Customer Acquisition via Explainable Deep Reinforcement Learning

Yicheng Song
Carlson School of Management, University of Minnesota, Minneapolis, MN 55455, ycsong@umn.edu

Wenbo Wang
HKUST Business School, Clear Water Bay, Kowloon, Hong Kong, wenbowang@ust.hk

Song Yao
Olin Business School, Washington University in St. Louis, St. Louis, MO 63130, songyao@wustl.edu

Abstract: Effective customer acquisition heavily hinges on sequential targeting to ensure appropriate marketing messages reach customers. Sequential targeting could guide customers through the acquisition process and thus optimize long-term revenue for the firm. Towards this goal, Reinforcement Learning (RL) has demonstrated great potential in facilitating sequential targeting during user acquisition. However, decisions made by RL during this process often lack explainability. We introduce the DRQN-Attention model, which optimizes the long-term reward of sequential targeting while enhancing the explainability of the decisions. The key idea of the proposed model is to revise Q-Learning by adding an attention mechanism to create a bottleneck, forcing the model to focus on features of the next ad exposure that will lead to optimal long-term rewards. We estimate our model using a comprehensive dataset from a digital bank. The empirical results show the proposed model is explainable and also outperforms state-of-the-art methods in terms of long-term revenue optimization. Specifically, the attention mechanism within the model functions as forward planning. The forward planning can spot those features in the next ad exposure that are more likely to lead to the optimal outcome. We further demonstrate how the model makes targeting decisions of advertising channel choices by showing that the model can: 1) learn optimal ad channels to target customers from different industries, 2) adjust advertising channels in response to dynamic customer behaviors, and 3) learn the seasonality of the customer's industry and calibrate ad channel correspondingly.

Key words: Explainable Reinforcement Learning, Customer Acquisition, DRQN-Attention, Long-term revenue optimization, Advertising channel choice

1. Introduction

Successful customer acquisition stands as a pivotal factor in the growth of businesses in the digital era. This process involves identifying potential customers and guiding them through the acquisition funnel (Song et al. 2022). Companies gauge the effectiveness of their customer acquisition strategies by assessing the

cost and efficiency of acquisition, considering them as key metrics (Chuck 2022). Notably, the customer acquisition cost on many e-commerce platforms is exceptionally high, emphasizing the critical importance of ensuring that these costs do not exceed the average lifetime value of acquired customers (Hoffman and Novak 2000). The significance of customer acquisition costs persists, even in emerging industries that leverage cutting-edge technologies such as Artificial Intelligence and Cloud Computing (AppsFlyer 2021).

The randomized experiment is a popular choice to evaluate the effects of various intervention policies for customer acquisition, including targeting different groups of customers (Frick et al. 2022), and using different ad formats and channels when deploying the ad (Reiley et al. 2012). However, these experiments often focus on examining the impact of each individual intervention, neglecting the potential interplay among them. This might lead to sub-optimal results (Song and Sun 2023), especially for the settings that need to consider a sequence of interventions such as sequential targeting. Furthermore, when the goal of an experiment is to find the optimal sequence of many interventions, the policy space grows exponentially, making it challenging to overcome the "curse of dimensionality" and identify the optimal policy using experiments. Fortunately, as suggested by the causal decision-making literature (Fernández-Loría and Provost 2022), an accurate causal effect estimation of each intervention via randomized experiments is often unnecessary for determining which interventions to use and their sequence so as to optimize the outcomes. Instead, the optimization only requires differentiating the effectiveness of different interventions under different contexts. Therefore, as an alternative to randomized experiments, a model-based approach aiming to achieve desired business outcomes could be a promising solution to the customer acquisition problem.

Machine learning models have become integral for developing cost-effective customer acquisition strategies. Commonly, existing machine learning-based targeting techniques leverage supervised learning models to predict the profitability of prospects under different individual marketing interventions (Simester et al. 2020). However, the customer's decision to purchase a product or service often entails a significant commitment of time and money, making the conversion less likely to be triggered by a single intervention. Recognizing this, prior work (Schwartz et al. 2017) frames customer acquisition as a challenge of converting prospects through a series of marketing messages sequentially across diverse contexts (e.g., different devices

and websites). Allocating these sequential marketing messages presents a challenge, as the responses of prospective customers to different messages are either unknown or only partially revealed. Understanding these responses improves over interactions, leading to a "learn-and-earn" or "exploration-and-exploitation" trade-off. Firms must strategically deploy various marketing messages, not only leveraging known responses for stable revenue but also exploring new messages with uncertain yet potentially more profitable outcomes. To address this objective, researchers have turned to Reinforcement Learning (RL), a framework designed to allocate a limited budget in a way that balances exploration and exploitation.

RL models have shown promise in addressing various business challenges, such as optimizing the ad click-through rate (Hauser et al. 2009), designing sequential promotions (Wang et al. 2022b), and guiding subject allocations for future trials based on past randomized experiments (Song and Sun 2023). These applications of RL demonstrate its versatility and effectiveness in making optimal decisions in complex business environments. Despite the successes, most RL models developed to facilitate business decision-making are black-box machine learning models without transparency, which leaves users of such models with a limited understanding of why particular decisions are made. However, it is crucial for RL models adopted by firms to be transparent and easily understandable pertaining to how decisions are made, what information is used, and why mistakes occur (Puiutta and Veith 2020). Otherwise, the lack of explainability can frustrate and confuse users and diminish their trust in intelligent systems, hindering the future applications of these models (Zhang and Curley 2018). Consequently, there has been a growing demand for "explainable" models. For example, DARPA started the "Explainable Artificial Intelligence" program in 2018 to create explainable high-performing models and to convey the reasoning process of machines that humans may understand (Turek 2018). Recently, Information Systems researchers also called for empowering machine learning models with explainability to address the lack of transparency in deep learning (Berente et al. 2021). Note that such explainability does not necessarily need to be clean causal effects identified from field experiments. Instead, it can be insights that enable researchers to glimpse into the decision-making process via the outputs of the machine learning models.

In this study, we present the Deep Recurrent Q-Network with Attention model (DRQN-Attention), designed to enhance the explainability of RL models for customer acquisition while preserving the primary

objective of optimizing long-term reward.[1] The model aims to to expose the right information to the right prospect in the right context. It is the prospects themselves who decide whether to apply for the service or buy the product. The model adopts the Deep Recurrent Q-Network (DRQN) (Hausknecht and Stone 2015) to model the Partially Observable Markov Decision Process (POMDP) of prospects, where customer states are only partially observed and cannot be fully represented by a few recent observations. To create a bottleneck in the RL agent and compel the model to focus on task-relevant information, the proposed model integrates the Q-Learning algorithm with attention mechanism (Vaswani et al. 2017). The customized attention mechanism allows direct observation of the information used by the model for decision-making, rendering the model more explainable than conventional RL models.

We review the related literature in Section 2 to identify the research gap and summarize the contributions of this study. Following that, we introduce the empirical context of this study in Section 3. In Section 4, we develop the DRQN-Attention that integrates a customized attention mechanism within DRQN. The evaluation of the DRQN-Attention model on a large dataset collected from a digital bank is presented in Section 5, where we demonstrate the model's superiority over state-of-the-art methods for long-term revenue optimization. To address privacy concerns, we adopt federated learning for the RL models in Section 5.3 and showcase that the model can be applied in privacy-sensitive settings without compromising service quality. In Section 6, we delve into gaining more insights into the attention mechanism results, demonstrating that the attention mechanism functions as forward planning. This forward planning identifies which features of the next ad exposure opportunity are more likely to lead to optimal outcomes. Further insights into the attention mechanism are explored in Section 7, where we illustrate that the attention mechanism helps managers better understand how the model chooses online advertising channels. This includes optimizing advertising channels for prospects from different industries, adjusting advertising channels to dynamic prospect behaviors, and developing sequential advertising channel plans that account for the seasonality of prospects from the agricultural industry.

[1] Throughout the paper, we use the terms "reward" and "revenue" interchangeably, as they convey the same meaning in our context.

2. Literature

Our research closely relates to two streams of literature. The first is the studies that develop RL models to facilitate business decision-making and problem-solving. The second stream is explainable RL, which is methodologically related to this study.

2.1. Reinforcement Learning For Business Decision-Making

RL has the potential to be a powerful tool for optimizing business goals by allocating limited resources effectively. In marketing, RL can be used to learn dynamic and adaptive marketing strategies by continuously learning from past experiences and fine-tuning the marketing messages to achieve optimal outcomes. With its ability to balance exploitation and exploration of marketing messages, RL can help companies drive better outcomes. Multi-Armed Bandit (MAB) is the first RL model adopted by researchers that optimizes the allocation of limited resources to facilitate marketing strategies (Hauser et al. 2009, 2014). While MAB can help firms optimize outcomes such as Click-Through Rate (CTR), a major limitation of such models is that they mainly focus on the immediate feedback from the prospects (e.g., ad click) but do not explicitly model long-term revenues.[2] However, ad click is just an intermediary step toward customer acquisition and firms are more interested in optimizing long-term revenues that consider both intermediary steps and the final goal (e.g. customer acquisition). Recognizing the importance of optimizing long-term revenues, new RL models have been developed. Building on the Q-Learning framework (Watkins and Dayan 1992), Wang et al. (2022b) propose a deep RL model for sequential targeting problems. This model first builds a predictive model to capture customers' responses to various marketing actions and then trains a deep RL agent to interact with the predictive model. The goal is to learn optimal sequential marketing strategies, considering the dynamic sequential behavior of customers to optimize long-term revenues. In a similar vein, Song and Sun (2023) develop an RL algorithm within the Bayesian framework. This algorithm learns the returns to sequential intervention strategies from historical randomized experiments, guiding subject allocation in future experiments to further improve intervention strategies.

[2] MAB is an RL model with a single state, where the following observation revealed by the environment to the agent is not influenced by the preceding actions (Sutton and Barto 2018).

From the perspective of model explainability, current approaches in this field mostly attempt to explain model decisions through post-hoc analysis. Specifically, they normally create a black-box RL model and then utilize post-hoc analysis to demonstrate its explainability, showing the variation of model outputs can be traced back to changes in model inputs. However, little advance has been made for intrinsic explainable RL models, especially for the settings of customer acquisition. Unlike post-hoc explanations, the intrinsic explainability of RL is achieved by designing models that are self-explanatory and incorporate explainability directly into their structures.[3] These explainable models are either globally explainable or can provide explanations for individual prediction, thereby offering accurate and undistorted explanations of model decisions (Du et al. 2019). In sum, we propose an intrinsic explainable RL model that provides explainability for customer acquisition, addressing the gap in the literature and meeting the needs of businesses.

2.2. Explainable Reinforcement Learning

RL has been successfully applied to a variety of real-world scenarios, including autonomous vehicles (Kiran et al. 2021), traffic management (Zhou et al. 2020), career planning (Kokkodis and Ipeirotis 2020), and other domains that require sequential decision-making. However, making RL models explainable is challenging due to two main factors. Firstly, RL models are often built using black-box machine learning models, inherently lacking explainability. Secondly, the design of RL models for sequential decision-making and long-term revenue optimization inevitably makes them difficult to explain. A recent survey by Milani et al. (2022) categorizes the state-of-the-art explainable RL models into three classes: 1) Feature importance explanations identify the features that affect an agent's action given the input state. 2) Learning process explanations show how past experiences led to the current decision. 3) Policy-level explanations illustrate the long-term behavior of the agent. Based on such schema, our study belongs to the 2nd category that aims to use attention mechanisms to identify important features in the incoming interaction that affect an agent's action choices to optimize long-term revenue.

Attention mechanisms have gained popularity in deep learning for several reasons (Brauwers and Frasincar 2021). Most attention mechanisms can be trained jointly with a base neural network model using

[3] This includes models such as decision trees, linear models, and attention models, among others

regular backpropagation. Additionally, attention introduces a form of explanation into neural network models, which are generally known for being highly complex and challenging to explain. Recent advancements in state-of-the-art explainable RL models have therefore incorporated attention mechanisms. As per the definition of the general attention model in Brauwers and Frasincar (2021), there are three major components in attention mechanisms: query q, key K, and value V. The query, designed based on the desired output of the model, instructs the attention model on which features to focus. It serves as a request for information or a question. Two separate matrices, keys matrix K and values matrix V, are then generated from the input features. The attention module's goal is to produce a weighted average of the value vectors in V, weighted by the attention weight a. The attention weight a results from the interaction between the query q and the keys matrix K through different attention score functions. The hypothesis behind modeling attention is that the magnitude of attention weight a correlates with the relevance of a specific input region to the model prediction. Visualizing attention weight a for a set of input and output pairs provides insights into the importance of different inputs for model decisions. The general attention model is versatile and can be applied to various problems. However, researchers need to tailor the attention mechanism (i.e., query q, key K, and value V) to address specific research questions in different contexts (Chaudhari et al. 2021). Next, we will introduce representative RL models that incorporate attention mechanisms and discuss the differences between their attention models and the one proposed in this study.

Many explainable RL models leveraging attention mechanisms were developed in the context of video games, where the zero-cost game stimulator ensures efficient RL model training, and well-structured game rules facilitate feasible and widely accepted model decision interpretation (Torrado et al. 2018, Shao et al. 2019). Specifically, Annasamy and Sycara (2019) integrate an attention mechanism with the Deep Q-network, which provides explanations for how an intelligent agent makes decisions when playing video games. To provide an understanding of the learned gameplay policy, the authors visualize clusters of state to represent attention maps for any action-value pair. Related, Mott et al. (2019) introduce a spatial attention mechanism to visualize information in an actor-critic RL model. By exploiting the attention maps on the spatial basis, one can understand how the RL model solves a task in the video game, identifying the type of

entities that the agent attends ("what") and spatial locations where the agent directs its attention ("where"). Shi et al. (2020) adopt a self-supervised explainable network to produce fine-grained attention masks to highlight task-relevant information in the game that most likely leads to the agent's decisions. Itaya et al. (2021) integrate an attention mechanism into the Asynchronous Advantage Actor-Critic model to analyze the decision-making of an intelligent agent. The authors introduce attention mechanisms into both the policy and value branches of the Actor-Critic model, using attention to reveal the reasons behind the two branches.

In line with these studies, we have developed an explainable RL model for customer acquisition by incorporating a customized attention mechanism. But different from these models developed in the context of video game control, the proposed model takes into account the nuances of customer modeling, rendering the customized attention mechanism different from previous works in several key dimensions:

1. For studies using video game play as the context, the latest video image frame generally contains complete information about the environment (e.g. Go Game, Atari game). Therefore, most models utilize the latest video image frame to construct query, key, and value in the attention mechanism. However, in the context of customer acquisition, the potential customer's state can rarely be described only by the latest observation. Accordingly, we use Recurrent Neural Network (RNN) to process potential customers' historical interactions, along with the static features, to construct the query.

