applied sciences

MDPI

Article

DRL-SRS: A Deep Reinforcement Learning Approach for Optimizing Spaced Repetition Scheduling

Qinfeng Xiao [1] and Jing Wang [2,3,*]

[1] School of Fashion and Textiles, Hong Kong Polytechnic University, Hong Kong, China; qin-feng.xiao@connect.polyu.hk
[2] School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China
[3] Engineering Research Center of Integration and Application of Digital Learning Technology, Ministry of Education, Beijing 100039, China
* Correspondence: wj@bjtu.edu.cn

Abstract: Optimizing spaced repetition schedules is of great importance for enhancing long-term memory retention in both real-world applications, e.g., online learning platforms, and academic applications, e.g., cognitive science. Traditional methods tackle this problem by employing handcrafted rules while modern methods try to optimize scheduling using deep reinforcement learning (DRL). Existing DRL-based approaches model the problem by selecting the optimal next item to appear, which implies the learner can only learn one item in a day. However, the most essential point to enhancing long-term memory is to select the optimal interval to review. To this end, we present a novel approach to DRL to optimize spaced repetition scheduling. The contribution of our framework is three-fold. We first introduce a Transformer-based model to estimate the recall probability of a learning item accurately, which encodes the temporal dynamics of a learner's learning trajectories. Second, we build a simulation environment based on our recall probability estimation model. Third, we utilize the Deep Q-Network (DQN) as the agent to learn the optimal review intervals for learning items and train the policy in a recurrent manner. Experimental results demonstrate that our framework achieves state-of-the-art performance against competing methods. Our method achieves an MAE (mean average error) score of 0.0274 on a memory prediction task, which is 11% lower than the second-best method. For spaced repetition scheduling, our method achieves mean recall probabilities of 0.92, 0.942, and 0.372 in three different environments, the best performance in all scenarios.

Keywords: spaced repetition; deep reinforcement learning; memory models; half-life regression; transformers

Citation: Xiao, Q.; Wang, J. DRL-SRS: A Deep Reinforcement Learning Approach for Optimizing Spaced Repetition Scheduling. *Appl. Sci.* **2024**, *14*, 5591. https://doi.org/10.3390/app14135591

Academic Editor: Douglas O'Shaughnessy

Received: 17 May 2024
Revised: 20 June 2024
Accepted: 26 June 2024
Published: 27 June 2024

1. Introduction

The study of effective learning for enhancing long-term memory has been a focal point in both the educational and academic fields. One such strategy, spaced repetition, has been proven to effectively enhance long-term memory by strategically scheduling review intervals [1,2]. Quantitative experiments conducted by Ebbinghaus [3] illustrate that human memory decays with a delay since the last review of the content but can immediately be reinforced by reviewing it. Traditional approaches have explored applying rule-based strategies for spaced repetition scheduling, such as the Pimsleur [4] and Leitner [5] systems. These strategies are intuitive to understand but lack the flexibility to adopt individual learning patterns. In real-world applications, spaced repetition scheduling is currently widely used in practice, especially in online learning platforms, such as SuperMemo [6], Anki [7], Duolingo [8], and MaiMemo [9].

With the large number of online learners on e-learning platforms, it is feasible to collect large-scale training data [9,10]. In addition, with the development of deep learning, the field has witnessed a paradigm shift towards data-driven methods, particularly deep reinforcement learning (DRL) [11–13]. Since review scheduling of spaced repetition can

be viewed as a sequential decision-making process, it is natural to adopt DRL for spaced repetition optimization [14–17]. Reddy's study is a pioneer work that formulates instruction as a partially observable Markov decision process (POMDP) planning problem [14]. TADS [17] employs a time-interval prediction module and Dyna-style planning [18] for spaced repetition optimization. However, these methods can suffer several drawbacks in real-world applications. Firstly, the simulation environments used in these works are ruled-based without real-world data, which may deviate from real-world situations. Secondly, previous works formulated the reinforcement learning problem as deciding the next item, which implies only one item will be learned in each learning session. However, in realistic situations, the learning process contains multiple days while a number of items should be learned in a day. The main purpose of the spaced repetition algorithm is to determine the optimized interval to review an item.

