Deep Reinforcement Learning-Based Recommender Algorithm Optimization and Intelligent Systems Construction for Business Data Analysis

Fan Xingyu[1, a], Lin Yang[2, b]
[1]Xidian University, Xi'an, China
[2]Marquette University, Wisconsin Milwaukee, USA
[a]faan@stu.xidian.edu.cn, [b]yang.lin1345@gmail.com

2022 IEEE Asia-Pacific Conference on Image Processing, Electronics and Computers (IPEC) | 978-1-6654-0902-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/IPEC54454.2022.9777623

***Abstract*—Business analysis plays an essential role in numerous companies and industries, which involves data collection, data analysis, and the construction of intelligent systems to make critical business decisions. By using a data model to filter through consumers' data, it is easier than ever to make recommendations to consumers for what they would enjoy using or purchasing. Added a real-time product Recommendation System framework, the Reinforcement Learning model is used to automatically learn the optimal recommendation strategy. Also, the Deep Q-Network (DQN) algorithm is modified by replacing the convolutional neural network with the Long Short-Term Memory (LSTM). The solution analyzes and predicts consumers' preferences in real-time, and makes up for the lack of iterative optimization of traditional Recommendation Systems in real scenarios.**

Keywords—Business Analysis, Recommend System, Reinforcement Learning, DQN, LSTM

I. INTRODUCTION

Compared with the traditional local stores, which have no direct communication with consumers and observation of whom behavior, modern services and products are mainly provided online. Although online sales may be difficult to understand customers' preferences accurately, due to the development of machine learning, especially the advancement of deep learning, it is possible to understand millions of customers completely online through business analysis based on which data.

There is an ocean of reasons that businesses use recommender systems. Take Alibaba as an example to explain. By sorting through their clients' search results and what they have purchased, customers' recommendations are constantly offered by information. Since a recommendation system is a perfect way to offer a compelling user experience, it is better to know their customers through content-based approaches, where the company will ensure that they keep coming back to them. As merchants learn what does and does not sell, Alibaba can offer their main target audience what they need exactly, which will quickly lead to more sales and more profit for the company. Moreover, a good user experience will still lead to more sales and more profit, which can provide their users with the right recommendations and ensure that they keep coming back [1].

Recommend systems have become increasingly popular in recent years, which assist customers in their information-seeking tasks by suggesting items (products, services, or information) that best fit their needs and preferences [2]. Nowadays, however, most of the existing recommendation systems regard the recommendation process as a static process for using several basic methods with a fixed greedy strategy. In fact, after receiving the recommendation, there is still a gap between the actual feedback and the expected behavior. The recommendation technique was viewed as a series of interactions between customers and the recommender agent, and reinforcement learning was offered as a way to learn the best suggestion strategies automatically. There are many advantages to choosing reinforcement learning: First of all, to meet the most suitable product recommendation for customers, reinforcement learning considers the nature of interaction in the recommendation process and the possible changes in customers' priority over time. These systems take into account changing choices and dynamic changes in personal circumstances to gain the best strategy; thus, these will be more intelligent when recommending products. In addition, the reinforcement learning-based system obtains the best strategy by maximizing the cumulative reward over time. Therefore, iterative processing will be carried out with the continuous accumulation of data and customer feedback, so that each recommendation is the latest and best recommendation [3].

II. REINFORCEMENT LEARNING

Reinforcement learning is the process of teaching machine learning models to make a series of decisions based on rewarding favorable behaviors and punishing undesirable actions [4]. A reinforcement learning agent can sense and comprehend its surroundings, execute actions, and learn through trial and error in general. Reinforcement learning devises a way of rewarding desired behaviors and punishing undesirable behaviors, assigning positive values to desired activities and negative values to undesirable behaviors to motivate the agent. To obtain an ideal solution, the agent is programmed to seek long-term and maximum overall reward [5].

A. Markov decision process

The solution in this paper can be compared to the MDP pattern, in which the agent is in charge of making product recommendations to potential clients. Generally, each MDP contains the following components corresponding to our product recommendation system:

1) Action space: Action refers to recommending products to a customer based on State.