2. Because existing studies use video game image frames to construct query, key, and value in the attention mechanism, they apply the attention weight to the feature map generated by a Convolutional Neural Network (CNN), and then use a deconvolution function to restore the attention weight back to the original image space. However, in the context of customer acquisition, there is no reverse engineering function when using RNN to process historical interactions with the customer. Hence we apply the attention map directly to the raw incoming interaction to highlight important features.

3. Different from just utilizing video game frames (i.e same source), we rely on different sources to construct the query, key, and value in the attention mechanism. The historical interaction and static information underpin the query, and the incoming interaction generates key and value. The rationale is that given static information and historical interactions of a prospect are fixed and immutable, the model's focus is directed

toward recognizing incoming interactions for long-turn reward optimization. This step allows the model to dynamically adjust the attention weight based on different historical interactions and static information, discovering features in the incoming interaction that lead to more promising outcomes.

Table 1 Attention Mechanism Components (Query, Key, Value) Comparison

	Setting	Query	Key	Value
Annasamy et al. (2019)	Video Game	CNN on video frames	Action specific query, randomly generated	Action specific value, randomly generated
Mott et al. (2019)	Video Game	ConvLSTM on game frames	CNN on game frames+spatial basis	CNN on game frames+spatial basis
Shi et al. (2020)	Video Game	Attention weight from auto-encoder model	Attention weight from auto-encoder model	Latest raw game frames
Itaya et al. (2021)	Video Game	CNN on video frames	CNN on video frames	CNN on video frames
Zhang et al. (2021)	Medical	MRI	MRI	MRI
Fei et al. (2021)	NLP	Word Embedding	Word Embedding	Word Embedding
Wang et al. (2019)	Stock Trading	Stock Representation	Stock Representation	Stock Representation
This Study	Customer Targeting	Historical Interaction +Static Info	NN on Incoming Interaction	Raw Incoming Interaction

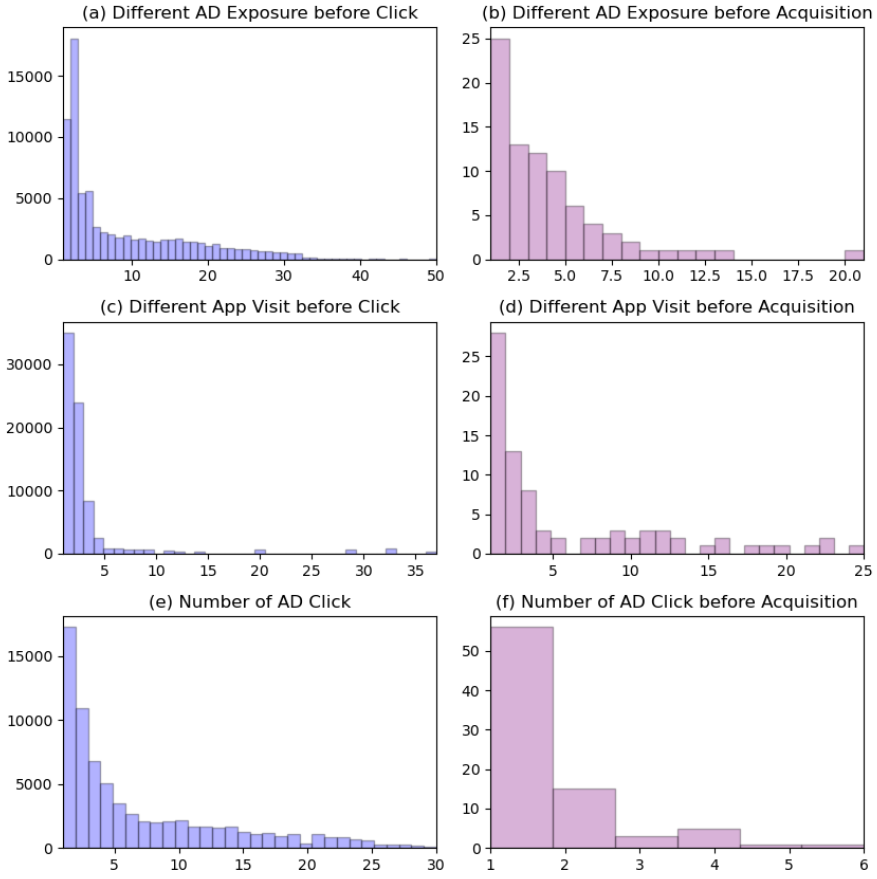
Note: CNN: Convolutional Neural Network, ConvLSTM: Convolutional LSTM Network, NN: Neural Network

While many works on explainable RL with attention mechanisms are developed in the context of video games to facilitate model training and decision interpretation, there are related works have explored other domains. For instance, Zhang et al. (2021) propose an interpretable deep reinforcement learning model for diagnosing Alzheimer's disease. They apply the attention mechanism to MRI images and clinical information, thereby explaining parameter weights for Alzheimer's disease diagnosis. Fei et al. (2021) develop an explainable RL model to automatically optimize attention distribution on sequential textual data to minimize natural language processing (NLP) task training losses. Their results demonstrate that the model can yield more reasonable attention distribution for NLP tasks. Additionally, Wang et al. (2019) focus on the quantitative stock trading problem, aiming to enhance investment strategy through an interpretable RL model. They customize the attention mechanism to capture the interrelationship among stocks, enriching investment strategy with additional information. These context-specific studies demonstrate how attention mechanisms can be customized to fit specific research questions. Table 1 summarizes the detailed differences in attention mechanisms across these studies.

In conclusion, the uniqueness of the customer acquisition problem that allows us to design a customized attention mechanism specifically for this task. To the best of our knowledge, there is no existing intrinsic explainable RL model specifically designed for customer acquisition problems. Our proposed model addresses two critical challenges: 1) it provides a general RL model for customer acquisition that can help companies optimize their long-term revenues, and 2) it enables managers to gain valuable insights into the decision-making processes of the model. The intrinsic explainability of our proposed model sets it apart from other existing models, allowing managers to understand informed decisions based on the results generated by the model.

3. Research Context

Figure 1 Potential Customer Behaviors Before Ad Click and Customer Acquisition.



Note: 1) (a) and (b) suggest that prospects often receive multiple different ads before clicking and applying for credit (customer acquisition). 2) (c) and (d) indicate that prospects are active in multiple advertising channels. 3) While there are many ad clicks in (e) but only a few of them lead to customer acquisitions in (f), and many acquisitions are preceded by multiple ad clicks.

We partnered with a digital bank in China for this research. Our partner provides online financial services to underbanked and unbanked Small and Medium-sized Enterprises (SMEs) in China, a market that is highly overlooked by traditional banks. As highlighted in the World Bank report (Bank 2023), SMEs play a vital role in most economies, particularly in developing countries. They constitute a significant portion of businesses globally and make substantial contributions to job creation and global economic growth. Nevertheless, access to financial services remains a significant constraint for SMEs and ranks as the second most frequently cited obstacle hindering SMEs' growth. The International Finance Corporation estimates that a staggering 65 million firms, equivalent to 40% of SMEs in developing nations, face an unmet financing gap of $5.2 trillion annually, which is 1.4 times greater than the current global lending level to SMEs (Corporation 2017). Therefore, the development of an efficient customer acquisition model that accurately targets potential SMEs clients and delivers essential financial information becomes imperative.

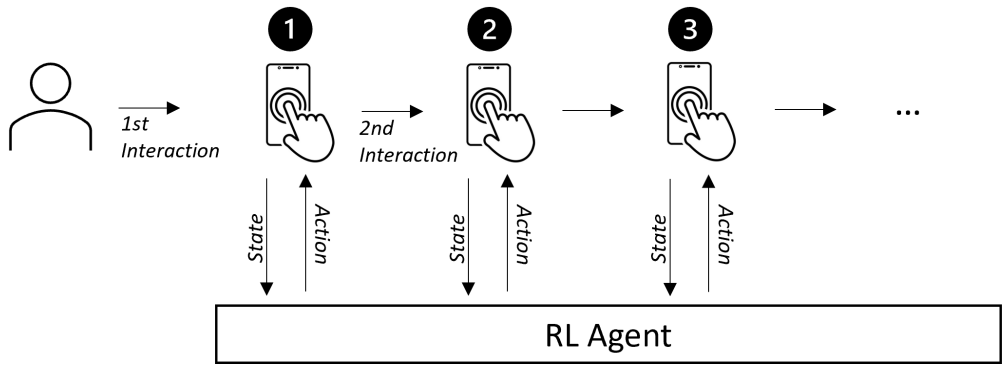
Our partner's advertising channels encompass numerous popular third-party mobile apps, with the customer acquisition team aiming to identify prospective borrowers by delivering various advertising messages across these channels. Their ad library comprises nearly two hundred different ad messages, each highlighting different aspects of their product, such as speedy loan approval within 15 minutes, daily interest rates as low as 0.01%, and loans specialized for small and microbusinesses, among others. It's essential to underscore that their loan service for SMEs is a standard public service, treating all individuals equally without differential treatment, such as offering higher interest rates based on personal attributes. Additionally, the service is accessible to anyone, with prospects making their own decisions about whether to apply for the service. Their existing customer acquisition system is developed based on contextual Multi-Armed Bandit (MAB) (Lu et al. 2010) to optimize the Click-Through Rate (CTR). When a potential customer visits one of the advertising channels, an ad exposure request is sent to the bank. The customer acquisition system then responds with decisions on whether to show an ad to the prospect by evaluating the cost and expected gain, and if affirmative, which ad to display. If an ad is shown, the prospect can click on it to obtain more information about the service and potentially apply for credit, marking a successful customer acquisition.

We collected clickstream data from a random sample of the bank's potential customers. Examining the activities of these potential customers reveals substantial heterogeneity in their responses to ads and paths

to customer acquisition (i.e., applying for credit). Figure 1 (a) and (b) suggest that potential customers often receive multiple different ads before they click and apply for credit, implying the combinations of different ads help move the prospects through the customer acquisition funnel. In addition, potential customers are active in multiple advertising channels as they have been receiving ads on different Apps (Figure 1 (c) and (d)), indicating the necessity to target them across channels. Finally, even though we observe many ad clicks (Figure 1 (e)), only a handful of the clicks lead to customer acquisitions (Figure 1 (f)). Moreover, as 32% (≈26/81) of acquisitions are preceded with more than one click, indicating CTR alone may not be a good metric and the targeting should be optimized by comprehensively considering both the final outcome (acquisition) and the intermediate steps (ad click). These observations reinforce our motivation to develop an intelligent model that meets the needs of cross-channel prospect targeting to optimize long-term revenue.

4. Model

Figure 2 Customer Acquisition via RL-enabled Sequential Targeting



Note: 1) State: ad exposure request that summarizes the state of the customer at the current interaction. 2) Action: whether to show an ad and, if yes, which ad to show.

We frame customer acquisition as an RL problem, where an intelligent RL agent interacts with potential customers by deploying a sequence of ad messages over time. As illustrated in Figure 2, whenever a potential customer interacts with a collaborated channel,[4] an ad exposure request that summarizes the state of the prospect is sent to the RL agent, and the agent will return with a decision of whether to show an ad, and if yes, which ad to show. The RL agent's objective is to optimize long-term rewards from prospects. Next, we formalize the three key elements (S, A, R) of the RL model in this setting, including:

[4] Prospect's arrival on the collaborated channel is an exogenous process.

- State S: A state $\boldsymbol{s_{it}} \in S$ summarizes the state of prospect i at the tth interaction, which consists of her interaction history (i.e., interactions $1,..,(t-1)$) and the contextual information of the current tth interaction (e.g. website, APP, time, ...). Once we observe the prospect's subsequent interaction (i.e. $t+1$), the prospect's state will transit to $s_{i(t+1)}$.
- Action A: The action space A is a collection of ads, as well as the option of not showing an ad ("no ad"). Accordingly, action in the tth interaction with prospect i, $a_{it} \in A$, is showing a specific ad or "no ad".
- Reward R: If the action a_{it} shows an ad, given the prospect's state $\boldsymbol{s_{it}}$, the intelligence agent will receive an immediate reward $r_{it} \in R$ based on the prospect's feedback. The prospect might click the ad, and further register as a customer (customer acquisition), which leads to positive rewards.[5] Otherwise, a negative reward is incurred due to the cost of displaying the ad. If the action is "no ad", r_{it} will be 0.

With the outlined components, we can formally define customer acquisition within the RL framework. When observing the state $\boldsymbol{s_{it}}$ of prospect i, RL aims to learn a value function $\mathbb{F}(\boldsymbol{s_{it}}, a_{it})$ that estimates the expected cumulative reward by applying action a_{it} under the state $\boldsymbol{s_{it}}$. This estimation goes beyond the immediate reward r_{it} and encompasses rewards in the future. Consequently, the RL model is inclined to select action a_{it} that leads to the optimal value of $\mathbb{F}(\boldsymbol{s_{it}}, a_{it})$, indicating a promising outcome. Next, we will explore various options of value function $\mathbb{F}(\boldsymbol{s_{it}}, a_{it})$. It is noteworthy that the RL framework described provides a general solution to customer acquisition problems, irrespective of specific contexts.

