or the state representations. First, both the multi-response and single-response representations (Panels b and c, respectively) identify significantly fewer latent segments than the original player characteristics. This suggests that the representation learning methods reduce information, likely filtering out redundant details that are irrelevant to long-term plater value under dynamic difficulty adjustments. (The redundancy is supported by Table 3, which shows that both polices outperform the `Standard CQL` policy that uses the original player characteristics directly.) Second, the multi-response representation identifies three distinct latent segments, whereas the single-response representation produces only two. This indicates that the single-response representation may lack the granularity needed to differentiate player statuses requiring distinct policy interventions. To further illustrate this point, we also present single-response representations adjusted to generate three clusters using a modified clustering algorithm (Panel d).

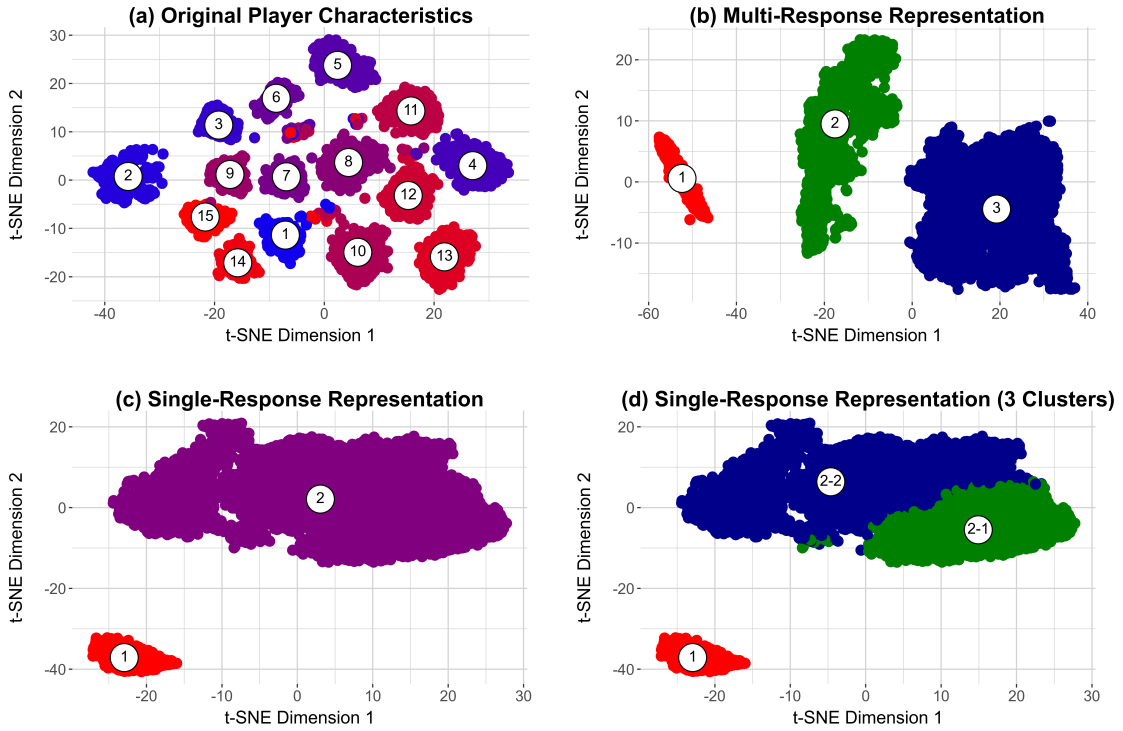
To better understand the additional variations captured by the multi-response representation compared to the single-response state representation, Table 4 summarizes the average values of key player characteristics for each segment. For a more detailed comparison, we use the single-response representations with three forced segments, where we find that the original Segment 2 is further divided into two sub-segments: Segment 2-1 and Segment 2-2.

Table 4 Summary of Player Characteristics

	Multi-Response State Representation			Single-Response State Representation		
Representative Player Characteristics	**Segment 1**	**Segment 2**	**Segment 3**	**Segment 1**	**Segment 2-1**	**Segment 2-2**
Is active last week?	0.64%	99.94%	100.00%	0.86%	99.93%	99.93%
# rounds played last week (if active last week)	1.00	22.56	51.56	2.00	41.77	40.26
Propensity of using coins last week (if active last week)	0.00%	9.35%	17.07%	0.00%	8.19%	17.10%
# coins used last week (if active last week)	0.00	11.63	21.75	0.00	6.86	23.36
Avg. points per round last week (if active last week)	0	36,507	38,377	6,065	39.849	36,633
Is active yesterday?	0.00%	2.25%	99.40%	0.00%	69.64%	60.00%
# rounds played yesterday (if active yesterday)	–	4.18	10.74	–	9.67	11.20
Propensity of using coins yesterday (if active yesterday)	–	0.00%	3.96%	–	1.46%	5.29%
# coins used yesterday (if active yesterday)	–	0.00	3.76	–	0.93	5.28
Avg. points per round yesterday (if active yesterday)	–	9,867	37,612	–	39,458	36,000
Short-term Responses (Under Difficulty Level 5)						
# days until next play	13.66	5.26	2.73	13.64	2.38	4.11
# rounds played during the day	4.56	6.85	10.66	4.54	9.44	9.45
# coins used during the day	1.15	2.26	3.60	1.14	1.08	4.21
Segment Size	9.34%	33.82%	56.84%	9.32%	29.58%	61.10%

Note. We present a representative subset of player characteristics in this table. A complete summary of player characteristics within each cluster in the table and the original Segment 2 of single-response representation which combines Segment 2-1 and Segment 2-2 is available in Table App-13 in the Online Appendix.

Figure 4 t-SNE Maps of Player Characteristics and Different State Representations



Note. Each dot in the plots denotes the state representation of one set of player characteristics. For both single-response and multi-response representations, each representation vector corresponds to the output of the learned encoding networks derived from one randomly sampled train-test split among the 20 train-test splits.

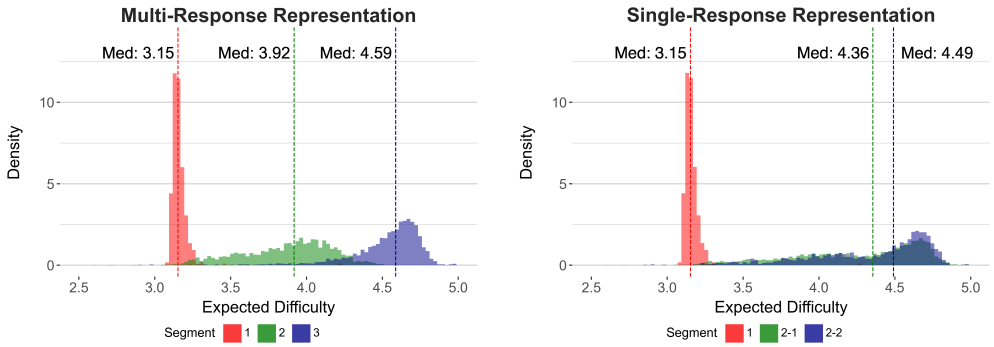
The multi-response representation identifies three distinct segments based on engagement, spending behavior, and game performance. Specifically, Segment 1 (column 2) represents a player status with very little activity in the week prior, no coin spending, and low points in completed rounds. Segment 2 (column 3) reflects players who recently became inactive: while they were moderately active with some coin spending and relatively high points earlier in the week, their activity, spending, and points dropped significantly on the previous day. Segment 3 (column 4) represents the most engaged state, characterized by high spending and high points both during the previous week and on the previous day. In contrast, the single-response representation differentiates between players with low engagement, minimal spending, and low performance (Segment 1) and those with higher engagement, more active spending, and greater ability (Segments 2-1 and 2-2) over the past week. Although Segments 2-1 and 2-2 differ in their overall coin spending behavior, they show no significant differences in previous-day activity; both sub-segments have similar values for the rounds played and the average points per round on the previous day.

Importantly, the short-term player responses between Segment 2 and Segment 3 of MRSR are significantly different. Players who have recently become inactive (Segment 2) tend to play fewer rounds on the current play day and take much longer to return (with a higher number of days until their next play) compared to the always-engaged segment (Segment 3). These findings clearly indicate that Segment 2 represents

a less valuable status compared to Segment 3. In contrast, Segments 2-1 and 2-2 of SRSR do not show significant differences in the number of rounds played during the day. In addition, while Segment 2-1 players spend fewer coins during the day than Segment 2-2 players, they are more likely to return sooner. Therefore, Segment 2-1 does not necessarily represent a lower long-term player value compared to Segment 2-2. These differences demonstrate that the MRSR method captures greater variations in player characteristics relevant to long-term player value compared to the SRSR method.

Next, we examine how the `MRSR` policy and `SRSR` policy adjust game difficulty for each segment. We calculate the expected difficulty level recommended by the `MRSR` policy for observations in each segment and present the resulting distributions in Figure 5. For the multi-response representation (the left panel), the policy assigns noticeably different difficulty levels to the three segments. For players in a low-activity state (Segment 1), the policy selects the easiest difficulty level. For players who became less active on the previous day (Segment 2), the policy recommends a lower difficulty level compared to consistently active players (Segment 3). This demonstrates the policy's ability to adjust difficulty based on recent player performance. In contrast, the single-response state representation (the right panel) allows the policy to distinguish between low- and high-engagement players but fails to differentiate effectively between players in Segments 2-1 and 2-2. The significant overlapping difficulty distributions for these segments indicate less precise and tailored difficulty adjustments.

Figure 5 Distributions of Expected Difficulty Levels Across Segments



Note. The plots show the expected difficulty distributions of policies learned using MRSR and SRSR, respectively, on 5,000 player characteristic samples. Each expected difficulty value is derived from the policy trained on the corresponding training set. The colors in the plots represent the three latent segments identified by the multi-response and single-response state representations, respectively. Dashed lines indicate the median expected difficulty within each latent segment.