To address the above challenges, we propose the DRL-SRS framework, which stands for the deep reinforcement learning approach for optimizing spaced repetition scheduling framework. The framework contains two modules. Our work first introduces the Transformer-based Half-Life Regression (THLR) model, which predicts the half-life of learning items with greater accuracy than traditional models based on real-world data. By capturing the temporal dynamics of recall likelihood, THLR provides a more nuanced understanding of the process of forgetting. Furthermore, following the problem definition of SSP-MMC [9], we develop a simulation environment that emulates the learner's daily review process, considering both inter-day and intra-day time dynamics. Although previous works have tried simulating student interactions for educational systems [19], they are not feasible for spaced repetition optimization. Lastly, we propose a deep reinforcement learning model for spaced repetition optimization based on a Deep Q-Network (DQN) augmented with a Long Short-term Memory (LSTM) mechanism.

Through extensive experiments, we demonstrate that our proposed framework significantly improves upon existing spaced repetition optimization methods. The results indicate that our approach not only enhances recall probabilities but also optimizes the scheduling process, leading to more efficient use of the learner's time and cognitive resources.

The rest of this paper is organized as follows: Section 2 reviews related work in spaced repetition optimization and reinforcement learning. Section 3 discusses the background knowledge like models for human memory used in our framework and basics of reinforcement learning. Section 4 details our methodology, including problem formulation, THLR model, simulation environment, and the DRL-based spaced repetition optimization. Section 5 presents our experimental results and analysis. Finally, Section 6 concludes the paper and discusses future research directions. The main innovation and features of our proposed framework against competing methods are summarized in Tables 1 and 2.

Table 1. Baseline methods in this paper for memory prediction. A comparison of integrated features is summarized in this table. ✓ denotes that the corresponding feature is equipped.

	Learning-Based	Temporal Dynamics	Attention Mechanism
Pimsleur [4]	-	-	-
Leitner [5]	-	-	-
HLR [8]	✓	-	-
DHP [9]	✓	-	-
GRU-HLR [10]	✓	✓	-
THLR (ours)	✓	✓	✓

Table 2. Competing methods in this paper for spaced repetition scheduling. A comparison of innovation and features is summarized in this table. ✓ denotes that the corresponding feature is equipped.

	Learning-Based	Temporal Dynamics	Reinforcement Learning
RANDOM	-	-	-
ANKI [7]	-	-	-
MEMORIZE [20]	-	-	-
HLR [8]	✓	-	-
Ours	✓	✓	✓

2. Related Work

Optimizing spaced repetition schedules [8] for enhancing long-term memory has been an important area of extensive research. Previous approaches to spaced repetition optimization can be divided into two categories, traditional approaches and recent reinforcement-learning approaches.

2.1. Traditional Spaced Repetition Algorithms

Prior to the introduction of deep learning paradigms, most scheduling algorithms were rule-based, leveraging heuristics derived from cognitive psychology and behavioral studies. Early works like Pimsleur [4] and Leitner [5] used simple rules, prioritizing items based on their novelty and difficulty. These methods are intuitive but lack the flexibility for individual memory patterns. SuperMemo [6] and Anki [7] are more sophisticated approaches that employ handcrafted rules to decide the next review interval and priorities of items within a learning session. Other methods such as the greedy algorithm [21] optimize the scheduling by selecting the item that maximizes the conditional marginal gain. MEMORIZE [20] uses stochastic optimal control to make a trade-off between recall likelihood and review times. However, these traditional approaches rely on the reliance of predefined rules and cannot adapt to the dynamic learning process of individual learners.