4.1. Deep Recurrent Q-Network

Q-Learning is a classical RL algorithm that estimates the cumulative long-term expected reward of executing an action for a given state (Watkins and Dayan 1992). Such estimated long-term rewards are known as Q-values. A higher Q-value $Q(\boldsymbol{s_{it}}, a_{it})$ indicates an action a_{it} is deemed to yield a better long-term reward given the state $\boldsymbol{s_{it}}$. For an episode containing an ordered trajectory of state, action, and reward $\{\boldsymbol{s_{it}}, a_{it}, r_{it}, s_{i(t+1)}\}$ obtained by executing the action a_{it} at the tth interaction, Q-value $Q(\boldsymbol{s_{it}}, a_{it})$ is defined as observed immediate reward r_{it} plus the max Q-value across all actions in the next state $s_{i(t+1)}$:

$$Q(\boldsymbol{s_{it}}, a_{it}) = r_{it} + \gamma \max_{a_{i(t+1)}} Q\left(\boldsymbol{s_{i(t+1)}}, a_{i(t+1)}\right) \tag{1}$$

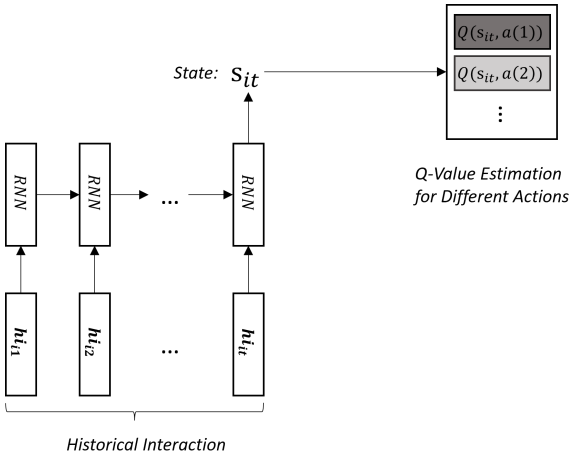
[5] The reward of customer acquisition should be larger than the reward of ad click. We will specify the reward setting in Section 5.1.

where $\gamma \in [0,1]$ is the discount factor that balances current and future rewards. Thus, the goal of the RL agent is to learn policy $S \rightarrow A$ which chooses the action a_t that leads to the optimal Q-value in each state. In many real-world settings, the space that describes the state and action is often too large to estimate Q values for all $|S| \times |A|$ pairs. Accordingly, as a general and flexible model, the deep neural network can be used to learn the complex state representation. Therefore, combining Q-Learning with a deep neural network (i.e., Deep Q-Network or DQN) can be used to accommodate complex inputs in real-world settings and learn actions to optimize the long-term reward. Such a model has been adopted by researchers to optimize sequential promotion design (Wang et al. 2022b). Specifically, they model DQN as a neural network with parameters θ, Q-values can be estimated using the function $Q(\boldsymbol{s_{it}}, a_{it}|\theta)$. To train such a neural network, the parameters θ are chosen to minimize the temporal difference via the following loss function:

$$\mathcal{L}(S,A,R|\theta) = \sum_{i,t}(Q(\boldsymbol{s_{it}}, a_{it}|\theta) - (r_{it} + \gamma \max_{a_{i(t+1)}} Q(\boldsymbol{s_{i(t+1)}}, a_{i(t+1)}|\theta)))^2 \tag{2}$$

One underlying assumption of DQN is the Markov Decision Process (MDP), where the future state is based solely on the most recent action and state: $P(s_{i(t+1)}|\boldsymbol{s_{it}}, a_{it}, ..., \boldsymbol{s_{i1}}, a_{i1}) = P(\boldsymbol{s_{i(t+1)}}|\boldsymbol{s_{it}}, a_{it})$. Such a Markov property, however, rarely holds in the real world, especially because a prospect's state can rarely be fully described by only the most recent observations. As an alternative to MDP, the Partially Observable Markov Decision Process (POMDP) better describes many real-world environments by acknowledging that the observation received by the agent is only a partial glimpse of the state. To demonstrate the advantage and feasibility of POMDP, Hausknecht and Stone (2015) propose Deep Recurrent Q-Network (DRQN), a combination of Recurrent Neural Network (RNN) and Deep Q-Learning. Different from DQN only uses the most recent video game frames to summarize the state of the environment, DRQN utilizes RNN to process sequential video game frames so as to summarize the state more comprehensively (Hausknecht and Stone 2015). Similarly, the state of a prospect is also a POMDP as it cannot be summarized by only the most recent observations. Thus, Song and Sun (2023) also utilize DRQN to process the interaction history of the prospect to summarize the state. Following these prior studies, we can adopt DRQN to learn the optimal customer targeting policies.

Figure 3 Framework of Deep Recurrent Q-Network (DRQN)



Note: Historical Interaction data $\{\boldsymbol{hi_{i1}}, \boldsymbol{hi_{i2}}, ..., \boldsymbol{hi_{it}}\}$ associated with prospect *i* till the *t*th interaction will be processed by an RNN to get the prospect state representation $\boldsymbol{s_{it}}$. We then predict the Q-values of excusing different actions based on state $\boldsymbol{s_{it}}$.

Figure 3 outlines the framework for applying DRQN to our setting. RNN is used to process historical interactions, where $\boldsymbol{hi_{it}}$ describes the agent's *t*th interaction with prospect *i* that consists of the contextual features of the interaction (including time, device, type of location, app, prospect reaction (click or no click), etc).[6] With interaction sequence $\{\boldsymbol{hi_{i1}}, ..., \boldsymbol{hi_{it}}\}$ associated with prospect *i*, a Gated Recurrent Unit (GRU) is utilized to process interaction sequence to learn the state $\boldsymbol{s_{it}}$:

$$\boldsymbol{s_{it}} = GRU(\boldsymbol{hi_{i1}}, \boldsymbol{hi_{i2}}, ..., \boldsymbol{hi_{it}}) \tag{3}$$

The detailed mechanism of GRU is specified in Appendix C. The state $\boldsymbol{s_{it}}$ is used to predict the Q-values $Q(\boldsymbol{s_{it}}, a_t|\theta_{RNN})$ of executing different action a_t given the state. The parameter θ_{RNN} of the DRQN model can be learned by minimizing the same loss function outlined in Equation 2. We also incorporate Double Q-Learning (Van Hasselt et al. 2016) and Dueling Network Architecture (Wang et al. 2016) into the DRQN model. The main benefit of these variants is that they improve the learning stability and generalize learning across actions without imposing any changes to the underlying RL algorithm.

[6] The detailed structure of the $\boldsymbol{hi_{it}}$ vector is specified in Appendix B. Prospect reaction is the 1/0 code to represent the click/unclick of the ad in the historical interaction. But the prospect reaction to the incoming interaction is unknown, we set it to -1.

4.2. DRQN-Attention

RL has great potential for improving the long-term revenue of customer acquisition decisions. However, such complex models usually do not explain their decisions, which is a barrier to adoption. Therefore, researchers revise the RL models by adding explanation components to build intrinsic explainable models. Such intrinsic explainable models aim to achieve explainability while maintaining highly competitive performance (Wang et al. 2022a). For instance, while the classical MAB uses a pure mechanical algorithm to select the next instance to trial, Cao and Leng (2021) integrate contextual bandit and decision tree into a unified model that adaptively collects new data points and provides explanations of model decisions by splitting user feature space via feature importance signaling.

Following this stream of research, we aim to empower the aforementioned RL models by explaining their decisions in a more intuitive fashion. Recently, the attention mechanism has been adopted to enhance the inherent explainability of machine learning models (Vaswani et al. 2017). Specifically, the attention mechanism is a technique that mimics the cognitive attention of human beings, which identifies "relevant" information from the input data and fades out the rest depending on the current cognitive task. As the attention layer explicitly weights input features, those weights can be used to identify which pieces of input features the model utilizes for different decisions. For example, Leng et al. (2020) adopt an attention mechanism within the geometric deep learning model to help increase the explainability of recommender systems on the social network. Attention mechanism has also been applied in the video game setting to enhance the explainability of the RL model's decisions during game plays (Mott et al. 2019).

Motivated by recent advancements in the literature, we endeavor to enhance the DRQN by integrating a tailored attention mechanism. Drawing on the general attention mechanism outlined in Section 2.2, our goal is to customize it based on the contextual characteristics of the customer acquisition problem. Specifically, we aim for the attention mechanism to dynamically adjust based on the prospect's static information and historical interactions. The objective is to identify task-relevant features from each incoming ad exposure opportunity, ultimately enhancing the customer acquisition policy. Given that static information and historical interactions are fixed and immutable, the model's focus is directed toward recognizing incoming

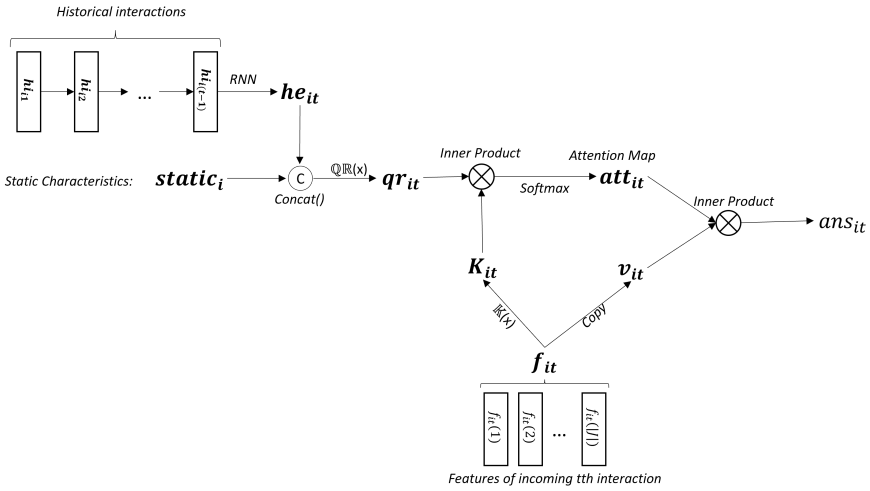
interactions for targeting opportunities. In the upcoming section, we will first enumerate the various data relevant to this context. Subsequently, we will delve into how to leverage these data for the customized attention mechanism.

For the advertising opportunity of the incoming tth interaction with prospect i, DRQN-Attention collects three parts of data: 1) $\boldsymbol{static_i}$ describes the static characteristics of the focal prospect. 2) Historical interactions up to the $(t-1)$th interaction with prospect i, i.e., $\{\boldsymbol{hi_{i_1}}, \boldsymbol{hi_{i_2}}, ..., \boldsymbol{hi_{i_{(t-1)}}}\}$. 3) $\boldsymbol{f_{it}}$ contains the features of the tth interaction with prospect i, which consists of contextual features describing the incoming ad exposure opportunity (e.g. App, location, connection type, time,...). $\boldsymbol{f_{it}}$ is represented via one-hot encoding as it is the target variable we will apply attention weight and the one-hot encoding schema of $\boldsymbol{f_{it}}$ makes attention weight explainable. The attention mechanism relies on the first two parts of the data (i.e., static characteristics and historical interactions) to place weights on the one-hot encoding representation of the features in the incoming interaction to learn the optimal targeting policy. Thus, with different static info and historical interactions, the model can automatically adjust the attention weights, signaling which features of the incoming interaction are used to make the decision.

As outlined in Figure 4, we customize a (Key, Value, and Query) system for attention mechanism to weigh the features of the incoming interaction based on static characteristics and historical interactions. All $|J|$ features of the incoming interaction $\{\boldsymbol{f_{it}}(1), ..., \boldsymbol{f_{it}}(|J|)\}$ are candidates to which we will place the attention weight. We first encode these features into a set of key-value pairs. We apply a dense layer $\mathbb{K}(x)$ on $\boldsymbol{f_{it}}$ to get a $k \times |J|$ key value matrix $\boldsymbol{K_{it}}$, where the jth scalar feature in $\boldsymbol{f_{it}}$ will be transformed to a k dimensional key column $\boldsymbol{K_{it}}(j)$. The value vector $\boldsymbol{v_{it}}$ will be set as the copy of $\boldsymbol{f_{it}}$. Then, we generate the query vector $\boldsymbol{qr_{it}}$ via query generating dense function $\mathbb{QR}(x)$ by processing the concatenation of prospect's static characteristics $Static_i$ and historical embedding $\boldsymbol{he_{it}}$ (generated by RNN that process the historical interactions $\{\boldsymbol{hi_{i_1}}, \boldsymbol{hi_{i_2}}, ..., \boldsymbol{hi_{i_{(t-1)}}}\}$). $\boldsymbol{qr_{it}}$ is a k dimensional vector that will interact with each column in the key matrix $\boldsymbol{K_{it}}$. Therefore, the attention logit map $\widetilde{\boldsymbol{att}}_{\boldsymbol{it}}(j)$ in Equation 4 below is the inner product between query vector $\boldsymbol{qr_{it}}$ and the jth column in $\boldsymbol{K_{it}}(j)$:

$$\widetilde{\boldsymbol{att}}_{\boldsymbol{it}}(j) = \frac{\boldsymbol{qr_{it}}\boldsymbol{K_{it}}(j)}{\sqrt{k}} \tag{4}$$

Figure 4 Outline of the Customized Attention Mechanism in the DRQN-Attention Model



Note: The customized attention mechanism uses static characteristics of the prospect and her historical interactions to query the features of the incoming interaction to identify task-relevant features. We first process historical interactions $\{\boldsymbol{hi_{i_1}},...,\boldsymbol{hi_{i(t-1)}}\}$ with a RNN to get historical embedding he_{it}. Then we construct the query vector $\boldsymbol{qr_{it}}$ based on the concatenation of he_{it} with the static feature of the prospect $\boldsymbol{static_i}$. Afterward, we generate the key value matrix $\boldsymbol{K_{it}}$ based on $\boldsymbol{f_{it}}$, the one-hot encoding feature vector of the incoming interaction. We then apply the query vector $\boldsymbol{qr_{it}}$ to query each column in key matrix $\boldsymbol{K_{it}}$. The result will enter a softmax function to generate the attention weight $\boldsymbol{att_{it}}$ across all $|J|$ features of the incoming interaction. Finally, we associate attention weight with value vector $\boldsymbol{v_{it}}$ (copy of $\boldsymbol{f_{it}}$) to generate the answer ans_{it} of the query. The answer will be used to learn the state representation $\boldsymbol{s_{it}}$ of the prospect.

We then take the softmax function on $\widetilde{\boldsymbol{att_{it}}}$ and get the normalized attention weight $\boldsymbol{att_{it}}(j)$:

$$\boldsymbol{att_{it}}(j) = \frac{exp(\widetilde{\boldsymbol{att_{it}}}(j))}{\sum_{j'} exp(\widetilde{\boldsymbol{att_{it}}}(j'))} \tag{5}$$

Each attention weight $\boldsymbol{att_{it}}(j)$ measures the importance of the jth feature of the incoming tth interaction when the RL model makes the targeting decision. Consequently, we create the answer ans_{it} to the query by multiplying $\boldsymbol{att_{it}}(j)$ to the corresponding value $\boldsymbol{v_{it}}(j)$ in $\boldsymbol{v_{it}}$ and summing over all $|J|$ features:

$$ans_{it} = \sum_{j=1}^{|J|} \boldsymbol{att_{it}}(j)\boldsymbol{v_{it}}(j) \tag{6}$$

The above process describes one attention procedure. Following Vaswani et al. (2017), we adopt a multi-head attention schema with n heads to generate n answers. Finally, all n answers $\{ans_{it}(1), ans_{it}(2), ..., ans_{it}(n)\}$ and n queries $\{\boldsymbol{qr_{it}}(1), \boldsymbol{qr_{it}}(2), ..., \boldsymbol{qr_{it}}(n)\}$ will form as the input to state generating layer $\mathbb{S}(x)$ to derive the state representation $\boldsymbol{s_{it}}$ for prospect i.

$$\boldsymbol{s_{it}} = \mathbb{S}(ans_{it}(1), ..., ans_{it}(n), \boldsymbol{qr_{it}}(1), ..., \boldsymbol{qr_{it}}(n)) \tag{7}$$

Similar to DRQN, the state $\boldsymbol{s_{it}}$ will be used to predict the Q-value $Q(\boldsymbol{s_{it}}, a_{it}|\boldsymbol{\Theta})$ for executing action a_{it}, where $\boldsymbol{\Theta}$ is a set that contains parameters of RNN and customized attention mechanism. Thus, we can seamlessly integrate the attention mechanism into the Q-Learning framework to learn the optimal targeting policy via Equation 1. Several properties of the proposed DRQN-Attention model are worth highlighting: First, the model is fully differentiable. All parameters in $\boldsymbol{\Theta}$ can be trained via back-propagation by minimizing the loss function in Equation 2. Second, the query vectors are a function of static characteristics and historical interactions of the focal prospect–this allows an automatic "forward planning" mechanism where the query function can actively query all the features in the next ad exposure opportunity, aiming to optimize the long-term reward. This process will unveil important features in the next ad exposure for decision-making. In Section 6, we will show how attention weights can guide the model to choose advertising channels under different scenarios.