2) Transition probability: Transition probability refers to the probability that the state changes from S_t to S_{t+1}.

3) Reward: The customer's feedback is used to determine the reward. r_t will be 1 if the buyer purchases the item that the agent advised; otherwise, it will be 0.

4) State space: State is defined as customers' preference, which is generated based on the customer's purchase history, i.e., the products that a customer bought.

5) Discount factor: Discount factor is a number that ranges from 0 to 1. This value represents the relative relevance of future rewards over now rewards. On the other hand, if γ= 1, the agent just considers the immediate reward; if γ= 0, the algorithm analyzes all future rewards for executing the current action [6].

Using the definitions above, our system searches through previous data to discover the best policy $\pi: S \rightarrow A$. Obviously, the technique should maximize the target customer's cumulative reward, and the best policy should also maximize the buying rate in order to find the best product that customers would be interested in and desire to buy. An agent's purpose is to develop an optimal policy, which is one that maximizes cumulative rewards in each state, as specified below:

$$V^*(s) = max_\pi E^\pi\{\sum_{K=0}^{\infty} \gamma^K r_{t+k} | s_t = s\} \quad (1)$$

where π denotes the state s policy that the agent will obey, and E^π is the policy's expectation, t denotes the current time step, and t + k denotes future time steps. Furthermore, at time step (t+k), r_{t+k} is the immediate reward. It was necessary to ensure that the recommendation agent found the best state-action value function, which is defined as follows for all scenarios and actions:

$$Q^*(s,a) = max_\pi E^\pi\{\sum_{K=0}^{\infty} \gamma^K r_{t+k} | s_t = s, a_t = a\} \quad (2)$$

B. Deep- Q Networks (DQN's)

OpenAI was the first to use DQN to play Atari games [7]. To approximate the state-action value function in DQN, convolutional neural networks are used, and stochastic gradient descent is used to optimize the objective function and update the parameter values. The maximum output of the Q-network determines the next action, which is based on all previous experience. The mean squared error (MSE) between the anticipated Q-value and the desired Q-value – Q* is the loss function. The mean squared error loss function [8] is used to train the Q-network , as shown in Figure 1.

$$L_i(\theta_i) = E_{s,a\sim p}.[(y_i - Q(s,a;\theta_i))^2] \quad (3)$$

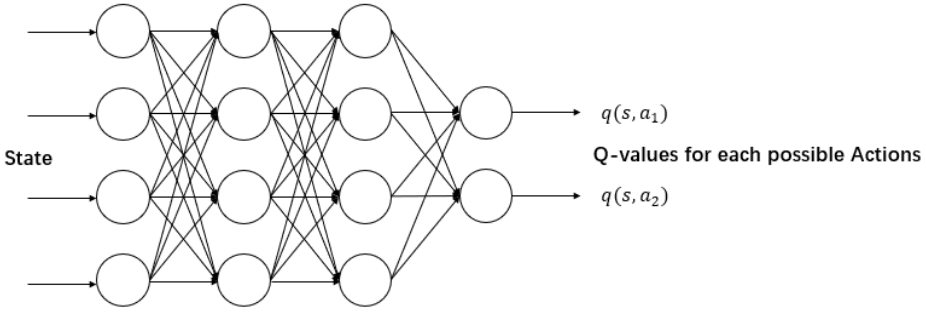

Figure 1. A diagram of policy network DQN

III. DATA PREPARATION AND EXPERIMENTS

A. Data preparation

The dataset is derived from Alibaba [9] and after data processing, 10 user characteristics are filtered out. Collect 100,000 suggestion sessions at random in chronological sequence, with the first 70% serving as the training/validation set and the remaining 30% serving as the test set. The initial state of a new session is gathered from the user's prior sessions. To construct the beginning state in this study, we used N = 10 previously purchased products.

B. Experiments

To begin with, start to train the recommendation agent. Deploy the DQN algorithm at first, which is used to estimate and approximate the state-action value function. Since the data used and the length of the time interval is constantly changing, variable padding is applied for observations before the data enters the convolutional network. Select 10 features of training data, which are filtered out from the datasets, and use the Huber function as a loos function.