2.2. Reinforcement Learning for Spaced Repetition

The advent of deep reinforcement learning has marked a significant shift in the optimization of spaced repetition schedules. DRL-based methods model the student as an environment and use agents to learn optimal policies through interactions. These methods have the advantage of learning from data and can adapt to individual learners' behaviors and performance.

One of the pioneering works in this area is the application of DRL to spaced repetition by Reddy [14], who formulated the review scheduling problem as a Partially Observable Markov Decision Process and used Trust Region Policy Optimization (TRPO) to learn an optimal policy. Sinha [15] introduced a method that uses a Recurrent Neural Network (RNN) to model the student's memory mechanism, providing pseudo rewards at each learning event to guide the policy learning process. This approach improved upon the assumption of constant intervals by incorporating the student's memory state into the decision-making process. Upadhyay [16] proposed a deep reinforcement learning framework that simultaneously decides the time intervals of reviews and the content for the students, treating the review process as a Marked Temporal Point Process (MTPP). This method, while innovative, assumes that the student is available for learning at the exact times determined by the policy, which may not align with the student's availability. The Time-Aware scheduler with Dyna-Style planning (TADS) [17] approach by Yang et al. addresses some of these limitations by incorporating varying time intervals between learning events and using Time-LSTM networks to learn an optimal content selection policy. TADS integrates Dyna-style planning to improve sample efficiency, leveraging both model-based and model-free reinforcement learning techniques.

Deep-learning-based approaches, particularly those utilizing reinforcement learning, have shown promise in creating more flexible and personalized spaced repetition schedules. These methods can learn from individual learning patterns, although they come with challenges such as naive simulation environments, improper problem definition, and the use of ineffective DRL algorithms.

3. Background

3.1. Models for Human Memory

3.1.1. Exponential Forgetting Curve

The Exponential Forgetting Curve (EFC) [3] is one of the most well-known memory models that depicts the forgetting curve along time elapsed. It states that an item has the highest recall likelihood when it has been reviewed and it is forgotten at an exponential rate. In this paper, we use the equation proposed in [22], which states the recall probability of a learning item as follows:

$$p_{\text{recall}} = e^{-\theta \cdot \Delta / S}, \tag{1}$$

where θ, Δ, and S denote the item difficulty, the time elapsed since the last review, and the student's memory strength of the item, respectively.

3.1.2. Half-Life Regression

Settle and Meeder [8] define half-life regression (HLR) as follows:

$$p_{\text{recall}} = 2^{-\Delta / h}, \tag{2}$$

where p, Δ, and h denote the recall likelihood, the elapsed time since the last review, and the half-life parameter, respectively. In practice, the ground-truth half-life h is estimated using a dedicated model. The estimated half-life parameter $\hat{h}_\Theta$ is calculated in [8] by:

$$\hat{h}_\Theta = 2^{\Theta \cdot \mathbf{x}}, \tag{3}$$

where $x = (x_\oplus, x_\ominus, \text{lex})$ is the feature vector, composed of the correct recall times, incorrect recall times, and the lexeme tag, separately. The HLR model is optimized using the following objective:

$$J_{\text{HLR}} = (p - \hat{p})^2 + \alpha (h - \hat{h})^2 + \lambda \parallel \Theta \parallel^2, \tag{4}$$

where p and $\hat{p}$ are recall probability and predicted recall probability, respectively.

3.1.3. Difficulty–Half-life–P(recall) HLR

The Difficulty–Half-life–P(recall) (DHP) HLR model is proposed in [9] to enhance the original HLR model. By taking recall probabilities, recall results, and difficulties of an item into consideration, DHP models the human memory with a Markov property. The state transition equation is defined as:

$$\begin{bmatrix} h_{i+1} \\ d_{i+1} \end{bmatrix} = \begin{bmatrix} h_i \cdot (e^{\theta_1 \cdot x_i} + 1) & e^{\theta_2 \cdot x_i} \\ d_i & d_i + \theta_3 \end{bmatrix} \cdot \begin{bmatrix} r_i \\ 1 - r_i \end{bmatrix}, \tag{5}$$

where h_i, r_i, d_i, $\{\theta_1, \theta_2, \theta_3\}$ and x_i indicate the half-life, the recall result, the difficult, model parameters, and the collection of states of the item at the i-th timestamp, respectively.