5. Empirical Analysis

5.1. Data

A random sample of 50k prospects is selected from the potential customer pool of the bank. For each prospect, the bank collects her/his static characteristics, including age, gender, residential city, and marital status. The clickstream data associated with these prospects are also collected over the period of Jun-Dec 2019. Visits to all advertising channels (third-party mobile Apps) of each prospect are tracked via device ID. Upon a prospect's visit to any advertising channels, the contextual features of this interaction are sent to the bank's server. These contextual features include the DateTime, Time Gap (in minutes) to the previous interaction, App ID, internet connection type, device type, and TOL (Type of Location).[7] After receiving the contextual features, the company's customer acquisition system (Contextual Multi-Armed Bandit) decides whether to show an ad and, if yes, which ad to show. If an ad is displayed, the prospect may ignore the ad,

[7] The data collection compliance with Personal Information Protection Law of the People's Republic of China.

or click the ad. After clicking the ad, the prospect can further submit a credit application, which is counted as a success customer acquisition. Table 2 presents the data summary.

To guide the model training, we need to set the reward properly. As the bank pays ¥75 per 1K ad exposures across all the collaborated Apps, we set the unit cost of ad exposure to ¥0.075. After consulting with our data provider, we set the average revenues of an ad click and a customer acquisition as ¥2.60 and ¥260, respectively. As the goal of the model is bringing in new customers and standard service will be applied to all customers, we do not distinguish the rewards of acquiring customers with different lifetime values. Additionally, after a prospect submits an application, the bank reviews it and decides whether to accept it or not. This decision-making process is carried out by the bank and is independent of the proposed model, which only aims to attract prospects to submit applications.

Table 2 Data Summary

	Entities	**Unique Number**
Static Feature	Age	52
	Gender	3
	Associated Industry	6
	City	312
	Marital Status	2
Dynamic Feature	Month	12
	Weekday	7
	Time	24
	Holiday	2
	Time Gap	51,165
	Internet Connection	3
	TOL	10
	Device	5
	App	96
	Ad	195
	Entities	**Total Number**
Interaction Summary	Interaction	3,555,175
	Ad Exposure	1,683,180
	Ad Click	77,210
	Customer Acquisition	81

Note: TOL: Type of Location (e.g. Restaurant)

5.2. Benchmark Models and Evaluation

We randomly divided the data into a training sample (80% of the prospects) and a hold-out sample (20% of the prospects). For evaluating the short-term and long-term revenues from the proposed model and benchmarks, we employ off-policy RL model evaluation. The detailed mechanism of off-policy evaluation is

outlined in Appendix D. Specifically, we compare the performance of the proposed model with the following RL algorithms:

1. **Contextual Multi-Armed Bandit** (Lu et al. 2010). Our partner currently adopts contextual MAB in their customer acquisition system. The model takes the inputs that combine the content features of the ad, static characteristics of the customer, historical interactions as well as contextual features of the incoming interaction. Given a state, the objective of the model is to learn the reward distribution (Gaussian) for each action. Then, ε-greedy is adopted to balance exploration and exploitation to decide the action to deploy.[8] The estimation is updated in real-time based on the feedback of the prospect. In contrast to the proposed model that aims to optimize the long-term reward, MAB is a myopic approach that focuses on immediate reward optimization (CTR).

2. **Deep Q-Network (DQN)** (Mnih et al. 2015). Here we adopt the classical DQN that uses a deep feed-forward network to process the features of the recent interactions with a prospect to summarize the state of the prospect. Specifically, following Mnih et al. (2015), we use a deep feed-forward network to process the concatenation of the latest four interactions the agent has encountered to generate state $\boldsymbol{s_{it}}$. Then the state is connected with a regular dense layer (regression) to predict Q-values of executing different actions. To ensure learning stability and generalizability, we also adopt Double-Q-Learning (Van Hasselt et al. 2016) and Dueling architectures (Wang et al. 2016) in DQN. It is worth noting that such a model setup makes it identical to the RL model proposed by Wang et al. (2022b) for sequential promotion problems.

3. **Interpretable DQN (I-DQN)** (Annasamy and Sycara 2019). DQN is not interpretable. Accordingly, I-DQN was developed to incorporate an attention mechanism to make model decisions more transparent. In contrast to the proposed model, I-DQN uses CNN to process the latest game frame (we apply CNN to the latest four interactions matrix in our setting) to generate the query, while the keys and values are randomly generated and linked to a specific action and Q-value pair. We adopted an open-source code [9] and trained the model on the empirical data. Following the original paper, we utilized the Upper Confidence Bound algorithm to direct action selection after training the model.

[8] We could not disclose the setting of epsilon due to NDA. By exploring the data generated by contextual MAB in Appendix E, we find the contextual MAB does try different actions under different states to introduce action randomness in the exploration process.

[9] Code: https://github.com/maraghuram/I-DQN

4. **Deep Recurrent Q-Network (DRQN)** (Hausknecht and Stone 2015). DRQN accommodates POMDP by processing the historical interactions using an RNN to summarize the state of the prospect. Different from the proposed model that integrates RNN within the attention mechanism to learn the state representation, DRQN utilizes standard many-to-one structure RNN to learn state representation $\boldsymbol{s_{it}}$ directly. Then the state is connected with a regular dense layer to predict Q-values of executing different actions. To ensure learning stability and generalizability, we also adopt Double-Q-Learning (Van Hasselt et al. 2016) and Dueling architectures (Wang et al. 2016) in DRQN.

5. **Dyna-DRQN** (Sutton and Barto 2018). The DQN, DRQN above, and the proposed model belong to the model-free RL family that does not explicitly model the environment. Such models require substantial sample data collected from the environment to improve the policy. The Dyna architecture revises the Q-learning framework that explicitly models the environment. It simultaneously learns from observations to model the environment and utilizes both observations and simulated data from the learned model to optimize the policy. Such adjustments make Dyna Q-Learning a more sampling-efficient model.

6. **Spatio-Temporal Attention Deep Recurrent Q-Network (ST-ADRQN)**. Etchart et al. (2019) introduced an attention network between LSTM's timesteps to learn the most important states from history. The output of LSTM at each timestamp is used as a state to estimate the Q values for different actions. DRQN-Attention is different from ST-ADRQN as DRQN-Attention aims to customize the attention mechanism to identify which channel is promising for the incoming interaction, while ST-ADRQN adopts the attention mechanism as to track back into the history to identify "important states". Additionally, ST-ADRQN utilizes the same data to construct all three components (i.e. Query, Key, Value) of the attention network, we use the historical state and static information as query, while features in the incoming interaction are used for key and value. Finally, there is an attention network in every timestep in ST-ADRQN, while there is only one attention network in the DRQN-Attention model.

In the off-policy evaluation, we adopt Fixed-M PERS (Mandel et al. 2016) that aims to sample episodes from the testing set that the sampled episodes follow a similar policy to the candidate RL model. An episode with length H represents an ordered trajectory of actions, rewards, and states, $(s_0, a_0, r_0, s_1, a_1, r_1, ..., r_H)$. For

Table 3 Average Reward Comparison across Different Models and Different Episode Lengths

Algorithm	AR@1	AR@3	AR@6
MAB	0.0568	0.1721	0.3406
DQN	0.0397	0.1939	0.3856
I-DQN	0.0351	0.1836	0.3789
DRQN	0.0450	0.1983	0.3993
ST-ADRQN	0.0382	0.1667	0.3628
Dyna-DRQN	0.0379	0.1621	0.3507
DRQN-Attention	0.0412	0.1990	0.3999

Note: All average rewards are statistically significantly different from that of MAB at 1% significance level.

example, when $H = 3$, we sample candidate episodes as $\{s_1, a_1, r_1, s_2, a_2, r_2, s_3, a_3, r_3\}$. For all the sampled episodes, we calculate the average reward per episode to evaluate model performance. More details of off-policy evaluation can be find in Appendix D. Table 3 compares the average rewards of sampled episodes via the off-policy evaluation with different lengths ($H = 1, 3, 6$). We have the following observations: 1) MAB performs better than all Q-Learning models (DQN, DRQN, Dyna-DRQN, DRQN-Attention) when we set $H = 1$, but Q-Learning models surpass MAB with longer episodes $H = 3, 6$. Such a comparison shows MAB is indeed superior at learning intervention strategies to optimize immediate reward, but Q-Learning models will learn strategical policies that better balance immediate reward and long-term rewards to achieve greater overall rewards. 2) DRQN and DRQN-Attention lead to higher rewards than the vanilla DQN. Their superior performances confirm the importance of incorporating more historical interactions and the necessity of using RNN to accommodate POMDP. 3) Compared to DRQN, the attention mechanism in DRQN-Attention does not sacrifice performance, as their performances are not statistically significantly different ($P = 0.708$). However, incorporating the attention mechanism helps explain the decision process of the model by emphasizing those task-relevant features of the incoming interaction (Section 6). 4) I-DQN also adopts an attention mechanism, but the design of the attention mechanism makes it not perform as well as the proposed DRQN-Attention. The key and value components in I-DQN are randomly generated and specific to an action and Q-value pair, serving to identify states where the agent expects to receive a specific Q-value by taking the given action. Due to the action space being much larger in our setting (e.g. 195 ad in our setting verse 5 actions in the spaceinvader game used in Annasamy and Sycara (2019)), such a treatment contains excessive noisy information as key and value are randomly generated to each action

and Q-value pair. Additionally, only using the latest interactions to learn the query might not accommodate POMDP. Thus, such attention mechanism design leads to inferior model performance in the customer acquisition setting. 5) ST-ADRQN also adopts attention networks into DRQN. But DRQN-Attention only has one attention module that processes the output of RNN, but ST-ADRQN has an attention module at each time step within RNN. Additionally, ST-ADRQN uses the same data to learn query, key and value in the attention network. Such a model setting makes ST-ADRQN more complex but less informative than DRQN-Attention. Beyond the testing performance in Table 3, we found ST-ADRQN was not fitting well in the training set as well. Such observation is consistent with the finding of ST-ADRQN performance in Fully Observable Pong game (Etchart et al. 2019). This is explained by He et al. (2016) that deeper and more complex neural networks are not always necessary to perform better than their simpler counterparts, the main reason is such models are harder to train. 6) Finally, Dyna-DRQN does not perform as well as other models, potentially because using simulated data introduces noise that undermines performance.

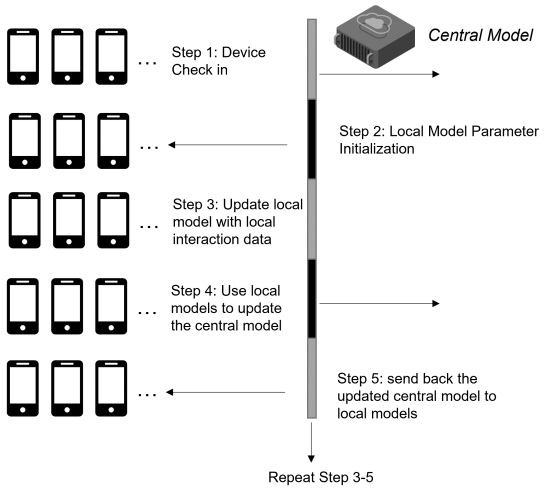
Furthermore, by comparing the average reward per interaction across different H values, we find out that the improvement in the performance of DRQN-Attention over MAB is not linear. When $H = 3$, the improvement of average reward per interaction of DRQN-Attention over MAB is ¥0.0089 (≈(0.1990-0.1721)/3). With $H = 6$, such improvement increases to ¥0.0098 (≈(0.3999-0.3406)/6). This comparison further confirms that the RL model aims to optimize the long-term reward rather than the immediate reward. To sum up, DRQN-Attention outperforms the company's current policy and other RL algorithms, showing its effectiveness in customer acquisition.

5.3. Federated RL for Privacy-Sensitive Context

Customers are becoming increasingly aware and protective of their data. With the rise of regulations such as GDPR and the initiatives from iOS and Android platforms, protecting user data privacy has become a widespread global trend (Kollnig et al. 2021).

Correspondingly, we should consider preventing personal data leakage and protecting personal privacy when applying the DRQN-Attention model. Privacy-Enhancing Technologies (PETs) have recently been identified as a pivotal area for future research in the context of privacy regulation because they offer a

Figure 5 Federated Learning of RL models



promising solution to the ongoing tension among privacy concerns, customer welfare, and the development of data-driven economy (Johnson 2022). The Information Commissioner's Office of the United Kingdom government defines PETs as "technologies that embody fundamental data protection principles by minimizing personal data use, maximizing data security, and empowering individuals" (ICO 2022). Federated Learning (FL) is one of the notable examples of PETs. Thus, we adopt federated learning to distribute the model training and deployment. Federated learning enables collaboration between agents without exchanging raw data, ensuring the privacy of sensitive information. This integration of federated learning and RL has led to the research area of federated RL (Qi et al. 2021). Specifically, we adopt the federated RL policy from Tehrani et al. (2021) to learn and deploy RL models. The federated approach allows us to train the model while keeping personal data stored and processed locally on local devices, with only intermediate model updates being communicated to a central model. The central model then aggregates the updates and sends a refined model back to the devices to support sequential targeting decisions. The federated learning process is illustrated in Figure 5. We evaluate the performance of the federated version of RL models using the same training and testing set. The result, shown in Table 4, indicates that the federated versions have a similar performance to the original version (Table 3), with only a marginal drop. This confirms the model can be applied in privacy-sensitive settings without compromising the service quality. [10]

[10] Appendix H discuss how to enhance the security and privacy of FL models to different attacks.

Table 4 Average Reward (AR) Comparison across RL Models via Federated Learning

Algorithm	AR@1	AR@3	AR@6
DQN	0.0388	0.1806	0.3809
I-DQN	0.0339	0.1802	0.3735
DRQN	0.0437	0.1956	0.3950
ST-ADRQN	0.0368	0.1619	0.3513
DRQN-Attention	0.0403	0.1958	0.3892

6. What do Attention Weights Capture?

While the attention map may not encapsulate the entire decision-making process, it provides valuable insights into the strategies learned by the model. In this section, we explore what attention weights capture.