These results suggest that incorporating multiple player responses into state representation learning helps the model capture meaningful variations in the data and improves policy learning. When the model considers only short-term spending behavior, it cannot identify players who are stuck, at a higher risk of churn, and require easier game difficulty. In contrast, including engagement and retention proxies enables the multi-response representations to identify this high-churn segment. This allows for more precise difficulty

adjustments that address player needs more effectively and improve retention. This approach gives managers actionable insights for developing a more responsive game design and increasing long-term player value.

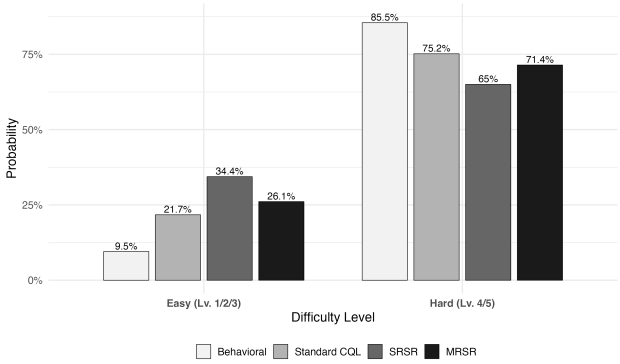
7.2. Interpreting Different Polices

Next, we examine how the three CQL policies adjust game difficulty and compare their effectiveness in balancing player engagement and monetization. We start by showing the difficulty distributions recommended by each policy for a "median player" — a hypothetical player with characteristics set at median values of the full dataset.[11]

Figure 6 shows the probability distributions of difficulty levels suggested by each of the policies. First, compared to the `Behavioral` policy, all three CQL policies significantly decrease the likelihood of assigning the hard mode (difficulty levels 4/5) and instead prioritize easier modes (difficulty levels 1/2/3). This shift explains why the behavioral policy performs suboptimally (as shown in Table 3) — many players likely faced levels that were too difficult, which may have reduced their engagement and future play.

Second, the `MRSR` policy achieves a more balanced approach to difficulty adjustment compared to the other RL policies. For instance, `Standard CQL` tends to make the game overly challenging, with an expected difficulty of 4.19, while `SRSR` makes it too easy, with an expected difficulty of 4.02. MRSR's balanced adjustment, with an expected difficulty of 4.16, aligns with its superior performance in off-policy evaluation and offers a more effective strategy for managing the trade-off between player engagement and monetization.

Figure 6 Difficulty Distribution for a Hypothetical Median Player



Note. Each probability suggested by a CQL-based model in the plot is calculated by averaging the corresponding probability suggested by the learned policies across 20 train-test splits. The difficulty distribution suggested by the behavioral policy is calculated through a logistic regression estimated using the full sample $\mathcal{D}$.

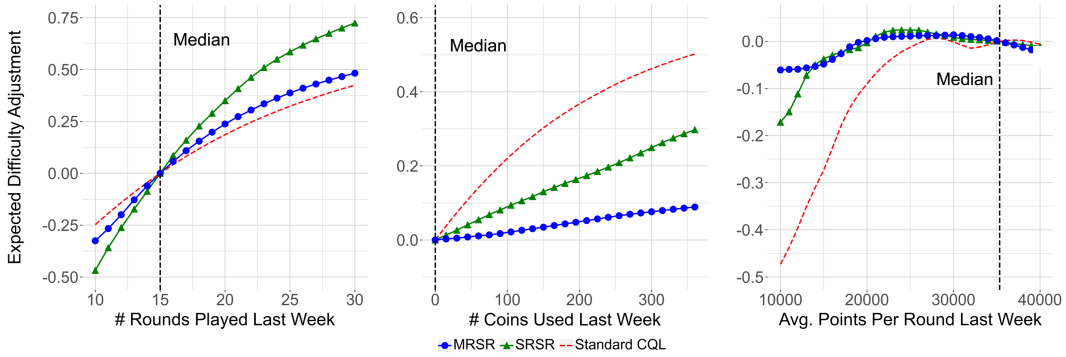
Next, we analyze how different CQL policies adjust difficulty based on player characteristics by calculating their marginal difficulty adjustments for individual characteristics. For this analysis, we vary one player

[11] Detailed information on the "median player" and specific values for each player characteristic are provided in Online Appendix F.2.

characteristic while holding all others constant at the values corresponding to a "median player." We adjust the focal characteristic around the median level and compute the resulting expected difficulty adjustment relative to the expected difficulty for the median player. A positive value indicates an increase in expected difficulty, while a negative value reflects a reduction. This analysis is conducted in two parts: first, focusing on characteristics related to behavior during the previous week, and second, focusing on characteristics related to behavior from the previous day.

Figure 7 presents the marginal difficulty adjustments for three key previous-week characteristics: number of rounds played (reflecting engagement level), number of coins spent (reflecting spending propensity), and average points per active round (reflecting progress ability). All three policies generally recommend higher difficulty for players with greater engagement and spending and lower difficulty for players who score significantly fewer points per round compared to the median player. Consistent with Figure 6, the `MRSR` policy exhibits a less aggressive approach to difficulty adjustments. It shows less sensitivity to changes in engagement compared to the `SRSR` policy and responds less to weekly coin spending and average points per round than both the `SRSR` and `Standard CQL` policies.

Figure 7 Difficulty Adjustment by Last-Week Variables

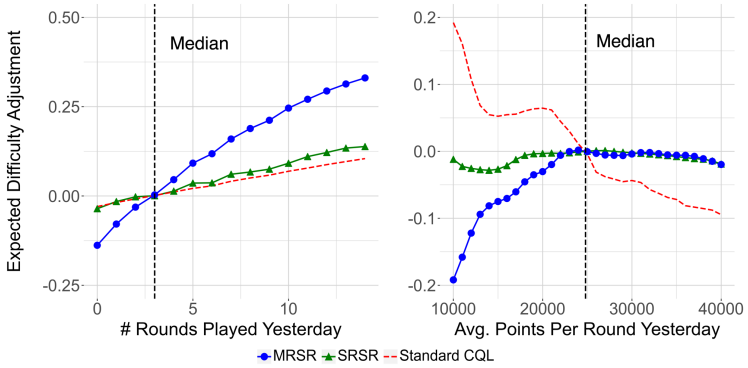


Note. Each plot reflects the expected difficulty adjustment by holding other player characteristics constant and adjusting the focal player characteristic around the value of the "median player." For each feature, the value of the "median player" is indicated by a dashed line. The expected difficulty adjustment is calculated by averaging the values suggested by the learned policies across 20 train-test splits.

The three policies also show distinct patterns in their difficulty adjustments based on player characteristics from the previous day. Figure 8 shows the marginal difficulty adjustments for the number of rounds played and the average points per active round from the previous day.[12] The `MRSR` policy demonstrates greater responsiveness to the number of rounds played compared to the other two policies. This observation aligns with the segment analysis in Table 4, where the multi-response representation learning identifies a key segment of players who became significantly inactive yesterday, enabling the development of a policy tailored to address this group.

[12] As the median value for coins spent during the previous week was zero, we exclude the marginal difficulty adjustment plot for coins spent yesterday for a median player.

Figure 8 Difficulty Adjustment Based on Yesterday's Variables



Note. Each plot reflects the expected difficulty adjustment by holding other player characteristics constant and adjusting the focal player feature around the value of the "median player." For each feature, the value of the "median player" is indicated by a dashed line. The expected difficulty adjustment is calculated by averaging the values suggested by the learned policies across 20 train-test splits.

Moreover, only the `MRSR` policy reduces difficulty for players who are stuck and exhibit significantly low scoring ability on the previous day. In contrast, the `SRSR` policy adjusts difficulty solely based on the player's rounds played during the previous day. This result highlights that the SRSR fails to capture importance of the scoring ability for difficulty adjustment. The `Standard CQL` policy, on the other hand, behaves counterintuitively by suggesting a sharp increase in difficulty for players with low scoring ability and a decrease for high-ability players. This contradicts the pattern shown in Figure 7. This inconsistency is likely due to the strong correlation between points scored yesterday and the average points from the previous week (partial correlation = 0.39, p-value< 0.001), which complicates the policy learning process and makes it difficult to distinguish between players with consistently low scoring ability and those who recently became stuck. These findings highlight the `MRSR` policy's ability to adjust difficulty more effectively for players experiencing recent engagement declines, as its multi-response representation learning captures more nuanced information about player behavior.

8. Conclusion and Future Directions

Reinforcement learning has become increasingly popular among companies aiming to dynamically personalize their marketing actions. Advances such as conservative Q-learning have helped address challenges related to developing RL policies using historical data. However, companies often rely on high-dimensional customer data that contain noisy or irrelevant information, making it difficult to develop effective policies. In addition, existing state construction methods often fail to capture long-term customer values, which are essential for optimizing policies. This paper tackles these issues by proposing a novel multi-response state representation learning method. The approach extracts policy-relevant information by leveraging multiple behavioral signals directly linked to long-term value, improving policy learning with high-dimensional customer data.

We validate the proposed method using experimental data from a F2P mobile puzzle game to dynamically adjust game difficulty. Our findings demonstrate that incorporating the proposed multi-response representation learning into a state-of-the-art offline RL method increases 30-day spending by 37% compared to directly using observed characteristics, and by 24% compared to using only immediate spending for state representation learning. These results underscore the importance of extracting policy-relevant information and incorporating multiple behavioral signals to optimize long-term values in offline reinforcement learning. Further policy analysis reveals several advantages of the multi-response approach in adapting to player dynamics, particularly by filtering out irrelevant information and identifying nuanced patterns critical for decision-making. Notably, the multi-response state representation identifies a unique player segment — those recently inactive or struggling after moderate activity in the past week. Unlike other benchmark policies that fail to account for these shifts, MRSR recommends lowering game difficulty for these players, effectively promoting re-engagement through adaptive and targeted difficulty adjustments.