Then we employed a modified version of the DQN method, substituting the convolutional neural network with the LSTM, a recurrent neural network with gating mechanisms for managing input flow.

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i),$$

$$f_t = \sigma(w_f[h_{t-1}, x_t] + b_f),$$

$$o_t = \sigma(w_o[h_{t-1}, x_t] + b_o),$$

i_t , f_t , o_t , and b_x represents input gate, forget gate, output gate randomly, and h_{t-1} represents the output of the previous LSTM block at timestamp t-1, and b_x represents biases for the respective gates(x).

Deep Q-networks are limited in that they can only learn a mapping from a small number of previous states; hence, if the user has a sequence of actions, DQN will not be able to recommend more precisely. When there are very lengthy time lags of uncertain size between key events, LSTM, which retains information for extended time periods, is well-suited to learn from experience to categorize, analyse, and predict time series. The combination can compensate for DQN's shortcomings, according to the researchers, who believe that by leveraging developments in LSTM, DQN can be improved to solve time series data more effectively. In the DQN algorithm, the LSTM is used to estimate the state-action value function. Variable padding is needed for observations before feeding them to the convolutional network since the amount of observations and the length of

time intervals varies. The following is a summary of the model , as shown in Figure 2.

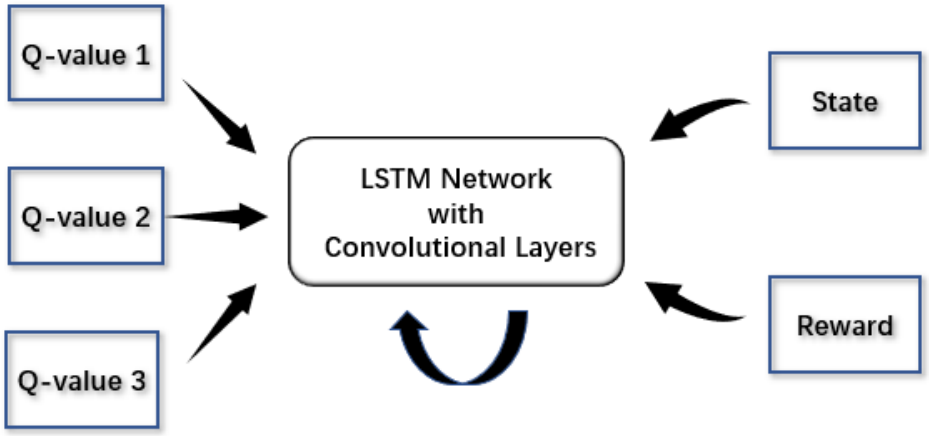

Figure 2. LSTM-based DQN

During each episode, the exploration rate drops from 0.9 to 0.1. Choose the Huber function as the loss function because it has a lower sensitivity to outliers than the mean-squares error and has been utilized in robust regression.

IV. Result

To compare the original DQN performance and the LSTM-based DQN, the figure of user-buy rate versus the number of episodes is shown below. According to Figure 3 and Figure 4, the LSTM-based DQN shows larger action spaces, stability, and lower variance.