For each interaction, the attention weights, denoted as $\boldsymbol{att_{it}}$, interact with $\boldsymbol{v_{it}}$ (a copy of $\boldsymbol{f_{it}}$) in Equation 6 through element-wise multiplication. The outcome of Equation 6 is then used to estimate the state $\boldsymbol{s_{it}}$ in Equation 7, influencing subsequent model decisions. Therefore, the high-weighted elements in $\boldsymbol{att_{it}}$ significantly impact state construction and, consequently, the model's decisions. Given that $\boldsymbol{f_{it}}$ is a one-hot encoding vector, each element in $\boldsymbol{f_{it}}$ is binary and represents a feature in the incoming interaction. For instance, if the 1st element in $\boldsymbol{f_{it}}$ is equal to 1, it indicates that the incoming interaction is from APP 1; similarly, if the 97th element is equal to 1, it denotes that the interaction occurred on Monday. Consequently, we are interested in the high-weight element in the attention weights $\boldsymbol{att_{it}}$ and whether the corresponding element in $\boldsymbol{f_{it}}$ is filled with 1. Specifically, within each $\boldsymbol{att_{it}}$, we identify the elements with the top 10 attention weights. We then categorize each interaction into two groups: 1) "Matched with Attention Weight": if there are ≥ 5 corresponding elements in $\boldsymbol{f_{it}}$ equal to 1. 2) "Unmatched with Attention Weight": if there are ≤ 2 corresponding elements in $\boldsymbol{f_{it}}$ equal to 1. In Table 5 (first two rows), we observe that across episodes with different lengths, the average rewards of episodes in the "Matched with Attention Weight" group consistently exceed those in the "Unmatched" group. This suggests that attention weight may be aimed at identifying features in the incoming session that lead to higher rewards.

However, if we are only interested in identifying those features of the incoming interaction that lead to higher rewards, there are other simpler options such as the association rule (Piatetsky-Shapiro 1991). Association rule intends to identify strong rules in the dataset using measures such as *confidence*. The

Table 5 Average Reward (AR) Comparison between Attention Mechanism and Association Rule

Group	AR@1	AR@3	AR@6
Matched with Attention Weight	0.0417	0.2298	0.4815
Unmatched with Attention Weight	0.0218	0.0845	0.1738
Matched with Association Rule	0.0433	0.1939	0.3946

confidence value of a rule, $X \Rightarrow Y$, is defined as, among all transactions starting from X, the proportion of the transactions containing Y. Specifically, we may define static characteristics and historical interactions as X, features of the incoming interaction as Y, and calculate the confidence values from all consecutive observations. We restrict this analysis to the incoming interactions that lead to positive rewards (e.g. click or customer acquisition). Therefore, the association rules could also capture which features of the incoming interactions are more likely to lead to higher rewards. In fact, in our data, the attention weights and the weights of features in Y based on the association rule have a high correlation of 0.8412. Similar to how we assign an incoming interaction into the "Matched with Attention Weight" group above, if an interaction contains ≥ 5 features with top-10 confidence values of the association rule, we assign it to the "Matched with Association Rule" group. In Table 5, we further compare the average rewards of episodes "Matched with Association Rule" against those "Matched with Attention Weight" (row 3 and row 1). With the exception of AR@1, episodes "Matched with Association Rule" consistently result in lower average rewards. The reason behind the attention weight having higher average rewards is that the association rule only focuses on immediate outcomes of the incoming interaction with prospects, while the attention mechanism considers optimal trajectories that lead to higher long-term rewards. Accordingly, the "Matched with Association Rule" group has a higher average reward with AR@1, but is dominated by the "Matched with Attention Weight" group with longer episodes. *Such a comparison shows that the attention mechanism spots those features in the incoming interaction that aim for optimal long-term reward. Consequently, the model learns to scan through possible future paths and decides on optimal ones with promising outcomes.* [11]

[11] To validate the explainability of model decisions using attention weights, we follow the attention weight validation procedure outlined by Serrano and Smith (2019). The results indicate that the high-weighted features indeed play a crucial role in generating model decisions. Additional details can be found in Appendix F.

Table 6 Attention Weight based on Industry

	IT	Fiance&Law	Education
Top 1	Time Gap	Time Gap	Time Gap
Top 2	APP: LiveStream1	Connection: WIFI	Monday
Top 3	Device: Cellphone	APP: LiveStream1	Tuesday
Top 4	Monday	TOL: Fiance &Insurance	Connection: WIFI
Top 5	APP: TechNews1	Device: Cellphone	APP: System2
Top 6	APP: FinanceNews1	Cumulative Impression	APP: E-Reading1
Top 7	TOL: Office Building	TOL: Office Building	APP: News3
Top 8	Cumulative Impression	APP: FinanceNews1	PP: Browser1
Top 9	TOL: Company&Factory	Connection: 4G	Device: Cellphone
Top 10	Connection: WIFI	APP: Browser1	TOL: Education&Research

7. Attention Weight and Advertising Channels

In this section, we explore how attention mechanism chooses advertising channels, aiming to obtain a better understanding of the decision-making process of the proposed model. Using three illustrative cases, we demonstrate how the attention mechanism plans the optimal path for ad exposures. In particular, we analyze how the attention mechanism adjusts its advertising channel choices across individual prospects and over time. The three specific cases include: 1) advertising channel choice for prospects from different industries, 2) advertising channel adjustment according to dynamic prospect behaviors, and 3) marketing channel planning that accounts for the seasonality of the agricultural industry. To facilitate these analyses, we first define two new concepts: 1) **Most Important Features** (*MIF*). For each $\boldsymbol{att_{it}}$, we call the corresponding 10 features in $\boldsymbol{f_{it}}$ with the highest weights in $\boldsymbol{att_{it}}$ as top-10 features. Across all $\boldsymbol{att_{it}}$, some features frequently appear as one of the top-10 features. We select features whose frequencies of being in the top-10 lists are above the 80-percentile and label them as *Most Important Features (MIF)*. In total, there are 26 *MIF*. *MIF* can be viewed as features that are important in general across all $\boldsymbol{att_{it}}$. 2) **Singular Features**. Given an $\boldsymbol{att_{it}}$, any of its top-10 features that are not within *MIF* will be considered as singular feature. Singular features can be viewed as how the attention weights of an interaction deviate from the general pattern.

Case 1: Advertising Channel Choice based on Associated Industry. In this first case, we collected all the interactions associated with groups of prospects working in specific industries, including IT, Finance

Table 7 Attention Weight Adjusted To Different Customer Behaviors

	Case 2.1	Case 2.2
Top 1	Time Gap	Time Gap
Top 2	Device: Cellphone	APP: LiveStream1
Top 3	AD Platform3	APP: CCTV1
Top 4	Wednesday	Device: Cellphone
Top 5	APP: E-Reading1	APP: Trans1
Top 6	Cumulative Impression	Cumulative Impression
Top 8	Connection: UK	Wednesday
Top 9	APP: Browser2	APP: Browser2
Top 10	APP: System2	Connection: UK

& Law, and Education. For interactions associated with each group of these prospects, we extracted the top 10 features based on the average attention weight and highlighted those singular features in Table 6. First, we noticed that prospects from the IT and Finance & Law industries share many similarities, as singular features like "APP: FianceNews1" (an app for financial news) and "TOL: Office Building" (locations like office buildings) are within the top 10 feature list for both. This makes sense as most of them work in office buildings and are interested in financial news to manage their businesses. However, prospects from the IT industry have their specific singular features like "APP: TechNews1" (an app for technology news) and "TOL: Company & Factory" (locations like companies and factories), while prospects from Finance & Law have their special singular feature of "TOL: Finance & Insurance" (locations like banks and financial institutions). Such comparison shows that the model can learn to target prospects from different industries based on their unique characteristics. When we turn to teachers in the education industry, the model learns completely different targeting patterns. The singular features of "APP: News3" (an app for general news) and "TOL: Education & Research" (schools and businesses in the education industry) appear in the top 10 list. Thus, we understand that technology news and financial news are less promising channels to target teachers compared to general news. Additionally, teachers are more likely to be located in schools or educational institutes rather than office buildings.

Case2: Advertising Channel Adjustment due to Dynamic Customer Behaviors. "APP: LiveStream1" (an App for streaming live sporting events and dramas) is the 5th *MIF* and a very popular app for targeting

Table 8 Attention Weight to Target Agricultural Entrepreneurs at Two Stages

	Pre Peak	Peak
Top 1	Time Gap	Time Gap
Top 2	APP: Weather1	APP: Weather1
Top 3	AD Platform3	APP: News1
Top 4	APP: News1	AD Platform3
Top 5	APP: CCTV1	TOL: Fiance &Insurance
Top 6	Cumulative Impression	APP: CCTV1
Top 8	APP: LiveStream2	Cumulative Impression
Top 9	APP: Browser2	APP: FinanceNews1
Top 10	APP: E-Commerce1	TOL: Company&Factory

customers. In Table 7, for case 2.1, we collect those interactions with prospects whose previous interactions include at least five ad exposures on this App, but no ad click or customer acquisition occurred. We then find the top-10 features for Case1 by averaging the attention weights in this group. We observe that "APP: LiveStream1" is not one of the top-10 features, even though it is the 5th MIF. The disappearance of "APP: LiveStream1" indicates that the model is able to automatically adjust to avoid ineffective channels. For case 2.2 in the same table, we turn our attention to "APP: CCTV1" (a closed-circuit television monitoring app), an App not being one of the MIF. We collect those interactions of prospects whose previous interactions include one or more ad clicks on this app. For the subsequent interactions with this group of prospects, the top 10 features include two Singular Features: "APP:CCTV1" and "APP:Trans1" (an App for language translation). It is unsurprising to see that "APP:CCTV1" is being emphasized. But why is "APP:Trans1" also being promoted? From the data, we find out the probability of an ad click on "APP:Trans1" conditional on a previous ad click on "APP:CCTV1" is 23.82%, substantially higher than the average rate in the data ($< 0.01\%$). The appearance of "APP:Trans1" as one top-10 feature indicates that the model can learn to target prospects across Apps over time for better long-term rewards. Such insight could help managers better understand the coordination of different ad channels for customer acquisition.

Case 3: Advertising Channel Seasonality for Targeting Agricultural Entrepreneurs. In this exercise, we analyze how the proposed model learns to target prospects who are identified as agricultural entrepreneurs by the bank. Major agricultural loans in China are seasonal (Agricultural 2022), which are used to finance grain purchase, storage, and transportation during the peak harvest seasons of agricultural

products. Due to the seasonality of such loans, it is interesting to see how the model behaves during different periods of the year. In the second half of 2019 (the duration of our data), July-September and September-December are the peak seasons for summer and fall grain purchases, respectively (LSWZ 2022). Thus, we segment the second half of 2019 into two stages: (1) Pre-Peak stage: June (2) Peak stage: July-December. The top 10 features based on the average attention weight during pre-peak and peak seasons for agricultural entrepreneurs are shown in Table 8. Singular Features are highlighted. It is not surprising to see non-*MIF* Apps "App: Weather1" (a weather forecasting app) and "APP: CCTV1" are among the top-10 lists, because they are essential Apps for the agricultural industry and hence effective channels to target agricultural entrepreneurs. Moreover, non-*MIF* Apps "APP: E-Commerce1" and "APP: LiveStream2" become promising advertising channels and appear in the top-10 list during the pre-peak stage, indicating the potential customers lean towards consumption and entertainment before the peak season. Thus, showing ads in these channels could influence potential customers by priming them with the bank as a future lender. Once the peak season arrives, non-*MIF* "TOL: Finance &Insurance" and "APP: FinanceNews1" become prominent on the top-10 list. They are strong signals as visiting a brick-and-mortar Finance & Insurance facility or browsing finance information online indicates the user might need financial services. Thus, the model learns to precisely target the prospects to meet their needs.

The agricultural industry exhibits different seasonality patterns than other industries, and these nuanced insights in Case 3 are typically overlooked or not precisely captured by marketing managers. They provided valuable assistance to our data provider on two fronts: 1. Industry Trends Monitoring: Agricultural industry clients were not traditionally part of our data provider's clientele. However, the RL model continuously updated its targeting policy through continuous interactions with prospects, signaling emerging patterns in the agricultural industry. This enabled managers to recognize the needs within this sector and its unique funding patterns. Consequently, they engaged with agricultural industry associations before the peak season to disseminate loan information and establish dedicated channels to meet these needs during peak seasons. Thus, the insights helped monitor industry trends and transformed business-to-consumer (2C) transactions into business-to-business (2B) business opportunities. 2. Customized Product Design: After identifying the

seasonality patterns in the agricultural industry, our data provider introduced a customized financial product specifically tailored for agricultural entrepreneurs. While other industries typically have steady cash flows, grain purchases/sell in agriculture occur only during short periods with intense transactions, with long idle periods in between.. Consequently, the customized product was designed to align with the seasonality of the agricultural industry, adopting a more flexible repayment mode instead of regular monthly installments.

Allen et al. (2023) discussed practical validation strategies for interpretations from explainable machine learning models. A key validation strategy involves randomly splitting the available data into a training and test set and validating the findings in the testing set. The fundamental idea is to utilize explainable machine learning models on the training data to generate interpretations or insights. Then, one can evaluate the model's insights on the test data. In each case above, we aims to conclude that the specific advertising channels in each case could lead to more promising outcomes (e.g., higher long-term rewards). Thus, we could verify in the testing data whether such insights and conclusions are valid. We employ the same validation strategy as in Section 6, which compares the attention weight with association rules. For each case in this section, we collect relevant data from the testing set. For example, we gather all data associated with the occupation of teachers. We then split this data into two sets: 1) Matched: where the prospect landed on one of the recommended advertising channels and an ad was shown to the prospect. 2) Unmatched: where the prospect didn't land on any of the recommended advertising channels, or an ad wasn't shown to the prospect in the recommended advertising channel. We also compare the average reward @1, 3, and 6 of these two groups in all three cases in this section. As shown in Table 9, the average reward in the matched group is higher than that of the unmatched group. Such a comparison validates the insights gained from the proposed model on the testing set, proving that the learned insights from the three cases are valid.

The presented cases illustrate that the proposed model can effectively learn optimal advertising channels for diverse populations, adapt to dynamic customer behaviors, and grasp the seasonality of the industry, allowing for corresponding adjustments in advertising channels. In addition to advertising channel analysis, attention weights are employed to scrutinize decisions related to the delivery of different ads (see Appendix I). A comparison with a post-hoc explainable model is also conducted to highlight differences in explainability with the proposed intrinsic explainable model, detailed in Appendix G.