Beyond the gaming setting, our framework has broad applicability across various domains where multiple behavioral signals can be incorporated into the augmented response space. For instance, in an e-commerce context, immediate behaviors such as clicks, product views, or time spent on a webpage provide valuable insights into customer engagement and help predict future value. Similarly, customer-initiated interactions with the company, such as using chatbots, submitting feedback forms, or engaging with customer service, can serve as critical signals. These behaviors not only indicate current engagement levels but may also reveal deeper intentions or issues, such as dissatisfaction or interest in specific products or services, which are essential for guiding long-term policy optimization. By identifying and integrating these behaviors into the augmented response space, the proposed method enables the development of more nuanced state representations that effectively capture the drivers of long-term customer value.

While the proposed MRSR method provides a valuable tool for companies to improve offline reinforcement learning and optimize customer long-term value, we acknowledge several limitations that present opportunities for future research. First, future research could investigate which short-term responses are most effective in supervised state representation learning across different marketing contexts and identify the specific elements of each response that contribute to better state representation. In many cases, the objective may focus on optimizing within-session engagement rather than long-term customer value, as seen in online media platforms (Song et al. 2019, Rafieian 2023). In such contexts, companies may need to prioritize immediate responses that are more closely tied to session retention or next-stage engagement within the same session. Furthermore, it is worth exploring robust approaches to avoid negative transfer by excluding irrelevant short-term responses. This may include modeling the relevance of each response to long-term value to enhance state representation effectiveness (e.g., Lee et al. 2016, 2018, Meng et al. 2021) or optimizing the weights assigned to different response prediction tasks (e.g., Kendall et al. 2018, Liu et al. 2019, Lin et al. 2019).

Second, while our method is developed within an offline policy learning framework, where the company aims to derive a dynamic policy from existing data, future research could explore modifications for compatibility with online policy learning (e.g., Wei et al. 2017, Wagenmaker and Pacchiano 2023). In an online setting, the need for conservative Q-learning could be removed, allowing the use of advanced reinforcement learning methods optimized for real-time learning. Currently, multi-response state representation learning is pre-trained before performing conservative Q-learning. However, in an online setting with real-time feedback, both the encoding network and the deep Q-network could be updated simultaneously. Future research could examine ways to integrate these steps into a unified online learning process.

Finally, while our empirical results highlight the benefits of multi-response state representations for dynamic game difficulty adjustment, future research could extend and adapt this approach to a variety of other marketing contexts. For example, in retail, the proposed method could be used to dynamically tailor personalized promotional offers based on customers' past transactions. In video-on-demand services, it could optimize customer engagement by refining content recommendations informed by historical viewing behaviors. Exploring these applications would provide valuable insights into its generalizability to enhance customer experience and drive long-term value across marketing domains.

Overall, we believe that the proposed MRSR method provides a robust approach for designing dynamic policies to optimize long-term customer outcomes, even when such behaviors are rare or when capturing dynamic heterogeneity is challenging. We encourage marketing researchers to further explore this method and examine its application across diverse business contexts, contributing to both the advancement of methodological innovation and the practical implementation of dynamic marketing policies.

Funding and Competing Interests

All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript. The authors have no funding to report.

References

Abel D, Arumugam D, Lehnert L, Littman M (2018) State abstractions for lifelong reinforcement learning. *International Conference on Machine Learning*, 10–19 (PMLR).

Abel D, Hershkowitz D, Littman M (2016) Near optimal behavior via approximate state abstraction. *International Conference on Machine Learning*, 2915–2923 (PMLR).

Allen C, Parikh N, Gottesman O, Konidaris G (2021) Learning markov state abstractions for deep reinforcement learning. *Advances in Neural Information Processing Systems* 34:8229–8241.

Amano T, Simonov A (2024) What makes players pay? an empirical investigation of in-game lotteries. *Available at SSRN 4355019* .

Anand A, Racah E, Ozair S, Bengio Y, Côté MA, Hjelm RD (2019) Unsupervised state representation learning in atari. *Advances in neural information processing systems* 32.

Andrew B, Richard S S (2018) Reinforcement learning: An introduction. *The MIT Press* .

Ascarza E (2018) Retention futility: Targeting high-risk customers might be ineffective. *Journal of marketing Research* 55(1):80–98.

Ascarza E, Hardie BG (2013) A joint model of usage and churn in contractual settings. *Marketing Science* 32(4):570–590.

Ascarza E, Netzer O, Hardie BG (2018) Some customers would rather leave without saying goodbye. *Marketing Science* 37(1):54–77.

Ascarza E, Netzer O, Runge J (2025) Personalized game design for improved user retention and monetization in freemium games. *International Journal of Research in Marketing* .

Bennouna A, Pachamanova D, Perakis G, Skali Lami O (2024) Learning the minimal representation of a continuous state-space markov decision process from transition data. *Management Science* .

Castelo-Branco B, Manchanda P (2023) Is video gaming addictive?: An empirical analysis. *Available at SSRN 4714079* .

Chen N, Elmachtoub AN, Hamilton ML, Lei X (2021) Loot box pricing and design. *Management Science* 67(8):4809–4825.

Ellickson PB, Kar W, Reeder JC (2022) Estimating marketing component effects: Double machine learning from targeted digital promotions. *Marketing Science* 0(0).

Ellickson PB, Kar W, Reeder III JC (2023) Estimating marketing component effects: Double machine learning from targeted digital promotions. *Marketing Science* 42(4):704–728.

Ester M, Kriegel HP, Sander J, Xu X, et al. (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. *kdd*, volume 96, 226–231.

Fader PS, Hardie BG, Lee KL (2005) "Counting your customers" the easy way: An alternative to the Pareto/NBD model. *Marketing Science* 24(2):275–284.

Fujimoto S, Hoof H, Meger D (2018) Addressing function approximation error in actor-critic methods. *International conference on machine learning*, 1587–1596 (PMLR).

Fujimoto S, Meger D, Precup D (2019) Off-policy deep reinforcement learning without exploration. *International conference on machine learning*, 2052–2062 (PMLR).

Gelada C, Kumar S, Buckman J, Nachum O, Bellem are MG (2019) Deepmdp: Learning continuous latent space models for representation learning. *International conference on machine learning*, 2170–2179 (PMLR).

Goli A, Reiley DH, Zhang H (2024) Personalizing ad load to optimize subscription and ad revenues: Product strategies constructed from experiments on pandora. *Marketing Science* .

Gönül F, Shi MZ (1998) Optimal mailing of catalogs: a new methodology using estimable structural dynamic programming models. *Management Science* 44(9):1249–1262.

Gordon BR, Zettelmeyer F, Bhargava N, Chapsky D (2019) A comparison of approaches to advertising measurement: Evidence from big field experiments at facebook. *Marketing Science* 38(2):193–225.

Hanna JP, Niekum S, Stone P (2021) Importance sampling in reinforcement learning with an estimated behavior policy. *Machine Learning* 110(6):1267–1317.

Hauser JR, Liberali G, Urban GL (2014) Website morphing 2.0: Switching costs, partial exposure, random exit, and when to morph. *Management science* 60(6):1594–1616.

Hauser JR, Urban GL, Liberali G, Braun M (2009) Website morphing. *Marketing Science* 28(2):202–223.

Hausknecht M, Stone P (2015) Deep recurrent q-learning for partially observable mdps. *2015 aaai fall symposium series*.

Hong J, Dragan A, Levine S (2023) Offline rl with observation histories: Analyzing and improving sample complexity. *arXiv preprint arXiv:2310.20663* .

Huang TW, Ascarza E (2024) Doing more with less: Overcoming ineffective long-term targeting using short-term signals. *Marketing Science* 43(3):863–884.

Huang Y, Jasin S, Manchanda P (2019) "level up": Leveraging skill and engagement to maximize player game-play in online video games. *Information Systems Research* 30(3):927–947.

Imbens GW, Rubin DB (2015) *Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction* (Cambridge University Press).

Kendall A, Gal Y, Cipolla R (2018) Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7482–7491.

Khan R, Lewis M, Singh V (2009) Dynamic customer management and the value of one-to-one marketing. *Marketing Science* 28(6):1063–1079.

Kingma DP (2014) Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Ko R, Uetake K, Yata K, Okada R (2024) When to target customers? retention management using dynamic off-policy policy learning. *Available at SSRN 4293532* .

Kumar A, Zhou A, Tucker G, Levine S (2020) Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems* 33:1179–1191.

Lee G, Yang E, Hwang S (2016) Asymmetric multi-task learning based on task relatedness and loss. *International conference on machine learning*, 230–238 (PMLR).

Lee HB, Yang E, Hwang SJ (2018) Deep asymmetric multi-task feature learning. *International Conference on Machine Learning*, 2956–2964 (PMLR).

Lee SY, Sudhir K, Uetake K (2024) A structural model of consumer utility generation in virtual environments. *Working paper* .

Levine S, Kumar A, Tucker G, Fu J (2020) Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *ArXiv* abs/2005.01643.

Li J, Lu H, Wang C, Ma W, Zhang M, Zhao X, Qi W, Liu Y, Ma S (2021) A difficulty-aware framework for churn prediction and intervention in games. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 943–952.

Liberali G, Ferecatu A (2022) Morphing for consumer dynamics: Bandits meet hidden markov models. *Marketing Science* 41(4):769–794.

Lin X, Zhen HL, Li Z, Zhang QF, Kwong S (2019) Pareto multi-task learning. *Advances in neural information processing systems* 32.

Liu S, Johns E, Davison AJ (2019) End-to-end multi-task learning with attention. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 1871–1880.

Liu X (2023) Dynamic coupon targeting using batch deep reinforcement learning: An application to livestream shopping. *Marketing Science* 42(4):637–658.