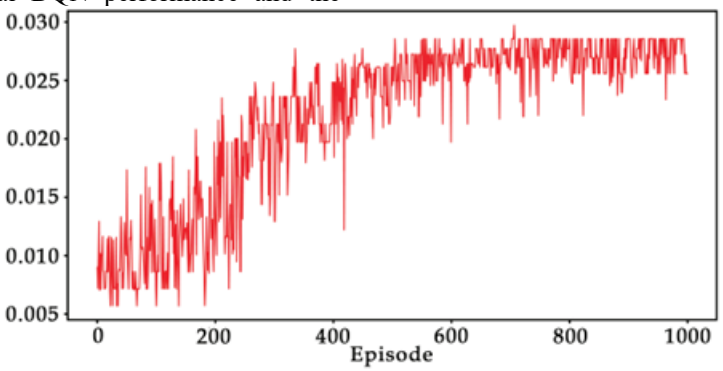

Figure 3. DQN performance

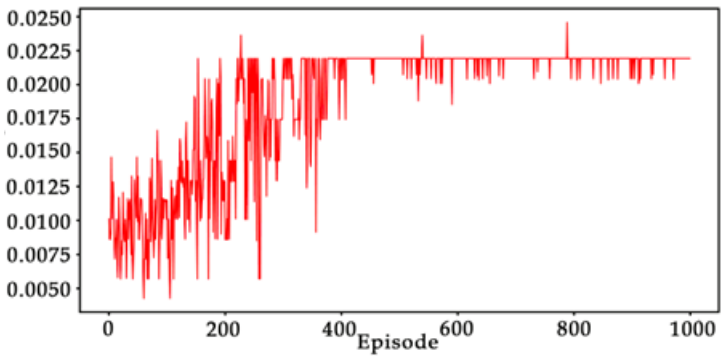

Figure 4. LSTM-based DQN performance

Randomly choose different datasets and data sizes by using LSTM-based DQN to do further testing. We discovered that LSTM-based DQN improved the DQN algorithm's convergence properties through testing. Compared with our experimental purpose, although LSTM-based DQN predicts more of the shortcomings of DQN, which may reduce the processing speed, after strengthening hardware or software facilities, it could influence the final outcomes scantly. The results are shown in Table I below, the value of recall, precision, or F1 score applied to different data sets or data volumes, the results of each test tend to be stable.

TABLE I LSTM-BASED DQN TESTING RESULT

Test	Precision	Recall	F1 score	MAP
#1	0.045	0.3158	0.0678	0.1254
#2	0.049	0.3876	0.0587	0.1786
#3	0.037	0.2989	0.0702	0.1359
#4	0.048	0.3087	0.0654	0.1345

V. CONCLUSION

The product suggestion technique is viewed as a series of interactions between customers and the recommender agent in this article, which uses Alibaba data. Such a method not only continuously iterates user feedback to get more accurate recommendations but also improves user experience during website use. The paper further compares the DQN model and LSTM-based DQN model and finds that the LSTM-based DQN model has better performance. The idea would be to overcome the challenge of user experience within the recommendation of products and improve the accuracy of recommendation even further.

REFERENCES

[1] Matt Payne, "Recommender Systems For Business - A Gentle Introduction", url: https://www.scalr.ai/post/recommender-systems-recommendation-systems

[2] Jie Bao, Yu Zheng, David Wilkie, and Mohamed Mokbel. 2015. Recommendations in location-based social networks: a survey. Geoinformatica 19, 3 (2015), 525–565.

[3] X. Zhao, L. Zhang, Z. Ding, D. Yin, Y. Zhao, and J. Tang, "Deep reinforcement learning for list-wise recommendations," arXiv preprint arXiv:1801.00209, 2017.

[4] Błażej Osiński and Konrad Budek, "What is reinforcement learning? The complete guide", url: https://deepsense.ai/what-is-reinforcement-learning-the-complete-guide/, Accessed: Oct 2021

[5] Joseph M. Carew, "reinforcement learning", url: https://searchenterpriseai.techtarget.com/definition/reinforcement-learning, Accessed: Oct 2021

[6] Silviu Pitis, "Rethinking the Discount Factor in Reinforcement Learning: A Decision Theoretic Approach", arXiv:1902.02893v1, [cs.LG] 8 Feb 2019.

[7] Ankit Choudhary, "A Hands-On Introduction to Deep Q-Learning using OpenAI Gym in Python", url: https://www.analyticsvidhya.com/blog/2019/04/introduction-deep-q-learning-python/, Accessed: Oct 2021

[8] DQN, deepmind.com, url: https://deepmind.com/research/dqn/, Accessed: Oct 2021

[9] Alibaba cloud, "User Behaviour Dataset from Alibaba", url: https://tianchi.aliyun.com/dataset/dataDetail?dataId=81505#1, Accessed: Oct 2021