Table 9 Average Reward between Matched/Unmatched Groups with Recommend Advertising Channels

Case	Group	AR@1	AR@3	AR@6
Industry: IT	Matched	0.0308	0.1037	0.2263
	Unmatched	0.0202	0.0805	0.1705
Industry: Fiance&Law	Matched	0.0284	0.0935	0.1866
	Unmatched	0.0213	0.0783	0.1760
Industry: Education	Matched	0.0253	0.0854	0.1580
	Unmatched	0.0206	0.0732	0.1532
Case 2.1	Matched	0.0262	0.0807	0.1538
	Unmatched	0.0217	0.0683	0.1361
Case 2.2	Matched	0.0382	0.2033	0.4539
	Unmatched	0.0191	0.0862	0.1701
Industry: Agriculture	Matched	0.0306	0.0583	0.1367
(Pre Peak)	Unmatched	0.0211	0.0517	0.1233
Industry: Agriculture	Matched	0.0736	0.2368	0.4236
(Peak)	Unmatched	0.0226	0.0757	0.1321

8. Conclusion

We introduce the DRQN-Attention model that aims to optimize the long-term reward for customer acquisition and meanwhile provide meaningful explanations for the model decisions. Our empirical analyses show that the proposed model results in higher long-term revenues than state-of-the-art RL models. Furthermore, we provide evidence that DRQN-Attention is able to pinpoint important features of targeting opportunities and guide advertising channel choices while presenting intuitive explanations for the decisions of the RL model. In short, the proposed model has good explainability and applicability and thus can be applied widely for customer acquisition. This study provides a valuable tool for digital businesses seeking to improve their customer acquisition efforts. By leveraging sequential messaging through RL, the proposed model can optimize customer acquisition and drive potential customers down the acquisition funnel.

We acknowledge certain limitations in our current study. Our proposed model excels in learning policies to optimize long-term rewards and reveals attention weights indicating how the model targets different prospects. However, the attention weight itself is not directly interpretable like the results of the Decision Tree, and extra effects like grouping based on static information or dynamic information of prospects and comparing with the most important features are needed to learn the general pattern of targeting different prospects. Future research avenues could explore methods for directly learning such patterns from the data.

Moreover, it's essential to note that the empirical data in our study may not comprehensively cover the whole state-action space. Consequently, the learned policy in our reinforcement learning model may not be globally optimal. If this model gets a chance to run in the on-policy environment, potential enhancements include employing Boltzmann exploration (Bertsekas and Tsitsiklis 1995) or other exploration approaches to refine the learned policy and update insights into targeting different prospect types. Additionally, the RL model learns to target prospects based on their static and dynamic features. A promising future direction involves incorporating a fairness perspective into the proposed model. This enhancement aims to ensure equitable delivery of marketing messages across prospects while still maintaining the goal of optimizing long-term rewards. Lastly, the proposed DRQN-Attention model presents a versatile solution designed to learn optimal sequential intervention policies for user/customer acquisition. Importantly, this model is not inherently reliant on cross-app tracking. Its applicability extends to various contexts, such as within a website or an app, where it can effectively deliver tailored sequential messages for each user interaction within the respective platform.

References

Agricultural BoC (2022) Agriculture seasonal loan. URL `https://www.abchina.com/cn/RuralSvc/Businesses/jjdk/`.

Allen GI, Gan L, Zheng L (2023) Interpretable machine learning for discovery: Statistical challenges and opportunities. *Annual Review of Statistics and Its Application* 11.

Annasamy RM, Sycara K (2019) Towards better interpretability in deep q-networks. *Proceedings of the AAAI conference on artificial intelligence*, volume 33, 4561–4569.

AppsFlyer (2021) Fintech marketers invested $3b on user acquisition in 2020 according to appsflyer. URL `https://www.bloomberg.com/press-releases/2021-06-09/fintech-marketers-invested-3b-on-user-acquisition-in-2020-according-to-appsflyer`.

Bank W (2023) Small and medium enterprises (smes) financ. *International Journal of Industrial Organization* URL `https://www.worldbank.org/en/topic/smefinance`.

Bento Ja, Saleiro P, Cruz AF, Figueiredo MA, Bizarro P (2021) Timeshap: Explaining recurrent models through sequence perturbations. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery Data Mining*, 2565–2573 (New York, NY, USA: Association for Computing Machinery), ISBN 9781450383325, URL `http://dx.doi.org/10.1145/3447548.3467166`.

Berente N, Gu B, Recker J, Santhanam R (2021) Managing artificial intelligence. *MIS quarterly* 45(3):1433–1450.

Bertsekas DP, Tsitsiklis JN (1995) Neuro-dynamic programming: an overview. *Proceedings of 1995 34th IEEE conference on decision and control*, volume 1, 560–564 (IEEE).

Brauwers G, Frasincar F (2021) A general survey on attention mechanisms in deep learning. *IEEE Transactions on Knowledge and Data Engineering* .

Cao J, Leng Y (2021) Adaptive data acquisition for personalized recommender systems with optimality guarantees on short-form video platforms. *Available at SSRN 3843086* .

Chaudhari S, Mithal V, Polatkan G, Ramanath R (2021) An attentive survey of attention models. *ACM Transactions on Intelligent Systems and Technology (TIST)* 12(5):1–32.

Cho K, Van Merriënboer B, Bahdanau D, Bengio Y (2014) On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* .

Chuck L (2022) The four pillars of customer acquisition strategy. URL `https://www.forbes.com/sites/forbesbusinesscouncil/2022/08/15/the-four-pillars-of-customer-acquisition-strategy/?sh=a84f6f63218d`.

Corporation IF (2017) *MSME Finance Gap: Assessment of the Shortfalls and Opportunities in Financing Micro, Small, and Medium Enterprises in Emerging Markets* (World Bank).

Du M, Liu N, Hu X (2019) Techniques for interpretable machine learning. *Communications of the ACM* 63(1):68–77.

Etchart M, Ladosz P, Mulvaney D (2019) Spatio-temporal attention deep recurrent q-network for pomdps. *Progress in Artificial Intelligence: 19th EPIA Conference on Artificial Intelligence, EPIA 2019, Vila Real, Portugal, September 3–6, 2019, Proceedings, Part I 19*, 98–105 (Springer).

Fei H, Zhang Y, Ren Y, Ji D (2021) Optimizing attention for sequence modeling via reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems* 33(8):3612–3621.

Fernández-Loría C, Provost F (2022) Causal decision making and causal effect estimation are not the same… and why it matters. *INFORMS Journal on Data Science* 1(1):4–16.

Frick TW, Belo R, Telang R (2022) Incentive misalignments in programmatic advertising: Evidence from a randomized field experiment. *Management Science* .

Géron A (2022) *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow* (" O'Reilly Media, Inc.").

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*.

He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hochreiter S, Schmidhuber J (1997) Long short-term memory. *Neural computation* 9(8):1735–1780.

Hoffman DL, Novak TP (2000) How to acquire customers on the web. *Harvard business review* .

ICO ICO (2022) Draft anonymisation, pseudonymization and privacy enhancing technologies guidance, .

Itaya H, Hirakawa T, Yamashita T, Fujiyoshi H, Sugiura K (2021) Visual explanation using attention mechanism in actor-critic-based deep reinforcement learning. *2021 International Joint Conference On Neural Networks (IJCNN)*, 1–10 (IEEE).

Johnson G (2022) Economic research on privacy regulation: Lessons from the gdpr and beyond .

Kiran BR, Sobh I, Talpaert V, Mannion P, Al Sallab AA, Yogamani S, Pérez P (2021) Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems* .

Kokkodis M, Ipeirotis PG (2020) Demand-aware career path recommendations: A reinforcement learning approach. *Management Science* .

Kollnig K, Binns R, Van Kleek M, Lyngs U, Zhao J, Tinsman C, Shadbolt N (2021) Before and after gdpr: tracking in mobile apps. *Internet Policy Review* 10(4).

Leng Y, Ruiz R, Dong X, Pentland A (2020) Interpretable recommender system with heterogeneous information: A geometric deep learning perspective. *Available at SSRN 3696092* .

LSWZ (2022) Grain circulation data, national food and strategic reserves administration. URL `http://www.lswz.gov.cn/html/ywpd/lstk/tj-sgsj.shtml`.

Lu T, Pál D, Pál M (2010) Contextual multi-armed bandits. *Proceedings of the Thirteenth international conference on Artificial Intelligence and Statistics*, 485–492.

Lundberg SM, Lee SI (2017) A unified approach to interpreting model predictions. Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, Garnett R, eds., *Advances in Neural Information Processing Systems 30*, 4765–4774 (Curran Associates, Inc.), URL `http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf`.

Mandel T, Liu YE, Brunskill E, Popović Z (2016) Offline evaluation of online reinforcement learning algorithms. *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Milani S, Topin N, Veloso M, Fang F (2022) A survey of explainable reinforcement learning. *arXiv preprint arXiv:2202.08434* .

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).

Mott A, Zoran D, Chrzanowski M, Wierstra D, Jimenez Rezende D (2019) Towards interpretable reinforcement learning using attention augmented agents. volume 32.

Piatetsky-Shapiro G (1991) Discovery, analysis, and presentation of strong rules. *Knowledge discovery in databases* .

Precup D (2000) Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series* 80.

Puiutta E, Veith EM (2020) Explainable reinforcement learning: A survey. *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, 77–95 (Springer).

Qi J, Zhou Q, Lei L, Zheng K (2021) Federated reinforcement learning: Techniques, applications, and open challenges. *arXiv preprint arXiv:2108.11887* .

Reiley D, Lewis RA, Schreiner T (2012) Ad attributes and attribution: Large-scale field experiments measure online customer acquisition. *Available at SSRN 2049457* .

Schwartz EM, Bradlow ET, Fader PS (2017) Customer acquisition via display advertising using multi-armed bandit experiments. *Marketing Science* 36(4):500–522.

Serrano S, Smith NA (2019) Is attention interpretable? *arXiv preprint arXiv:1906.03731* .

Shao K, Tang Z, Zhu Y, Li N, Zhao D (2019) A survey of deep reinforcement learning in video games. *arXiv preprint arXiv:1912.10944* .

Shi W, Huang G, Song S, Wang Z, Lin T, Wu C (2020) Self-supervised discovering of interpretable features for reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44(5):2712–2724.

Silver D, Huang A, Maddison CJ, Guez A, Sifre L, Van Den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, et al. (2016) Mastering the game of go with deep neural networks and tree search. *Nature* 529(7587):484–489.

Simester D, Timoshenko A, Zoumpoulis SI (2020) Targeting prospective customers: Robustness of machine-learning methods to typical data challenges. *Management Science* 66(6):2495–2522.

Song Y, Sahoo N, Srinivasan S, Dellarocas C (2022) Uncovering characteristic response paths of a population. *INFORMS Journal on Computing* 34(3):1661–1680.

Song Y, Sun T (2023) Ensemble experiments to optimize interventions along the customer journey: A reinforcement learning approach. *Management Science* .

Sutton RS, Barto AG (2018) *Reinforcement learning: An introduction* (MIT press).

Tehrani P, Restuccia F, Levorato M (2021) Federated deep reinforcement learning for the distributed control of nextg wireless networks. *2021 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 248–253 (IEEE).

Theocharous G, Thomas PS, Ghavamzadeh M (2015) Personalized ad recommendation systems for life-time value optimization with guarantees. *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Torrado RR, Bontrager P, Togelius J, Liu J, Perez-Liebana D (2018) Deep reinforcement learning for general video game ai. *2018 IEEE Conference on Computational Intelligence and Games (CIG)*, 1–8 (IEEE).

Turek M (2018) Explainable artificial intelligence. *Defense Advanced Research Projects Agency, no date. https://www.darpa. mil/program/explainable-artificial-intelligence* .

Van Hasselt H, Guez A, Silver D (2016) Deep reinforcement learning with double q-learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. *NIPS 2017*, 5998–6008.

Wang J, Zhang Y, Tang K, Wu J, Xiong Z (2019) Alphastock: A buying-winners-and-selling-losers investment strategy using interpretable deep reinforcement attention networks. *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 1900–1908.

Wang T, He C, Jin F, Hu YJ (2022a) Evaluating the effectiveness of marketing campaigns for malls using a novel interpretable machine learning model. *Information Systems Research* 33(2):659–677.

Wang W, Li B, Luo X (2022b) Deep reinforcement learning for sequential targeting. *Management Science* .

Wang Z, Schaul T, Hessel M, Hasselt H, Lanctot M, Freitas N (2016) Dueling network architectures for deep reinforcement learning. *International conference on machine learning*, 1995–2003 (PMLR).

Watkins CJ, Dayan P (1992) Q-learning. *Machine learning* 8(3-4):279–292.

Werbos PJ (1990) Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE* 78(10):1550–1560.

Wiegreffe S, Pinter Y (2019) Attention is not not explanation. *arXiv preprint arXiv:1908.04626* .

Xu G, Li H, Liu S, Yang K, Lin X (2019) Verifynet: Secure and verifiable federated learning. *IEEE Transactions on Information Forensics and Security* 15:911–926.

Yang X, Feng Y, Fang W, Shao J, Tang X, Xia ST, Lu R (2022) An accuracy-lossless perturbation method for defending privacy attacks in federated learning. *Proceedings of the ACM Web Conference 2022*, 732–742.

Yin X, Zhu Y, Hu J (2021) A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions. *ACM Computing Surveys (CSUR)* 54(6):1–36.

Zhang J, Curley SP (2018) Exploring explanation effects on consumers' trust in online recommender agents. *International Journal of Human–Computer Interaction* 34(5):421–432.

Zhang Q, Du Q, Liu G (2021) A whole-process interpretable and multi-modal deep reinforcement learning for diagnosis and analysis of alzheimer's disease. *Journal of Neural Engineering* 18(6):066032.

Zhou F, Yang Q, Zhang K, Trajcevski G, Zhong T, Khokhar A (2020) Reinforced spatiotemporal attentive graph neural networks for traffic forecasting. *IEEE Internet of Things Journal* 7(7):6414–6428.