Ma L, Sun B, Kekre S (2015) The squeaky wheel gets the grease—an empirical analysis of customer voice and firm intervention on twitter. *Marketing Science* 34(5):627–645.

Meng Z, Yao X, Sun L (2021) Multi-task distillation: Towards mitigating the negative transfer in multi-task learning. *2021 IEEE International Conference on Image Processing (ICIP)*, 389–393 (IEEE).

Miao J, Jain S (2024) Designing loot boxes: Implications for profits and welfare. *Marketing Science* .

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

Naumzik C, Feuerriegel S, Weinmann M (2022) I will survive: Predicting business failures from customer ratings. *Marketing Science* 41(1):188–207.

Netzer O, Lattin JM, Srinivasan V (2008) A hidden markov model of customer relationship dynamics. *Marketing science* 27(2):185–204.

Rafieian O (2023) Optimizing user engagement through adaptive ad sequencing. *Marketing Science* 42(5):910–933.

Rafieian O, Kapoor A, Sharma A (2024) Multiobjective personalization of marketing interventions. *Marketing Science* .

Reinartz WJ, Kumar V (2003) The impact of customer relationship characteristics on profitable lifetime duration. *Journal of marketing* 67(1):77–99.

Rutz O, Aravindakshan A, Rubel O (2019) Measuring and forecasting mobile game app engagement. *International Journal of Research in Marketing* 36(2):185–199.

Simester D, Timoshenko A, Zoumpoulis SI (2020) Efficiently evaluating targeting policies: Improving on champion vs. challenger experiments. *Management Science* 66(8):3412–3424.

Simester DI, Sun P, Tsitsiklis JN (2006) Dynamic catalog mailing policies. *Management science* 52(5):683–696.

Song Y, Sahoo N, Ofek E (2019) When and how to diversify—a multicategory utility model for personalized content recommendation. *Management Science* 65(8):3737–3757.

Song Y, Sun T (2024) Ensemble experiments to optimize interventions along the customer journey: A reinforcement learning approach. *Management Science* 70(8):5115–5130.

Song Y, Wang W, Yao S (2024) Customer acquisition via explainable deep reinforcement learning. *Information Systems Research* .

Van der Maaten L, Hinton G (2008) Visualizing data using t-sne. *Journal of machine learning research* 9(11).

Wagenmaker A, Pacchiano A (2023) Leveraging offline data in online reinforcement learning. *International Conference on Machine Learning*, 35300–35338 (PMLR).

Wang L, Lowry PB, Luo X, Li H (2023a) Moving consumers from free to fee in platform-based markets: an empirical study of multiplayer online battle arena games. *Information Systems Research* 34(1):275–296.

Wang W, Li B, Luo X, Wang X (2023b) Deep reinforcement learning for sequential targeting. *Management Science* 69(9):5439–5460.

Wang Z, Dai Z, Póczos B, Carbonell J (2019) Characterizing and avoiding negative transfer. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11293–11302.

Wei CY, Hong YT, Lu CJ (2017) Online reinforcement learning in stochastic games. *Advances in Neural Information Processing Systems* 30.

Wu S, Zhang HR, Ré C (2020) Understanding and improving information transfer in multi-task learning. *International Conference on Learning Representations*, URL `https://openreview.net/forum?id=SylzhkBtDB`.

Xue S, Wu M, Kolen J, Aghdaie N, Zaman KA (2017) Dynamic difficulty adjustment for maximized engagement in digital games. *Proceedings of the 26th international conference on world wide web companion*, 465–471.

Yang J, Eckles D, Dhillon P, Aral S (2024) Targeting for long-term outcomes. *Management Science* 70(6):3841–3855.

Yoganarasimhan H, Barzegary E, Pani A (2023) Design and evaluation of optimal free trials. *Management Science* 69(6):3220–3240.

Zhang A, McAllister R, Calandra R, Gal Y, Levine S (2021) Learning invariant representations for reinforcement learning without reconstruction. *International Conference on Learning Representations* .

Zhang Y, Bradlow ET, Small DS (2015) Predicting customer value using clumpiness: From rfm to rfmc. *Marketing Science* 34(2):195–208.

Zhao Y, Yang S, Shum M, Dutta S (2022) A dynamic model of player level-progression decisions in online gaming. *Management Science* 68(11):8062–8082.

Zhu Y, Simester D, Parker JA, Schoar A (2020) Dynamic marketing policies: Constructing markov states for reinforcement learning. *Available at SSRN 3633870* .

Online Appendix

A. Summary Statistics of The Empirical Data

In this section, we present summary statistics for the variables described in Section 5.3, including the assigned difficulty levels implemented by the company, observed player characteristics, and short-term player responses. Table App-1 summarizes the distribution of assigned difficulty levels, which were driven by the two-wave field experiments. Notably, the majority of play-day samples are assigned the highest difficulty level, as more engaged players were more likely to be observed and received the most difficult game setting. This imbalance highlights the potential for substantial extrapolation error (discussed in Section 4.2), underscoring the importance of employing conservative learning approaches in this context.

Table App-1 Summary of Assigned Difficulty Levels ($a_{i,t}$)

Variable	Mean	S.D.	Min	Q 25%	Median	Q 75%	Max
Difficulty	4.33	1.32	1	5	5	5	5

Table App-2 provides a summary of player characteristics used for DDA policy learning prior to a player being assigned a difficulty level. It includes various variables reflecting players' gameplay behavior and ability, such as the number of active days, rounds played, coins used, and points per round since starting the game. Additionally, it presents statistics on recent gameplay activity, including variables from the previous day and the last week. The table also includes binary indicators for days of the week to capture potential temporal patterns in player behavior. All variables are summarized with their mean, standard deviation, minimum, quartiles, and maximum values. The playing behaviors exhibit substantial variation across player characteristics, highlighting the potential for implementing a more personalized DDA policy. Notably, the "Has churned?" variable shows no variation because difficulty levels were only assigned to active players. This comprehensive summary underscores the variability in player engagement and performance metrics.

Table App-3 summarizes player characteristics after a difficulty level was assigned. Compared to Table App-2 (before the difficulty assignment), notable differences are observed in the "Has churned?" variable, which now exhibits variation, with a mean of 7%, indicating that some players stopped playing. Some measures of recent gameplay, such as rounds played and points per round yesterday or last week, show slight shifts, suggesting potential changes in player behavior following the difficulty adjustment. Overall, the assignment of difficulty levels appears to have introduced variability in churn status and influenced player engagement and performance metrics.

Table App-2 Summary of Player Characteristics Prior to Assigning Difficulty Levels ($\mathbf{x}_{i,t}$)

Variable	Mean	S.D.	Min	Q 25%	Median	Q 75%	Max
Has churned?	0	0	0	0	0	0	0
# active days since start	29.93	24.78	1	11	22	42	170
# days since start	56.18	38.61	1	24	49	82	183
# rounds played since start	363.64	362.69	20	111	237	492	3,265
# coins used since start	150.56	500.63	0	70	70	140	17,570
Winning rate since start	0.41	0.17	0.02	0.29	0.41	0.53	1.00
Avg. stars per round since start	1.16	0.26	0.23	0.99	1.14	1.31	2.85
Avg. points per round since start	34,762.37	4,946.48	11,390.16	31,445.67	34,902.62	38,103.15	143,586.48
Avg. snake length per round since start	5.16	0.38	1.98	4.92	5.18	5.41	8.78
Max level ever reached	50.13	23.83	20	35	40	60	274
# coins in storage	25.59	71.92	0	0	0	30	4,430
# days since last gameplay	3.54	7.80	1	1	1	3	170
Is Monday?	0.14	0.34	0	0	0	0	1
Is Tuesday?	0.14	0.34	0	0	0	0	1
Is Wednesday?	0.14	0.35	0	0	0	0	1
Is Thursday?	0.14	0.35	0	0	0	0	1
Is Friday?	0.14	0.35	0	0	0	0	1
Is Saturday?	0.14	0.35	0	0	0	0	1
Is Sunday?	0.14	0.36	0	0	0	0	1
Is active yesterday?	0.58	0.49	0	0	1	1	1
# rounds played yesterday	6.27	9.02	0	0	3	10	91
# coins used yesterday	2.82	27.84	0	0	0	0	2,880
Winning rate yesterday	0.15	0.26	0.00	0.00	0.00	0.25	1.00
Avg. stars per round yesterday	0.54	0.59	0.00	0.00	0.38	1.00	3.00
Avg. points per round yesterday	21,506.26	21,002.36	0.00	0.00	24,800.00	37,286.96	216,800.00
Avg. snake length per round yesterday	2.96	2.59	0.00	0.00	4.45	5.15	14.00
# active days last week	3.64	2.19	0	2	4	5	7
# rounds played last week	37.93	38.08	0	10	29	52	389
# coins used last week	17.27	95.48	0	0	0	0	7,280
Winning rate last week	0.29	0.26	0.00	0.04	0.23	0.48	1.00
Avg. stars per round last week	0.92	0.50	0.00	0.67	1.00	1.21	3.00
Avg. points per round last week	33,871.07	14,916.28	0.00	29,212.29	35,329.41	41,350.00	273,700.00
Avg. snake length per round last week	4.66	1.60	0.00	4.67	5.05	5.43	14.00

The "Has churned?" variable variable shows no variation before the intervention in each transition observation, as the company could only assign a difficulty level to players who were still "alive".