Appendix

A. Table of Notations

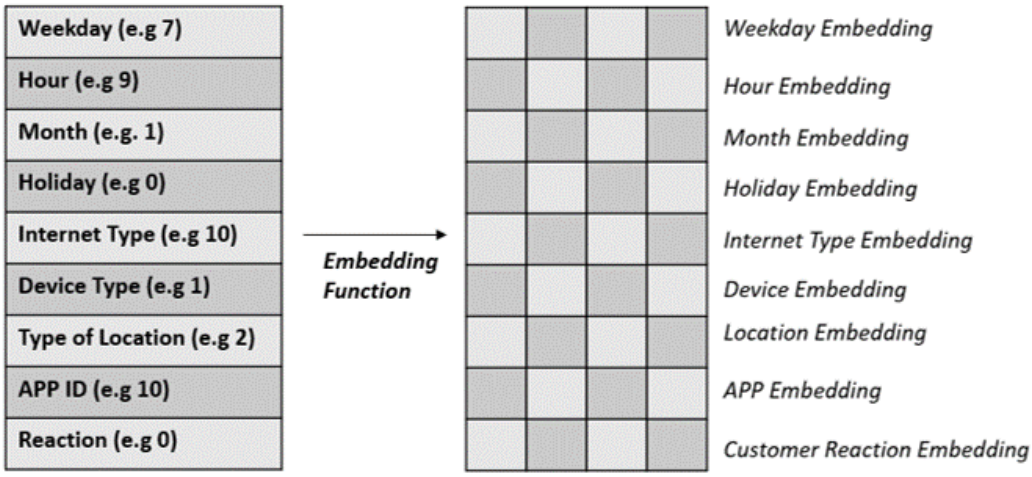
Table 10 Table of Notations

	Notation	Format	Description
Data	i	Scalar	customer index
	t	Scalar	Time index
	j	Scalar	Index of feature in incoming interaction
	k	Scalar	Dimension of query and key
	g	Scalar	Dimension of the state
	h	Scalar	Dimension of the historical interaction
	e	Scalar	Dimension of the embedding for historical interaction
	J	Set	Collection of features in incoming interaction
	$\boldsymbol{hi_{it}}$	$1 \times h$	Historical interaction of prospect i at timestep t
	$\boldsymbol{static_i}$	1×10	Static information of prospect i
	$\boldsymbol{f_{it}}$	$1 \times \lvert J \rvert$	Features of incoming interaction of prospect i at timestep t
RL	$\boldsymbol{s_{it}}$	$1 \times g$	State of prospect i at timestep t
	a_{it}	Scalar	Action of RL model to prospect i at timestep t
	r_{it}	Scalar	Immediate reward of taking action a_{it} under state $\boldsymbol{s_t}$
	$Q(\boldsymbol{s_t}, a_t)$	Scalar	Expected cumulative reward after taking action a_t under state $\boldsymbol{s_t}$
	γ	Scalar	Discount factor in the Bellman Equation
Attention Mechanism	$\boldsymbol{qr_{it}}$	$1 \times k$	Query vector
	$\boldsymbol{K_{it}}$	$k \times \lvert J \rvert$	Key Matrix
	$\boldsymbol{v_{it}}$	$1 \times \lvert J \rvert$	Value vector
	$\boldsymbol{att_{it}}$	$1 \times \lvert J \rvert$	Attention weight vector
	$\widetilde{\boldsymbol{att_{it}}}$	$1 \times \lvert J \rvert$	Pre-Normalized attention weight vector
GRU	$\boldsymbol{m_t}$	$g \times 1$	Hidden state of GRU at time t
	$\boldsymbol{z_t}$	$g \times 1$	Update gate of GRU at time t
	$\boldsymbol{re_t}$	$g \times 1$	Reset gate of GRU at time t
	$\boldsymbol{W_z}$	$g \times e$	Parameter matrix in update gate
	$\boldsymbol{W_r}$	$g \times e$	Parameter matrix in reset gate
	$\boldsymbol{W_m}$	$g \times e$	Parameter matrix in state update function
	$\boldsymbol{U_z}$	$g \times e$	Parameter matrix in update gate
	$\boldsymbol{U_r}$	$g \times e$	Parameter matrix in reset gate
	$\boldsymbol{U_m}$	$g \times e$	Parameter matrix in state update function
	$\boldsymbol{b_z}$	$g \times 1$	Parameter vector in update gate
	$\boldsymbol{b_r}$	$g \times 1$	Parameter vector in reset gate
	$\boldsymbol{b_m}$	$g \times 1$	Parameter vector in state update function

B. Structure of Contextual Vectors for Historical Interaction

The structure of $\boldsymbol{hi_{it}}$ is illustrated in Figure 6. Specifically, $\boldsymbol{hi_{it}}$ describes the agent's tth interaction with prospect i that consists of the contextual features of the interaction (including DateTime, device, location, app, prospect reaction, etc). Different element in the vector represents different information. For example, 7 in the first element means this interaction happened on Sunday. 9 in the second element means this interaction happened at 9am. 10 in the 8th element

Figure 6 Structure of Historical Interaction Vector



means that this interaction happened on the APP with the id of 10. The vector $\boldsymbol{hi_{it}}$ will be processed by an embedding function $\mathbb{E}()$. Specifically, we utilize the embedding function provided by Keras, as detailed in the API documentation [12]. This function is designed to convert positive integers (indexes) into dense vectors of a fixed size. As shown in Figure 6, when applying the embedding function to APP ID, it transforms into a 4-dimensional vector. For instance, APP ID1 will be represented as [0.16, 0.78, 0.13, 0.62], while APP ID 5 results in a different 4-dimensional vector [0.81, 0.32, 0.75, 0.96]. The choice to use the embedding function rather than opting for one-hot encoding for categorical data is motivated by a desire to avoid sparse input, particularly when dealing with a large number of choices in the category, such as 96 Apps in our setting. As explained in Geron's book (Géron 2022) Chapter 13 (pages 466-471), an embedding serves as a trainable dense vector representing a category. For instance, the Keras function $Embedding(7,2)$ indicates there are 7 choices in the category, and each choice is represented by a 2-dimensional vector. The matrices of size 7x2 are parameters that undergo learning in the network training process.

It is crucial to clarify that our model explanation or attention mechanism is applied to the features of incoming interactions $\boldsymbol{f_{it}}$, as opposed to historical interactions $\boldsymbol{hi_{it}}$. Unlike the features of historical interactions, which undergo processing by the embedding function, we have opted for one-hot encoding for the features of incoming interactions. The rationale behind this choice stems from the fact that embedding vectors lack explainability. Applying attention weights directly to black-box embedding vectors could obscure the decision-making process of the model. Therefore, for the features of incoming interactions, we have chosen one-hot encoding to make the visualization of high-weight

[12] https://keras.io/api/layers/core_layers/embedding/

features, as presented in Tables 5-7, feasible. This choice ensures the vector representation of f_{it} remains explainable. For example, if there is a '1' in the 5th element of the one-hot encoding vector, it signifies that the incoming interaction is from App 5. The attention weight is subsequently applied to this one-hot encoding vector. Consequently, the highly weighted values in the attention weight have direct and explainable correspondences. This approach provides a clear understanding of which features in the incoming interaction the attention weight is focusing on.

C. Recurrent Neural Network to Learn State Representation

A Recurrent Neural Network (RNN) is a type of neural network that aims to process sequential data or time series data. These deep learning algorithms are commonly used for temporal problems, such as language translation, natural language processing, and speech recognition. In our setting, we use RNN to process historical interactions to learn state representation. Specifically, we choose the many-to-one RNN structure [13], where the input is the historical interactions and the only output at the end of the RNN is the state representation. We have explored three options for RNN: 1) Vanilla RNN (Werbos 1990), 2) Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber 1997), and 3) Gated Recurrent Unit (GRU) (Cho et al. 2014). GRU is selected due to its superior model performance.

There are two gates in a GRU at a given time step t: the update gate and reset gate:

$$\boldsymbol{z_{it}} = \sigma(\boldsymbol{W_z}\mathbb{E}(\boldsymbol{hi_{it}}) + \boldsymbol{U_z} m_{i(t-1)} + \boldsymbol{b_z}) \tag{8}$$

$$\boldsymbol{re_{it}} = \sigma(\boldsymbol{W_r}\mathbb{E}(\boldsymbol{hi_{it}}) + \boldsymbol{U_r} m_{i(t-1)} + \boldsymbol{b_r}) \tag{9}$$

where $\boldsymbol{hi_{it}}$ is the historical interaction for prospect i at time t, $\mathbb{E}()$ is the embedding function, $m_{i(t-1)}$ is the memory from the previous time step, and σ is the sigmiod function. Addtionally, $\boldsymbol{W_z}$, $\boldsymbol{U_z}$, and $\boldsymbol{b_z}$ are parameters associated with the update gate $\boldsymbol{z_{it}}$. $\boldsymbol{W_r}$, $\boldsymbol{U_r}$, and $\boldsymbol{b_r}$ are parameters related to the reset gate. With two gates ready, we can update the candidate's hidden state $\hat{m}_{it}$ as:

$$\hat{m}_{it} = \phi(\boldsymbol{W_m}\mathbb{E}(\boldsymbol{hi_{it}}) + \boldsymbol{U_m}(\boldsymbol{re_{it}} \otimes m_{i(t-1)}) + \boldsymbol{b_m}) \tag{10}$$

where $\otimes$ is Hadamard product operator and ϕ is tanh activation function. $\boldsymbol{W_m}$, $\boldsymbol{U_m}$, and $\boldsymbol{b_m}$ are parameters utilized for updating the candidate hidden state. The updated memory at time step t is defined as:

[13] https://stanford.edu/ shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks

$$\boldsymbol{m_{it}} = \boldsymbol{z_{it}} \otimes m_{i(t-1)} + (1 - \boldsymbol{z_{it}}) \otimes \hat{m}_{it} \tag{11}$$

Within this updating process, $\boldsymbol{m_{io}}$ denotes the initial state at step 0. Adhering to the standard setting in GRU and other RNN models, $\boldsymbol{m_{io}}$ is randomly generated. All the follow-up states in GRU will be updated based on EQ 11. By feeding GRU with historical interaction $\{\boldsymbol{hi_{i1}}, ..., \boldsymbol{hi_{it}}\}$, the output of GRU at the last step is $\boldsymbol{m_{it}}$, which is used to represent state $\boldsymbol{s_{it}}$ in DRQN model. However, for the DRQN-Attention model, the GRU generates $\boldsymbol{he_{it}}$ (historical embedding) by process input sequence $\{\boldsymbol{hi_{i1}}, ..., \boldsymbol{hi_{i(t-1)}}\}$, as depicted in Section 4.2 of the manuscript.

D. Off-Policy Evaluation

Many RL algorithms assume that an agent actively interacts with an online environment to learn from its own collected experience and evaluate the learned model in the same setting. So, the performance of these algorithms is evaluated via on-policy interaction with the environment. Traditionally, algorithms have been evaluated via on-policy format on simple hand-designed problems, often with a small number of states. Recently, many works adopted simulators (e.g., Atari video games environment (Mnih et al. 2015)) as a testbed, or directly played with human experts on those well-designed games (Silver et al. 2016), to evaluate RL algorithms. However, it is challenging to evaluate RL algorithms on real-world problems (e.g., business decisions) via on-policy, as it can be extremely expensive and risky to collect extensive data using an unjustified RL system to interact with the real-world environment. On the other hand, those applications in simulation settings require high-fidelity simulators that are challenging to build. Fortunately, for many applications, there exist pre-collected data which can be utilized to make RL training and evaluation feasible, and enable better generalization by incorporating diverse prior experiences.

Off-policy RL using an offline dataset of logged interactions is an important tool for real-world applications. Off-policy RL can help (1) train an RL model, and (2) empirically evaluate the RL model using existing data, while the existing data is generated by an executing agent that is different from the model to be evaluated.[14] From the model training perspective, the Q-learning framework that DRQN-Attention builds on is a well-known off-policy RL algorithm (Sutton and Barto 2018).

We aim to sample episodes with different lengths that match the learned policy of the proposed model. Here an episode represents an ordered trajectory of states, actions, and rewards. For example, episode

[14] There is no requirement that the executing agent must be a purely random policy. Instead, we only need to know the probabilities of generating different actions under different states of the executing agent.

$\{s_1, a_1, r_1, s_2, a_2, r_2, ..., s_H, a_H, r_H\}$ is obtained by executing the agent (i.e., contextual multi-armed bandit in our setting) H times in the environment. However, there is a mismatch of distributions: we need episodes sampled to follow the distribution of the proposed model (i.e., DRQN-Attention); but we have data drawn from the distribution of the executing agent (i.e., contextual MAB). Importance sampling is a technique for handling such a mismatch and there is a wide range of off-policy evaluators in the literature (Precup 2000, Theocharous et al. 2015). Unfortunately, most of these off-policy evaluators assumed the environment is MDP, which is inappropriate in our setting. Therefore, we adopt Fixed-M per-episode rejection sampling (Fixed-M PERS, Mandel et al. 2016) to evaluate the model performance, as this evaluator naturally accommodates POMDP. Intuitively, Fixed-M PERS aims to sample from hold-out data and select those episodes more favorably when they match the learned action policy of the RL model to be evaluated. Fixed-M PERS is a well-established method with the following nice properties: 1) When applying Fixed-M PERS, given the current history and that the algorithm accepts episode samples of observations and rewards, the observations and rewards are drawn from a distribution identical to the distribution the algorithm would have encountered if it was to run online. This is known as true sample property [15]. 2) Based on the episode samples accepted by Fixed-M PERS, we can derive an unbiased estimate of the reward obtained by the RL algorithm in the episode if it was to run online. This is known as unbiased estimation of the average reward of episode property. 3) The dynamics of the environment in our empirical setting depend not only on the most recent observation (i.e., MDP) but also on the full history of interactions (i.e., POMDP). Fixed-M PERS is a representation-agnostic evaluator, which does not require Markov assumption.

When applying Fixed-M PERS, we need to decide the length of episodes to sample. Denote the length of the episode as H and we evaluate the performance of different models with $H = 1, 3, 6$.[16] For example, when $H = 3$, we sample candidate episodes as $\{s_1, a_1, r_1, s_2, a_2, r_2, s_3, a_3, r_3\}$. For all the sampled episodes, we calculate the average reward per episode to evaluate model performance. The detailed algorithm is listed below:

Fixed-M PERS (Mandel et al. 2016) aims to sample from the dataset D that contains episodes, where each episode d represents an ordered trajectory of actions, rewards, and states, $(s_0, a_0, r_0, s_1, a_1, r_1, ..., r_H)$ obtained by executing existing policy e for H steps in the environment. Fixed-M PERS performs rejection sampling at the episode level. It

[15] The requirement for sample true (Mandel et al. (2016) Theorem 6.1) is met in our empirical setting. Please refer to Appendix E for more details.

[16] Mandel et al. (2016) show Fixed-M PERS is less efficient for longer episodes as the acceptance rate in the rejection sampling fell sharply, leading to few or zero accepted episodes. We also found that the average number of ad exposure for customer acquisition is 3.6296, and 86% of the customer acquisitions get less than or equal to 6 ad exposures. Thus, we set maximal $H = 6$.