Table App-3 Summary of Player Characteristics After Assigning Difficulty Levels ($\mathbf{x}_{i,t+1}$)

Variable	Mean	S.D.	Min	Q 25%	Median	Q 75%	Max
Has churned?	0.07	0.26	0	0	0	0	1
# active days since start	30.93	24.78	2	12	23	43	171
# days since start	61.01	39.60	2	29	54	87	214
# rounds played since start	372.34	363.90	21	119	246	501	3,289
# coins used since start	153.99	509.73	0	70	70	140	17,570
Winning rate since start	0.41	0.16	0.02	0.29	0.40	0.52	1.00
Avg. stars per round since start	1.15	0.24	0.23	0.99	1.14	1.30	2.85
Avg. points per round since start	34,928.31	4,946.09	11,102.53	31,631.17	35,075.00	38,260.96	145,433.83
Avg. snake per round length since start	5.16	0.37	1.97	4.93	5.18	5.41	8.85
Max level ever reached	51.12	23.70	20	39	41	61	279
# coins in storage	24.84	71.22	0	0	0	30	4,430
# days since last gameplay	4.83	8.30	1	1	1	3	31
Is Monday?	0.15	0.35	0	0	0	0	1
Is Tuesday?	0.14	0.35	0	0	0	0	1
Is Wednesday?	0.14	0.35	0	0	0	0	1
Is Thursday?	0.14	0.35	0	0	0	0	1
Is Friday?	0.14	0.35	0	0	0	0	1
Is Saturday?	0.14	0.35	0	0	0	0	1
Is Sunday?	0.15	0.36	0	0	0	0	1
Is active yesterday?	0.56	0.50	0	0	1	1	1
# rounds played yesterday	5.96	8.82	0	0	2	9	91
# coins used yesterday	2.27	25.83	0	0	0	0	2,880
Winning rate yesterday	0.14	0.25	0.0	0.0	0.0	0.2	1.0
Avg. stars per round yesterday	0.52	0.58	0.00	0.00	0.21	1.00	3.00
Avg. points per round yesterday	20,775.80	21,100.66	0.00	0.00	23,150.00	36,895.74	216,800.00
Avg. snake length per round yesterday	2.85	2.60	0.00	0.00	4.33	5.13	14.00
# active days last week	3.52	2.30	0	2	4	5	7
# rounds played last week	36.99	39.06	0	7	27	53	389
# coins used last week	15.96	94.61	0	0	0	0	7,280
Winning rate last week	0.26	0.26	0.00	0.00	0.20	0.45	1.00
Avg. stars per round last week	0.86	0.53	0.00	0.56	0.96	1.17	3.00
Avg. points per round last week	32,243.04	16,639.54	0.00	27,479.17	34,962.73	41,253.57	273,700.00
Avg. snake length per round last week	4.40	1.89	0.00	4.58	5.00	5.40	14.00

Table App-4 presents the summary statistics for short-term player responses. Notably, coin spending behavior is highly sparse, with over 75% of observations showing no spending at all. This highlights the limitations the company could face when relying solely on immediate coin spending as the supervision target for supervised state representation learning.

Table App-4 Summary of Short-Term Player Responses ($\mathbf{r}_{i,t}$)

Variable	Mean	S.D.	Min	Q 25%	Median	Q 75%	Max
# coins spent in the day ($v_{i,t}$)	3.43	30.69	0	0	0	0	2900
# days until next play ($r_{i,t}^{\text{Days}}$)	4.83	8.30	1	1	1	3	31
# rounds played in the day ($r_{i,t}^{\text{Rounds}}$)	8.70	8.42	1	3	6	11	92

B. Implementation Details

B.1. Behavioral Policy Estimation

In this section, we outline the estimation process for the behavioral policy used in policy evaluation. Based on the field experiment design of the gaming company, we identify three experimental stages: (i) Stage 1, from June 11, 2014, to July 15, 2014; (ii) Stage 2, from July 16, 2014, to August 9, 2014; and (iii) Stage 3, from August 10, 2014, to August 31, 2014. Players are further categorized into four groups based on the timing of their treatment assignment: (i) Group 1, players beginning treatment in Stage 1; (ii) Group 2, players beginning treatment in Stage 2; (iii) Group 3, players beginning treatment in Stage 3; and (iv) Group 4, players assigned to the control group.

This classification results in twelve unique combinations, denoted as $\{(\text{Group } g, \text{Stage } s)\}_{g=1,2,3,4,\ s=1,2,3}$. Leveraging this structure, we decompose the behavioral policy $\pi_b(a_{i,t}|\mathbf{x}_{i,t})$ as follows:

$$\pi_b(a_{i,t}|\mathbf{x}_{i,t}) = \sum_{g=1}^{4}\sum_{s=1}^{3} \mathbb{P}_b(a_{i,t}|\mathbf{x}_{i,t}, i \in \text{Group } g, t \in \text{Stage } s)\mathbb{P}_b(i \in \text{Group } g, t \in \text{Stage } s|\mathbf{x}_{i,t}). \quad \text{(App-1)}$$

Here, $\mathbb{P}_b(a_{i,t}|\mathbf{x}_{i,t}, i \in \text{Group } g, t \in \text{Stage } s)$ represents the probability that a player i with observed characteristics $\mathbf{x}i, t$, assigned to Group g, receives difficulty level $a_{i,t}$ in Stage s. This probability is *known* from the company's difficulty assignment rule and is restricted to values in $0, 1$.

The second term, $\mathbb{P}_b(i \in \text{Group } g, t \in \text{Stage } s|\mathbf{x}_{i,t})$, represents the *posterior probability* that a player with observed characteristics $\mathbf{x}_{i,t}$ belongs to Group g and Stage s. Unlike the first term, which is deterministic, this probability introduces uncertainty and requires estimation. To estimate it, we use a logistic regression model combined with 5-fold cross-fitting.

Using the resulting estimates, we compute $\hat{\pi}_b(a_{i,t}|\mathbf{x}_{i,t})$ based on Equation (App-1) for both the training and testing sets. These estimates are subsequently used for conservative Q-learning and offline policy evaluation, respectively.

B.2. Implementation of Deep Learning Models

As described in Section 6.1, we implement a five-layer fully connected neural network (four hidden layers and one output layer) with 16 units per hidden layer for state representation learning. Each hidden unit uses the Rectified Linear Unit (ReLU) activation function. The resulting structure for each network is as follows:

- **Encoding Network**: The encoding network has an input dimension of 32 (representing the 32 player characteristics) and an output dimension of 15 (representing the state representation dimension). A linear activation function is applied between the last hidden layer and the output player. This structure results in a total of 1,599 parameters.
- **Response Prediction Networks**: The response prediction network for each short-term response has an input dimension of 20 (combining the 15-dimensional state representation and the five difficulty levels)

and an output dimension of 1 (representing the short-term response). A linear activation function is applied between the last hidden layer and the output layer. This results in a total of 1,169 parameters.

- **Action Prediction Network**: The action prediction network has an input dimension of 30 (combining the 15-dimensional state representations from before and after the playday) and an output dimension of 5 (representing the five difficulty levels), with a total of 1,397 parameters. A Softmax activation function is applied to the output layer to generate probabilities.
- **Sequence Order Prediction Network**: The sequence order prediction network has an input dimension of 30 (combining the dimensions of two state representations) and an output dimension of 1 (representing the sequence order label), with a total of 1,329 parameters. A Softmax activation function is applied to the output layer to produce the probability.

Note that we use the same network structures for all corresponding networks in the `SRSR` benchmark described in Section 6.2.

For the conservative Q-networks, we implement a five-layer fully connected neural network (four hidden layers and one output layer) with 16 units per hidden layer, using the ReLU activation function for each hidden unit. A linear activation function is applied between the final hidden layer and the output layer. The input dimension is 15 for both `MRSR` and `SRSR` in Section 6.2 (representing the state representation dimension), with an output dimension of 5 (corresponding to five difficulty levels), resulting in a total of 1,157 parameters. For `Standard CQL` and `Myopic` in Section 6.2, the CQL networks both have an input dimension of 32 (representing 32 player characteristics) and an output dimension of 5, resulting in a total of 1,429 parameters.

In Equation (5), we assign equal weights to $\mathcal{L}_{\text{Response}}$, $\mathcal{L}_{\text{Inv}}$, and $\mathcal{L}_{\text{Ratio}}$ by setting $\alpha = \beta = 1$. Validity checks in Online Appendix C confirm that this weight configuration preserves both the immediate coin spending predictability and the Markov property in the state representations. For the conservative Q-learning component, we set $\eta = 10$, consistent with Kumar et al. (2020).[1]

To calibrate the deep neural networks, we use mini-batch stochastic gradient descent with a batch size of 1,024. For the state representation learning step, the initial learning rate is set to 0.001, while for the subsequent conservative Q-learning, it is set to 0.00001. The Adam optimizer (Kingma 2014) is employed to adaptively adjust the learning rate during training. An early stopping rule is applied, stopping training if the total loss difference between two consecutive epochs is less than 0.0001. This approach improves computational efficiency and mitigates overfitting, as training stops once the model performance stabilizes. We also provide robustness checks for various neural network architectures and sizes in Online Appendix D. While the results vary slightly across different model configurations, the conclusions remain consistent with those obtained using the above configuration.

[1] Kumar et al. (2020) uses $\eta = 5$, which corresponds to $\eta = 10$ in our setting as they apply a scaling factor of 0.5 to $\mathcal{L}_{\text{Bellman}}$ in their formulation.

C. Validity Checks for Learned State Representation

C.1. Predictive Ability of State Representation Learning Models

In this section, we validate that the learned state representations preserve both the Markov property and the predictability of short-term coin spending inherent in the original observed player characteristics. To conduct this validation, we randomly split the data into training and testing sets. Using the training set, we train an immediate coin spending prediction model, an action prediction network, and a sequence order prediction network with the observed player characteristics. We then compare the root mean squared error (RMSE) for immediate coin spending, as well as the cross-entropy losses for the action and sequence order prediction tasks, against those of the corresponding components in the multi-response and single-response representations on the testing set. To ensure consistency, all models use the same neural network architecture, as outlined in Online Appendix B.2. To enhance robustness, we repeat this process across 20 different splits and report the average cross-entropy losses and RMSE values.

Table App-5 presents the results. First, the RMSE values for immediate coin spending prediction do not show statistically significant differences between models based on observed player characteristics, single-response state representation, and multi-response state representation. These findings confirm that both learned state representations retain the predictability of immediate coin spending. Second, the cross-entropy losses for the action prediction models (column 2) and sequence order prediction models (column 3) show no statistically significant differences, indicating that both single-response and multi-response state representations preserve the Markov property.

Table App-5 Prediction Validity of Different Representations

	Response Prediction Loss	Action Prediction Loss	Sequence Order Prediction Loss
Observed Player Characteristics	0.9675 (0.0268)	0.5798 (0.0492)	0.1024 (0.0603)
Single-Response State Representation	0.9312 (0.0271)	0.5220 (0.0232)	0.1049 (0.1276)
Multi-Response State Representation	0.9172 (0.0542)	0.5945 (0.0777)	0.1235 (0.0951)

We report the mean RMSE for immediate coin spending prediction, and the mean cross-entropy loss for action prediction and sequence order prediction across 20 fully random train-test splits, with standard deviation shown in parentheses.

C.2. Different Weights for Markov Preservation Tasks

In this section, we assess the sensitivity of immediate coin spending predictability and the preservation of the Markov property to varying weight configurations within the MRSR framework.

For this analysis, we train the multi-response state representations using a range of weight configurations, $(\alpha, \beta) \in \{(0,0), (0.25, 0.25), \ldots, (3,3)\}$. We then evaluate the RMSE for immediate coin spending prediction and compute the cross-entropy loss for action prediction and sequence order prediction, utilizing the corresponding components of the trained neural networks.

The results, summarized in Table App-6, reveal several key insights. First, for immediate coin spending predictability, the prediction losses remain consistent across all weight configurations, with no statistically

significant differences observed. This indicates that preserving the Markov property does not compromise immediate coin spending predictability. Second, for action prediction and sequence order prediction, introducing positive values for α and β significantly reduces the corresponding prediction losses, demonstrating the effectiveness of the supervised representation learning structure in maintaining the Markov property.

Moreover, when $\alpha \geq 0.75$ and $\beta \geq 0.75$, the prediction losses for action and sequence order prediction stabilize, showing no statistically significant differences across weight configurations. These findings confirm that the selected weight configuration ($\alpha = 1$ and $\beta = 1$) in the main analysis is sufficient to preserve both the Markov property and immediate coin spending predictability.

Table App-6 Prediction Validity of Different Task Weights

α	β	Response Prediction Loss	Action Prediction Loss	Sequence Order Prediction Loss
0.00	0.00	0.9131 (0.0525)	6.6647 (3.9280)	5.8758 (2.9055)
0.25	0.25	0.9101 (0.0571)	0.6440 (0.0589)	0.1898 (0.1059)
0.50	0.50	0.9231 (0.0632)	0.6334 (0.0699)	0.1740 (0.0942)
0.75	0.75	0.9234 (0.0654)	0.6231 (0.0780)	0.1602 (0.0995)
1.00	1.00	0.9172 (0.0542)	0.5945 (0.0777)	0.1235 (0.0951)
1.25	1.25	0.9156 (0.0542)	0.5707 (0.0767)	0.1174 (0.0890)
1.50	1.50	0.9112 (0.0544)	0.5524 (0.0726)	0.0774 (0.0678)
1.75	1.75	0.9224 (0.0713)	0.5595 (0.0714)	0.0971 (0.0848)
2.00	2.00	0.9096 (0.0551)	0.5307 (0.0452)	0.0773 (0.0580)
2.25	2.25	0.9101 (0.0539)	0.5418 (0.0622)	0.0663 (0.0377)
2.50	2.50	0.9108 (0.0583)	0.5257 (0.0309)	0.0820 (0.0578)
2.75	2.75	0.9112 (0.0533)	0.5155 (0.0206)	0.0702 (0.0383)
3.00	3.00	0.9194 (0.0635)	0.5268 (0.0418)	0.0830 (0.0583)

We report the mean RMSE for immediate coin spending prediction, and the mean cross-entropy loss for action prediction and sequence order prediction across 20 fully random train-test splits, with standard deviation shown in parentheses.

D. Robustness Checks

D.1. Depth of Neural Networks

In this section, we evaluate the robustness of our results by varying the depth of the neural networks to ensure that our findings are not overly dependent on a specific architecture. Specifically, we replicate the main analysis using seven-layer and nine-layer fully connected neural networks in place of the original five-layer networks.

Table App-7 summarizes the evaluation results. The overall patterns remain consistent with those obtained using the five-layer networks. Our proposed `MRSR` approach (column 2) consistently outperforms all other benchmarks, except on the first day following policy implementation (row 1 for each architecture), where the `Myopic` policy (column 5) achieves the highest performance. Furthermore, `SRSR` (column 3) consistently exceeds the performance of CQL in long-term outcomes across all neural network structures. Notably, the long-term policy values of `MRSR` and the benchmark policies show no statistically significant differences across the three neural network depths, confirming the robustness of our findings to changes in network architecture.

Table App-7 Cumulative Coin Spending with Varying Depths of Neural Network Architectures

# Days	MRSR	SRSR	Standard CQL	Myopic	Behavioral
5 Layers					
1	5.06 (0.04)	4.99 (0.04)	5.00 (0.04)	**5.20 (0.04)**	4.88 (0.03)
5	**12.11 (0.82)**	11.66 (0.80)	11.05 (0.64)	10.65 (0.29)	9.20 (0.07)
10	**18.09 (1.37)**	16.38 (1.23)	15.61 (1.02)	14.27 (0.42)	11.88 (0.09)
15	**21.96 (1.58)**	19.16 (1.38)	18.30 (1.16)	16.40 (0.48)	13.45 (0.10)
20	**27.23 (2.21)**	23.25 (1.98)	22.21 (1.79)	18.23 (0.66)	14.42 (0.11)
25	**31.23 (2.53)**	26.47 (2.28)	24.16 (2.00)	19.31 (0.71)	15.12 (0.12)
30	**35.08 (2.69)**	28.38 (2.51)	25.57 (2.13)	20.21 (0.74)	15.60 (0.12)
7 Layers					
1	5.09 (0.04)	4.99 (0.04)	5.01 (0.04)	**5.21 (0.06)**	4.88 (0.03)
5	**11.95 (0.41)**	11.75 (0.92)	11.45 (0.96)	10.73 (0.39)	9.20 (0.07)
10	**18.21 (0.70)**	17.28 (1.48)	15.99 (1.40)	14.52 (0.61)	11.88 (0.09)
15	**22.73 (0.84)**	20.76 (1.69)	18.65 (1.63)	16.74 (0.73)	13.45 (0.10)
20	**29.57 (1.20)**	26.21 (2.45)	22.59 (2.27)	18.95 (0.99)	14.42 (0.11)
25	**33.96 (1.43)**	28.90 (2.69)	25.22 (2.67)	20.18 (1.10)	15.12 (0.12)
30	**39.30 (1.88)**	32.12 (2.96)	28.42 (3.53)	21.15 (1.19)	15.60 (0.12)
9 Layers					
1	5.07 (0.04)	5.00 (0.04)	4.99 (0.03)	**5.19 (0.06)**	4.88 (0.03)
5	**13.16 (1.06)**	12.40 (1.10)	11.21 (0.67)	10.96 (0.29)	9.20 (0.07)
10	**19.37 (1.55)**	17.40 (1.62)	15.71 (1.11)	14.98 (0.45)	11.88 (0.09)
15	**23.07 (1.73)**	20.35 (1.83)	18.54 (1.33)	17.35 (0.53)	13.45 (0.10)
20	**30.34 (2.91)**	26.06 (3.16)	22.65 (2.08)	20.21 (0.78)	14.42 (0.11)
25	**34.80 (3.25)**	29.93 (3.67)	24.99 (2.39)	21.67 (0.88)	15.12 (0.12)
30	**38.62 (3.47)**	32.03 (3.89)	27.70 (2.90)	22.84 (0.94)	15.60 (0.12)

We report the mean policy value across 20 fully random train-test splits, with standard deviations shown in parentheses. Results are presented with neural network structure with 5 layers, 7 layers and 9 layers.

D.2. Dimensions of State Representations

Next, we assess the robustness of our results by varying the dimensions of state representations, ensuring that our findings are not overly dependent on the chosen state representation dimensions. Specifically, we replicate the MRSR and SRSR approaches from the main analysis, replacing the state representation dimension of 15 with dimensions of 20 and 25, respectively.

The policy evaluation results for different state representation dimensions are presented in Table App-8. The observed patterns remain consistent across all dimensions and align with the results from the main analysis (15 dimensions). The `MRSR` policy (column 2) consistently outperforms all other benchmarks, except on the first day following policy implementation (row 1 for each state representation dimension), where the `Myopic` policy (column 5) achieves the highest performance. Additionally, `SRSR` (column 3) consistently outperforms the CQL in long-term performance across all tested dimensions.

Importantly, no statistically significant differences are observed in the long-term policy values across the three different levels of state representation dimensions for both `MRSR` and `SRSR`. These findings confirm the robustness of our results to variations in state representation dimensions.