Algorithm 1 Fixed-M Per-Episode Rejection Sampling Evaluator

1: Input: Executing policy e, evaluated policy b, state space S, binary transition matrix T denoting whether a nonzero transition probability from one state to another, maximum horizon H, and Dataset of episodes D with each episode length of H.
2: Output: AER, where $AER(i)$ is the ith Accepted Episode cumulated Reward
3: Initialize $M_s = 1.0$ for all $s \in S$
4: **for** h=1 to H **do**
5: **for** $s \in \mathcal{S}$ **do**
6: Update $M_s^{'} = max_a \frac{\pi_b(a|s)}{\pi_e(a|s)} max_{s'}(T(s,a,s')M_{s'})$
7: **end for**
8: $M = M^{'}$
9: **end for**
10: $i = 1$
11: **for** $d \in \mathcal{D}$ **do**
12: $p = 1.0, R = 0, s = []$
13: Get start state st of the episode d
14: **for** $(o,a,r) \in d$ **do**
15: $s = (s,o)$
16: $p = p\frac{\pi_b(a|s)}{\pi_e(a|s)}$
17: $R = r + \gamma R$
18: **end for**
19: Sample $\mu \sim Uniform(0,1)$
20: **if** $\mu \leq \frac{p}{M_{st}}$ **then**
21: $AER(i) = R$
22: $i = i + 1$
23: **end if**
24: **end for**
25: **return** AER

will first compare evaluated policy b (RL model needs to evaluate) against the executing policy e (MAB) to get the ratio of the probability of executing an action under a state between two policies, then accept or reject the episode according to whether a random variable sampled from the uniform distribution is lower than the computed ratio. In order to ensure that rejection sampling returns a sample from the candidate distribution that represents the distribution as applying evaluated policy b online, it is critical to set ratio normalization constant M correctly. Define the probability of executing action a under state s via policy e as $\pi_e(a|s)$, the probability of executing action a under state s via policy b as $\pi_b(a|s)$. The ratio $\frac{\pi_b(a|s)}{\pi_e(a|s)}$ can grow extremely large, we need an M such that $\frac{\pi_b(a|s)}{M\pi_e(a|s)}$ is a probability between 0 and 1. Therefore, M should be assigned as a constant that represents the maximum possible ratio. The detailed implementation of Fixed-M PERS is shown in Algorithm 1, where the for-loop between lines 4-10 is used to construct M, and the for-loop in lines 11-24 for episode-level rejection sampling.

We note that the learned intervention policy may only be optimal within the explored space given the training data but may be sub-optimal globally. This is mainly caused by the off-policy training using the archived data, where the model can only learn from the limited action combinations within the explored space, and there may be unexplored action combinations that could result in better performance. The DRQN-Attention model, when applied in an

on-policy setting, can benefit from RL exploration strategies to enhance intervention policies. RL models inherently balance exploitation and exploration, taking advantage of learned policies for stable revenue while also exploring new intervention opportunities for policy improvement. Such exploration could be a new advertisement, a new advertisement channel, or existing channels but under-explored. This can be achieved through classical exploration methods such as ε-greedy (Sutton and Barto 2018) and Boltzmann exploration (Bertsekas and Tsitsiklis 1995). The key idea of ε-greedy and Boltzmann exploration is not only utilizing the known promising action (e.g. the action leads to the highest Q-Value), but there are chances to explore other actions which might have the potential to improve the future reward. Additionally, the model can be further optimized using the Upper Confidence Bound (UCB) (Annasamy and Sycara 2019) to balance the exploration and exploitation.

E. Action Randomness of the Collected Data

In off-policy RL, action policy π_b of the proposed model (e.g. DRQN-Attention) aims to learn from the data that is drawn from a different action policy π_e of the executing agent (e.g. contextual MAB). If $\pi_e(a|s) = 0$ for certain state-action pair, the proposed policy π_b will never get a chance to explore executing a under state s, which might lead to learning an inferior targeting policy due to under-exploration. To ensure the proposed model could learn optimal targeting policies under different states, it would be better to minimize the cases of $\pi_e(a|s) = 0$. As the contextual MAB adopts ε-greedy to balance exploration and exploitation, there is a randomness to guide contextual MAB to try different actions under different states. To show the randomness, we first check how many unique actions have been executed when a customer lands an APP. The average number of unique actions is 190.5 (theoretical maximal is 196) per APP, indicating that the executing agent does explore almost all the actions on different APPs. While APP is just a proportion of features being used to construct state representation and other features (historical interactions, time, location) also matter. Thus, we further check the unique number of actions being explored under different states. As the states are represented by 20-dimensional vectors, we first run K-means clustering on all the state representations. Based on the elbow method, we set the number of clusters to 54. Then we check how many unique actions have been executed under different state clusters. The average number of unique actions is 190.1 (theoretical maximal is 196) per state cluster, this also indicates that the executing agent does explore almost all the actions under different states. Therefore, we can conclude that the executing agent does comprehensive state-action explorations, which will help the proposed model learn optimal policy.

For model evaluation, Fixed-M PERS requires $\pi_b(ep) > 0 \rightarrow \pi_e(ep) > 0$ for all possible episodes ep (Mandel et al. 2016). We have checked the proposed model π_b and executing agent π_e for all possible episodes ep with episode length of $H = 1, 3, 6$, the requirement is met in our empirical setting.

F. Validation of Attention Weight

Serrano and Smith (2019) tested the reliability of attention mechanisms in Natural Language Processing (NLP) settings. They used a bidirectional LSTM to process word tokens and connected it with an attention mechanism to predict the target variable (such as a review rating or the sentiment of a text). To test the validity of the attention weights, they zeroed out the attention weights from highest to lowest and re-normalized the remaining weights. If the attention weights were reliable, a change in the model prediction should occur quickly. However, the authors found that they had to zero out 90% of the weights on average to observe a change in the model's decision, suggesting that the attention weights were not explainable. This highlights the need for further validation of explainability through attention weights in NLP tasks.

We have adopted the same validation strategy in Serrano and Smith (2019) to the proposed DRQN-attention model. We zero out the attention weights from highest to lowest and re-normalized the remaining weights. Unlike the NLP tasks, where 90% of the weights had to be zeroed out to change the model decision, in our setting, on average, only 7.1% of the highest weights had to be zeroed out (standard deviation 0.013). This sensitivity of the attention weights in our setting can be attributed to the way the attention weights are applied. In our model, the attention weights are applied to the incoming interaction features, which are represented through one-hot encoding (96 dimensions for APPs, 7 dimensions for location types, 5 dimensions for devices, etc.). These features are generally independent of each other. However, in NLP tasks, the attention weights are applied to word embeddings, and there are strong contextual dependencies between words in any text. As a result, if a few keywords are erased, the meaning of the text can still be inferred and the overall tone will not change. On the other hand, in our setting, changing any feature might mean: 1) the user is using APP1 rather APP2, 2) the user is on a mobile phone rather than Pad, 3) the incoming interaction happens on Monday rather than Friday. Thus, a minor change in the feature vector of the incoming interaction can dramatically change the context of the interaction, leading to the sensitivity of the attention weights. As noted by Wiegreffe and Pinter (2019), the validity of attention weights is highly dependent on the task, and attention weights work well in uncontextualized settings. From our analysis, we have tested the sensitivity and quality of the attention weights in the proposed model, and the results show that the highly weighted features are highly responsible for the model's decisions.

G. Model Explainability Comparison

The proposed model, as an intrinsic explainable model, can be compared with a post-hoc explainable model to highlight the differences in explainability between these two representative explainable models. The closest black box

model to the proposed model is DRQN, and we applied TimeSHAP (Bento et al. 2021) on top of this model. Bento et al. (2021) underscores that blindly applying SHAP (Lundberg and Lee 2017) to RNNs may overlook the importance of past events and features throughout the sequence. It tends to attribute significance predominantly to features of the current input. In response to these limitations, we have explored TimeSHAP (Bento et al. 2021), a model-agnostic recurrent explainer. TimeSHAP builds upon KernelSHAP and extends its capabilities to the sequential domain, offering a more comprehensive solution to address the challenges posed by applying SHAP to RNN-style models.

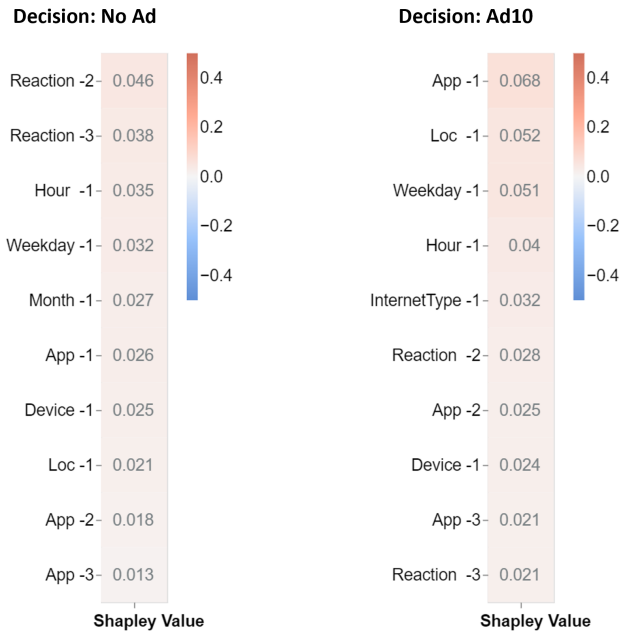
TimeSHAP provides a variety of methods, each serving specific purposes based on the desired explanations[17]. Local methods offer a detailed view of a model decision corresponding to a specific sequence being explained. On the other hand, global methods aggregate local explanations of a given dataset to present a holistic view of the model's behavior. Within the global methods, TimeSHAP enables the analysis of the Shapley value of a time lag (e.g., the effect of all entities in lag 2) or a specific input entity (e.g., the effect of historical customer reaction). However, it's noteworthy that there isn't a dedicated tool for the analysis of time lag $\times$ entity interactions (e.g., the Shapley value of customer reaction in lag 2). Considering our specific analysis needs, we have opted for the local event analysis tool in TimeSHAP. This choice facilitates the examination of randomly selected cases, offering detailed insights into model decisions at the level of individual sequences.

We present the Shapley values for two distinct cases: one with no-ad decision and the other involving the decision to show AD10. Figure 7 visually represents the SHAP values for these cases. In the plot, the -1 index corresponds to features of the incoming interaction, while the -2 index is associated with features in the latest historical interaction. Observing the plot, we discern that the customer's reactions in the previous interactions (-2, -3) and the temporal information of the incoming interaction predominantly influence the decision not to show the ad. This observation suggests that the model learns from the customer reactions in preceding interactions, determining that targeting this particular customer may not be financially advantageous. Contrastingly, the decision to display AD10 in the second case is primarily influenced by the information pertaining to the incoming interaction, such as APP, location, weekday, and hour. This signals that the DRQN model adaptively relies on different types of information to make the decision.

The analyses of TimeSHAP explanations reveal distinct differences with DRQN-Attention: 1) TimeSHAP explanations are intricately tied to specific decisions, providing insights into which features in historical and incoming interactions contributed to a particular model decision. DRQN-Attention offers dynamic explanations based on static

[17] https://github.com/feedzai/timeshap/tree/main

Figure 7 **Top Sharpley values for two cases with a decision of not showing AD, verse showing AD10**



characteristics, historical interactions, and contextual information. 2) The explanations from TimeSHAP aim to identify which features in the historical interaction as well as incoming interaction are more responsible for the model decision. However, the proposed model looks forward to identifying which features in the incoming interaction signal a more profitable targeting. These differences highlight the varied applications of the two models.

H. Federal Learning and Attacks

To ensure data security and privacy in FL, it is crucial to consider potential attacks. Two relevant types of attacks are: passive attacks, involving the observation of information or learning from a system without altering it, and active attacks, aiming to manipulate the system's resources or operations (Yin et al. 2021). In the context of FL, passive attackers typically observe computations, including weights, gradients, and the final model during the training and inference phases. On the other hand, active attackers may seek to influence the FL system by manipulating model parameters to achieve adversarial goals. For the proposed model, we can enhance privacy protection by employing commonly adopted cryptographic techniques (Xu et al. 2019) and perturbation techniques (Yang et al. 2022) within FL. Cryptographic techniques widely used in privacy-preserving machine learning include homomorphic encryption, secret sharing, and secure multi-party computation. Perturbation techniques involve adding noise to the original data, ensuring that statistical information calculated from the perturbed data remains statistically indistinguishable from the original data. These methods could enhance the security and privacy of FL models in our context.

I. More Attention Analysis Result

The proposed model could learn optimal advertising channels via attention analysis. Beyond that, attention weight can also be used to analyze the decision of delivering different marketing messages.

The proposed model is trained to deliver different marketing messages (e.g. ads) based on different contexts to optimize the long-term reward. Thus, we can analyze the attention weights based on different actions and the variations could unveil the basis on which the model makes different decisions. For attention weights associated with each ad action, we take the average of attention weights and highlight these singular features are not in MIF. While all ads deliver a similar message that the user could apply at most 3 million RMB with a daily interest rate as low as 0.01%, there are subtle variations in different ads. For AD 33, the distinct feature of this ad is to emphasize the bank is funded by a tech giant in China. The first column in Table 11 shows the top 10 most important feature associated with this ad. Interestingly, we find out that two singular features (APP: Video2 (Endorser) and APP: News2 (Endorser)) in the top-10 list are Apps that are developed by the tech giant. Thus, the proposed model has learned that it will be preferable to show the ad with an endorsement from a prestigious company to potential customers who are existing users of the company's products. Another example is AD 63 and 112 where the two ads share similar content. One major difference between them, however, is that AD 112 has the keyword "Loan specialized to enterprise" while AD 63 emphasizes "Loan specialized to micro-enterprise". Such a nuance leads to the difference in their attention weight shown in the last two columns in Table 11. It is clear that TOL: Office Building is within the top-10 list of both ads, implying that entrepreneurs working at Office Building will be attracted by both ads. However, we also find that TOL: Company&Factory is within the top-10 list of AD 112 and TOL: Restaurant&Shopping is within the top-10 list of AD 63. Such a difference reveals the interesting company size variation among entrepreneurs at different TOLs. Specifically, entrepreneurs at TOL: Company&Factory tend to have a larger company size than those at TOL: Restaurant&Shopping. Accordingly, AD 63 has a greater appeal to "micro-entrepreneurs" while AD 112 appeals to "entrepreneurs". These examples show that the proposed model could learn to deliver the right marketing message to the right target.

Table 11 Feature Weight based on Different Marketing Message

	AD33	AD112	AD63
Top 1	Time Gap	Time Gap	Time Gap
Top 2	APP: LiveStream1	APP: LiveStream1	APP: LiveStream1
Top 3	Device3	WeekDay3	WeekDay3
Top 4	APP: Video2 (Endorser)	Device3	Connection: 4G
Top 5	APP: E-Reading1	TOL: Office Building	TOL: Office Building
Top 6	WeekDay1	Connection: 4G	Connection: UK
Top 7	Cumulative Impression	APP: System$_2$	TOL: Restaurant&Shopping
Top 8	APP: SportsNews1	TOL: Company&Factory	Cumulative Impression
Top 9	APP: News2 (Endorser)	Cumulative Impression	Connection: WIFI
Top 10	APP: System2	Connection: UK	APP: Browser2