Table App-8 Cumulative Coin Spending with Different Dimensions for State Representation

# Days	MRSR	SRSR	Standard CQL	Myopic	Behavioral
15 Dimensions					
1	5.06 (0.04)	4.99 (0.04)	5.00 (0.04)	**5.20 (0.04)**	4.88 (0.03)
5	**12.11 (0.82)**	11.66 (0.80)	11.05 (0.64)	10.65 (0.29)	9.20 (0.07)
10	**18.09 (1.37)**	16.38 (1.23)	15.61 (1.02)	14.27 (0.42)	11.88 (0.09)
15	**21.96 (1.58)**	19.16 (1.38)	18.30 (1.16)	16.40 (0.48)	13.45 (0.10)
20	**27.23 (2.21)**	23.25 (1.98)	22.21 (1.79)	18.23 (0.66)	14.42 (0.11)
25	**31.23 (2.53)**	26.47 (2.28)	24.16 (2.00)	19.31 (0.71)	15.12 (0.12)
30	**35.08 (2.69)**	28.38 (2.51)	25.57 (2.13)	20.21 (0.74)	15.60 (0.12)
20 Dimensions					
1	5.06 (0.04)	4.95 (0.03)	5.00 (0.04)	**5.20 (0.04)**	4.88 (0.03)
5	**12.48 (0.95)**	11.27 (0.62)	11.05 (0.64)	10.65 (0.29)	9.20 (0.07)
10	**17.98 (1.47)**	15.81 (1.00)	15.61 (1.02)	14.27 (0.42)	11.88 (0.09)
15	**21.17 (1.62)**	18.60 (1.23)	18.30 (1.16)	16.40 (0.48)	13.45 (0.10)
20	**26.87 (2.36)**	23.26 (2.27)	22.21 (1.79)	18.23 (0.66)	14.42 (0.11)
25	**30.82 (2.70)**	25.33 (2.57)	24.16 (2.00)	19.31 (0.71)	15.12 (0.12)
30	**33.74 (2.89)**	27.51 (2.83)	25.57 (2.13)	20.21 (0.74)	15.60 (0.12)
25 Dimensions					
1	5.05 (0.03)	4.99 (0.04)	5.00 (0.04)	**5.20 (0.04)**	4.88 (0.03)
5	**11.92 (0.55)**	11.53 (0.67)	11.05 (0.64)	10.65 (0.29)	9.20 (0.07)
10	**17.94 (0.85)**	16.50 (1.10)	15.61 (1.02)	14.27 (0.42)	11.88 (0.09)
15	**22.22 (0.95)**	19.76 (1.36)	18.30 (1.16)	16.40 (0.48)	13.45 (0.10)
20	**28.76 (1.40)**	24.81 (2.13)	22.21 (1.79)	18.23 (0.66)	14.42 (0.11)
25	**32.07 (1.53)**	27.75 (2.43)	24.16 (2.00)	19.31 (0.71)	15.12 (0.12)
30	**36.36 (1.64)**	30.87 (2.91)	25.57 (2.13)	20.21 (0.74)	15.60 (0.12)

We report the mean policy value across 20 fully random train-test splits, with standard deviations shown in parentheses. Results are presented with state representation dimension of 15, 20 and 25.

D.3. Different Discount Factors

Finally, we assess the stability of our results across varying the discount factor γ_0, ensuring that our findings are not overly sensitive to this parameter. Specifically, we evaluate the results for $\gamma_0 \in 0.9, 0.85, 0.8$. Table App-9 presents the evaluation results. Overall, the patterns are consistent with those observed when $\gamma_0 = 0.95$. Our proposed `MRSR` policy (column 2) consistently outperforms all other benchmarks, except on the first day following policy implementation (row 1 for each discount factor), where the `Myopic` policy (column 5) performs best. Notably, when $\gamma_0 = 0.8$, reflecting a more short-term-focused perspective that prioritizes immediate player spending, `Standard CQL` fails to outperform the myopic policy after 30 days. This highlights the importance of leveraging multiple customer responses to capture information relevant to long-term value, even when the DDA policy adopts a more myopic approach.

Table App-9 Long-term Values of Varying Discount Factors

# Days	MRSR	SRSR	Standard CQL	Myopic	Behavioral
$\gamma_0 = 0.9$					
1	5.02 (0.04)	4.97 (0.03)	4.97 (0.05)	**5.20 (0.04)**	4.88 (0.03)
5	**11.01 (0.75)**	10.35 (0.62)	10.17 (0.59)	9.88 (0.23)	8.62 (0.07)
10	**15.02 (1.22)**	13.42 (0.89)	13.07 (0.76)	12.43 (0.33)	10.51 (0.08)
15	**16.96 (1.39)**	14.86 (0.96)	14.51 (0.82)	13.59 (0.36)	11.37 (0.08)
20	**19.20 (1.72)**	16.37 (1.18)	15.97 (0.99)	14.36 (0.43)	11.79 (0.09)
25	**20.56 (1.84)**	17.18 (1.23)	16.53 (1.04)	14.73 (0.45)	12.03 (0.09)
30	**21.44 (1.92)**	17.60 (1.26)	17.03 (1.08)	14.98 (0.45)	12.18 (0.09)
$\gamma_0 = 0.85$					
1	5.03 (0.04)	4.96 (0.04)	4.97 (0.04)	**5.20 (0.04)**	4.88 (0.03)
5	**10.29 (0.62)**	9.40 (0.39)	9.13 (0.28)	9.27 (0.19)	8.16 (0.06)
10	**12.91 (0.93)**	11.41 (0.54)	11.08 (0.40)	11.07 (0.26)	9.50 (0.07)
15	**13.89 (1.02)**	12.14 (0.58)	11.80 (0.43)	11.70 (0.27)	9.97 (0.07)
20	**14.87 (1.17)**	12.71 (0.65)	12.31 (0.48)	12.04 (0.30)	10.16 (0.08)
25	**15.15 (1.20)**	12.95 (0.66)	12.48 (0.48)	12.17 (0.30)	10.26 (0.08)
30	**15.37 (1.22)**	13.08 (0.66)	12.59 (0.49)	12.26 (0.30)	10.32 (0.08)
$\gamma_0 = 0.8$					
1	5.03 (0.04)	4.96 (0.04)	4.97 (0.04)	**5.20 (0.04)**	4.88 (0.03)
5	**9.18 (0.45)**	8.71 (0.37)	8.21 (0.11)	8.74 (0.15)	7.76 (0.06)
10	**10.93 (0.68)**	10.15 (0.51)	9.38 (0.14)	10.01 (0.20)	8.71 (0.07)
15	**11.44 (0.73)**	10.55 (0.52)	9.72 (0.15)	10.36 (0.21)	8.98 (0.07)
20	**11.75 (0.77)**	10.76 (0.55)	9.89 (0.17)	10.51 (0.22)	9.07 (0.07)
25	**11.84 (0.78)**	10.83 (0.55)	9.94 (0.17)	10.57 (0.22)	9.11 (0.07)
30	**11.90 (0.79)**	10.88 (0.55)	10.01 (0.17)	10.62 (0.22)	9.15 (0.07)

We report the mean policy value across 20 fully random train-test splits, with standard deviations shown in parentheses. Results are presented with $\gamma_0 \in \{0.9, 0.85, 0.8\}$.

E. Additional Benchmarks

In this section, we build upon the results presented in Section 6.3.2 by incorporating additional benchmark methods for state representation learning.

E.1. Benchmark Policies for Comparison

We develop the following alternative benchmarks to highlight the effectiveness of the proposed multi-response state representation learning method.

- **Myopic policy with Multi-Response State Representation** (`Myopic: MRSR`): We first construct the state representation using the proposed MRSR framework. Then, instead of applying CQL, we train a myopic policy by setting the discount factor to zero in conservative Q-learning and use this representation.
- **CQL with encoder-decoder state representation** (`Encoder-Decoder`): We construct the state representation using a standard encoder-decoder structure that minimizes the reconstruction error of the original customer characteristics. This representation is then applied to conservative Q-learning.

- **CQL with PCA** (`PCA`): We use Principal Component Analysis (PCA) to derive the state representation by selecting the first fifteen principal components, which capture the most variance in the data. This representation is then applied to conservative Q-learning.
- **CQL with only Markov property preservation** (`Markov`): We exclude the short-term response prediction component from state representation learning and directly implement the method proposed by Allen et al. (2021), which focuses exclusively on preserving the Markov property during representation learning. The resulting representation is then used in conservative Q-learning.
- **CQL with dual-response state representation using retention and monetization** (`DRSR: R+M`): This approach builds on MRSR but focuses solely on short-term recency and coin spending for the customer response prediction task. The resulting state representation is then utilized in conservative Q-learning.
- **CQL with dual-response state representation using engagement and monetization** (`DRSR: E+M`): This approach builds on MRSR but focuses solely on short-term engagement intensity and coin spending for the customer response prediction task. The resulting state representation is then utilized in conservative Q-learning.

To ensure a fair comparison, we use the same neural network architecture described in Section 6.3.2. Specifically, we maintain a state representation dimension of 15 and implement a five-layer fully connected neural network for both deep learning-based state representation learning and conservative Q-learning.

E.2. Long-Term-Oriented versus Myopic Policies

Table App-10 compares the proposed method with various myopic policies and the behavioral policy. First, with the same state variables, reinforcement learning-based policies (`MRSR` in column 2 versus `MRSR: Myopic` in column 3 and `Standard CQL` in column 4 versus `Myopic` in column 5) consistently outperform myopic policies in the long term, except on the first day following policy implementation. Second, `MRSR: Myopic` (column 3) achieves significantly higher policy values compared to `Myopic` (column 5) over the long term, highlighting the advantages of MRSR in capturing and summarizing relevant player characteristics. Notably, `MRSR: Myopic` even surpasses the `Standard CQL` policy, further demonstrating the benefits of state representation learning.

Table App-10 Cumulative Coin Spending: Long-Term-Oriented versus Myopic Policies

# Days	MRSR	MRSR: Myopic	Standard CQL	Myopic
1	5.06 (0.04)	**5.25 (0.05)**	5.00 (0.04)	5.20 (0.04)
5	**12.11 (0.82)**	12.09 (0.77)	11.05 (0.64)	10.65 (0.29)
10	**18.09 (1.37)**	17.38 (1.35)	15.61 (1.02)	14.27 (0.42)
15	**21.96 (1.58)**	20.74 (1.67)	18.30 (1.16)	16.40 (0.48)
20	**27.23 (2.21)**	24.60 (2.32)	22.21 (1.79)	18.23 (0.66)
25	**31.23 (2.53)**	27.15 (2.71)	24.16 (2.00)	19.31 (0.71)
30	**35.08 (2.69)**	29.60 (3.11)	25.57 (2.13)	20.21 (0.74)

We report the mean policy value across 20 fully random train-test splits, with standard deviations shown in parentheses. Results are presented with $\gamma_0 = 0.95$.

E.3. Comparison Across Different State Representation Learning Methods

Table App-11 compares the policy values of CQL constructed using different state representations. First, we observe that `MRSR` (column 2) and `SRSR` (column 3) significantly outperform all other methods, highlighting the superior performance of supervised representation learning compared to unsupervised approaches. Second, `Markov` (column 4), `PCA` (column 5), and `Encoder-Decoder` (column 6) consistently outperform `Standard CQL` (column 7) in the long term. This demonstrates the advantages of dimensionality reduction in minimizing noise from the original player characteristics.

Table App-11 Cumulative Coin Spending: Different State Representation Learning Methods

# Days	MRSR	SRSR	Markov	PCA	Encoder-Decoder	Standard CQL
1	**5.06 (0.04)**	4.99 (0.04)	4.98 (0.03)	5.00 (0.03)	3.97 (0.45)	5.00 (0.04)
5	**12.11 (0.82)**	11.66 (0.80)	10.63 (0.21)	11.27 (0.59)	8.89 (1.10)	11.05 (0.64)
10	**18.09 (1.37)**	16.38 (1.23)	15.29 (0.66)	15.61 (0.89)	12.53 (1.58)	15.61 (1.02)
15	**21.96 (1.58)**	19.16 (1.38)	18.10 (0.84)	18.22 (1.00)	15.52 (2.04)	18.30 (1.16)
20	**27.23 (2.21)**	23.25 (1.98)	22.18 (1.99)	22.54 (1.54)	21.21 (3.33)	22.21 (1.79)
25	**31.23 (2.53)**	26.47 (2.28)	24.42 (2.32)	24.66 (1.68)	23.99 (3.87)	24.16 (2.00)
30	**35.08 (2.69)**	28.38 (2.51)	26.59 (2.65)	26.48 (1.81)	27.09 (4.50)	25.57 (2.13)

We report the mean policy value across 20 fully random train-test splits, with standard deviations shown in parentheses. Results are presented with $\gamma_0 = 0.95$.

E.4. Comparison Across Different Supervision Customer Responses in State Representation Learning

Table App-12 further compares CQL policies using different supervised state representation learning methods with varying supervision customer responses. The results show that `MRSR` (column 2) significantly outperforms both `DRSR: E+M` (column 3) and `DRSR: R+M` (column 4) in the long term. Additionally, both `DRSR: E+M` (column 3) and `DRSR: R+M` (column 4) outperform `SRSR` (column 5) and the standard `Standard CQL` approach (column 6) in the long term. These results highlight the importance of incorporating insights from retention or engagement when learning customer relationship states. Furthermore,

retention and engagement intensity provide distinct information, each contributing uniquely to understanding customer long-term value.

Table App-12 Cumulative Coin Spending: Different Customer Responses in State Representation Learning

# Days	MRSR	DRSR: E+M	DRSR: R+M	SRSR	Standard CQL
1	5.06 (0.04)	5.06 (0.04)	5.01 (0.03)	4.99 (0.04)	5.00 (0.04)
5	12.11 (0.82)	12.47 (0.95)	11.65 (0.56)	11.66 (0.80)	9.20 (0.07)
10	18.09 (1.37)	17.51 (1.36)	17.23 (0.99)	16.38 (1.23)	11.05 (0.64)
15	21.96 (1.58)	20.59 (1.52)	20.82 (1.30)	19.16 (1.38)	18.30 (1.16)
20	27.23 (2.21)	26.58 (2.55)	27.38 (2.29)	23.25 (1.98)	22.21 (1.79)
25	31.23 (2.53)	30.02 (2.84)	30.30 (2.61)	26.47 (2.28)	24.16 (2.00)
30	35.08 (2.69)	32.74 (3.04)	33.47 (3.00)	28.38 (2.51)	25.57 (2.13)

We report the mean policy value across 20 fully random train-test splits, with standard deviations shown in parentheses. Results are presented with $\gamma_0 = 0.95$.

F. Details for Policy Explanation

F.1. Summary Statistics of Latent States

Table App-13 expands on Table 4 in the main text by providing a comprehensive summary of all player characteristics associated with the latent states identified using both multi-response and single-response state representation methods.

F.2. Construction and Summary of the Hypothetical "Median" Player

In Section 7.2, we construct a hypothetical "median" player to illustrate how different DDA policies adjust difficulty for a representative player as certain characteristics vary. Specifically, we use the median values from Table App-2, with two exceptions: the number of rounds played last week and day-of-week variables. For these variables, the following rules are applied:

- # Rounds Played Last Week: We use the median value from observations where the number of rounds played is less than 40, resulting in an empirical median of 15. This restriction reflects the company's policy of adjusting difficulty levels only for players with fewer than 40 rounds played in the previous week.
- Day of the Week Variables ("Is Monday?", "Is Tuesday?", ..., "Is Sunday?"): Since these are binary variables with median values of zero, we calculate the average expected difficulty and difficulty distribution by iteratively setting each variable to one (representing the selected day) while keeping the others at zero and holding all other variables constant.

Table App-13 Full Summary of Player Latent States

	Multi-Response Representation			Single-Response Representation			
Representative Player Characteristics	**Segment 1**	**Segment 2**	**Segment 3**	**Segment 1**	**Segment 2**	**Segment 2-1**	**Segment 2-2**
# active days since start	18.73	28.45	34.60	18.61	32.32	59.49	19.16
# days since start	70.56	57.55	55.90	70.65	56.51	88.36	41.09
# rounds played since start	183.99	313.22	432.66	182.35	388.23	764.95	205.85
# coins used since start	108.48	146.76	187.89	95.06	173.91	177.80	172.04
Winning rate since start	45.58%	41.76%	39.86%	45.61%	40.56%	33.79%	43.85%
Avg. stars per round since start	1.23	1.18	1.13	1.23	1.15	1.03	1.21
Avg. points per round since start	32,802	34,373	35,375	32,782	35,003	36,611	34,224
Avg. snake length per round since start	5.11	5.14	5.17	5.11	5.16	5.20	5.14
Max level ever reached	38.57	48.05	54.95	38.18	52.41	71.66	43.09
# coins in storage	18.02	23.89	23.47	16.30	23.80	18.04	26.60
# days since last gameplay	19.29	3.23	1.02	19.30	1.84	1.64	1.94
Is Monday?	10.92%	15.26%	13.90%	10.94%	14.40%	12.98%	15.09%
Is Tuesday?	11.13%	13.54%	15.10%	11.16%	14.51%	14.47%	14.53%
Is Wednesday?	14.78%	14.13%	14.85%	14.81%	14.58%	13.86%	14.93%
Is Thursday?	12.85%	12.48%	13.90%	12.66%	13.39%	13.93%	13.13%
Is Friday?	17.56%	13.96%	13.83%	17.81%	13.85%	14.74%	13.42%
Is Saturday?	15.85%	15.43%	13.19%	15.67%	14.05%	14.81%	13.68%
Is Sunday?	16.92%	15.20%	15.24%	16.95%	15.22%	15.21%	15.22%
Is active yesterday?	0.00%	2.25%	99.40%	0.00%	63.15%	69.64%	60.00%
# rounds played yesterday (if active)	–	4.18	10.74	–	10.70	9.67	11.20
Propensity of using coins yesterday (if active)	–	0.00%	3.96%	–	4.04%	1.46%	5.29%
# coins used yesterday (if active)	–	0.00	3.76	–	3.86	0.93	5.28
Winning rate yesterday (if active)	–	14.25%	26.03%	–	26.30%	17.78%	30.42%
Avg. stars per round yesterday (if active)	–	0.72	0.93	–	0.93	0.84	0.98
Avg. points per round yesterday (if active)	–	9,867	37,612	–	37,128	39,458	36,000
Avg. snake length per round yesterday (if active)	–	3.98	5.16	–	5.14	5.24	5.09
Is active last week?	0.64%	99.94%	100.00%	0.86%	99.93%	99.93%	99.93%
# rounds played last week (if active)	1.00	22.56	51.56	2.00	40.75	41.77	40.26
Propensity of using coins last week (if active)	0.00%	9.35%	17.07%	0.00%	14.19%	8.19%	17.10%
# coins used last week (if active)	0.00	11.63	21.75	0.00	17.98	6.86	23.36
Winning rate last week (if active)	0.00%	31.66%	30.76%	0.00%	31.10%	21.16%	35.92%
Avg. stars per round last week (if active)	0.00	1.02	1.00	0.25	1.01	0.90	1.06
Avg. points per round last week (if active)	0	36,507	38,377	6,065	37,682	39,849	36,633
Avg. snake length per round last week (if active)	3.00	5.12	5.19	3.40	5.16	5.24	5.12
Responses (Under Difficulty level 5)							
# days until next play	13.66	5.26	2.73	13.64	3.54	2.38	4.11
# rounds played in the day	4.56	6.85	10.66	4.54	9.45	9.44	9.45
# coins used in the day	1.15	2.26	3.60	1.14	3.19	1.08	4.21
Segment Size	9.34%	33.82%	56.84%	9.32%	90.68%	29.58%	61.10%

The full summary of within-cluster average player characteristic values is presented in the table. Segment 2 of single-response representation is a combination of Segment 2-1 and Segment 2-2, as is shown in Figure 4.

References

Allen C, Parikh N, Gottesman O, Konidaris G (2021) Learning markov state abstractions for deep reinforcement learning. *Advances in Neural Information Processing Systems* 34:8229–8241.

Kingma DP (2014) Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Kumar A, Zhou A, Tucker G, Levine S (2020) Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems* 33:1179–1191.