3.2. Reinforcement Learning

Conventional reinforcement learning (RL) algorithms [23] contain an environment, an agent (policy), and reward functions. The agent (or policy) π learns to behave in such a way as to maximize a notion of cumulative reward. The learning process is based on the agent's experiences, which consist of states $\mathcal{S}$, actions $\mathcal{A}$, and rewards $\mathcal{R}$. In general, our goal is to find the best policy $\pi^*(s)$ such that getting the best discounted cumulative reward $\mathcal{R} = \sum_{t=t_0}^{\infty} \gamma^{t-t_0} r_t$, where γ is the discounted factor and r_t is the reward at step t.

4. Methodology

As illustrated in Figure 1, our framework consists of three components, the THLR memory prediction module, a simulation environment, and a DRL-based spaced repetition algorithm. The THLR memory prediction module is used for estimating the recall probability of learning items, taking into account the probability history, recall history, and interval history. The simulation environment mimics a learner's daily review process, considering both inter-day and intra-day dynamics. The DRL-based spaced repetition optimization utilizes a DQN with an LSTM mechanism to learn the optimal review intervals for effective long-term memory retention. The problem formulation and detailed descriptions of the three components are given below.

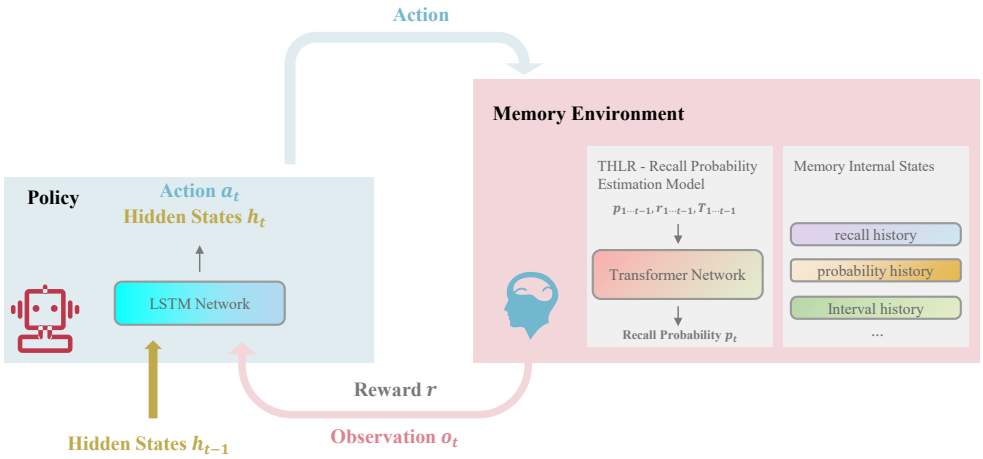

Figure 1. The overall view of our proposed framework for spaced repetition optimization using deep reinforcement learning. The arrow represents the computation flow.

4.1. Problem Formulation

The goal of spaced repetition optimization is to schedule learning items such that the learner memorizes all items in the long-term memory, meanwhile, minimizing memory costs. In detail, given N learning items, the whole learning process consists of a learning event sequence in which a learning event e is represented as a vector $e_i = (w_i, \Delta_i, r_i, p_i)$. For an incoming item w_i, which has not been reviewed for Δ_i days, the probability of the learner recalling the item successfully is equal to p_i. The optimization algorithm has to determine the optimal interval for the item after the recalling result r_i. In this paper, we formulate this problem into a reinforcement learning paradigm. We define each RL component as follows:

- State space. The state space $\mathcal{S}$ depends on the memory model. For EFC, the state space encodes the item difficulty, the time delay, and the memory strength. For DHP, the states include the difficulty, the time delay, and the recall result.
- Action space. The action space $\mathcal{A} = \{1, 2, \cdots, T\}$ consists of review intervals ranging from 1 to the maximum interval T. The action $a_t \in \mathcal{A}$ indicates that the current item was scheduled a_t days ago after its review at time t.
- Observation space. Away from the internal memory states, the agent can only obtain observations $\mathcal{O}$ including the time delay, the recall result, and the recall probability no matter the choice of memory model.
- Reward function. Following previous works [14,15,17], we construct the reward function using the recall probability of the current item, which is provided by the memory model $\mathcal{M}$.

To this end, our goal is to find the optimal policy performing the best action (optimal interval) to gain maximum rewards.

4.2. Transformer-Based Half-Life Regression

The first component of our framework is to calculate the half-life for a given item effectively. Unlike previous half-life regression methods [8–10,24], we adopt Transformer [25] to predict the half-life of given items, since Transformer has been proven to be effective for time-series prediction [26]. To this end, we propose Transformer-based Half-life Regression. The use of a Transformer for half-life regression is based on several considerations. Firstly, as with HLR [8], the Transformer can effectively capture temporal dynamics. Secondly, rather than manually designing the state transition like DHP [9], THLR can flexibly adopt various memory patterns. Finally, the Transformer achieves better performance than other recurrent networks, such as gated recurrent unit (GRU) networks [27], in experimental results. At the same time, we do not consider more complex Transformer variants since the features used in our task are relatively simple.

Given $\{r_{1:t-1}, p_{1:t-1}, \Delta_{1:t-1}\}$, the predicted half-life is calculated by:

$$\hat{h}_t = \text{Transformer}(r_{1:t-1}, p_{1:t-1}, \Delta_{1:t-1}), \tag{6}$$

where the following factors are considered in our model:

- Previous recall results $r_{1:t-1}$;
- Previous recall probabilities $p_{1:t-1}$;
- Previous review intervals $\Delta_{1:t-1}$.

Based on the estimated half-life, we can calculate the recall probability of the item according to [8]:

$$p_t = 2^{-\frac{\Delta_t}{\hat{h}_t}}. \tag{7}$$

4.3. Schedule Optimization Environment

The second component of our framework is the simulation environment. The environment is based on the memory model trained in Section 4.2 or other baselines such as EFC and HLR. Unlike previous works [14,17], the environment proposed in this paper contains two phases, inter-day and intra-day. In the intra-day phase, the environment simulates the learner to respond for items due today one by one, until the daily time limit $cost_{limit}$ is reached. Successful recall and unsuccessful recall cost $cost_{r_t=1}$ and $cost_{r_t=0}$, respectively. For a new item (never reviewed before), the item will be arranged for review the next day and the time cost is $cost_{new}$. When the daily cost is exhausted, the remaining items are postponed to the next day, and the date day is processed to the next day $day = day + 1$.

The detailed simulation process of our environment is shown in Algorithm 1. I_t represents the current item. The properties, *due_day*, *last_day*, *type*, *attempts*, *intervals*, *recalls*, and *probabilities* of I_t represent newest day to review, last day of review, item type, number of attempts, interval history, recall result history, and recall probability history of the item, respectively. $\mathcal{M}$ indicates the memory model used in the environment. The method $\mathcal{M}.get()$ fetches the next item from the memory. $\mathcal{M}.prob(I_t)$ calculates the recall probability of the current item I_t. The method $\mathcal{M}.put(I_t)$ puts the item with updated states back into the memory.

Algorithm 1: The simulation process of the environment

Data: *Action* a_t, *Memory* $\mathcal{M}$
Result: *Reward* r, *Observation* o_t, *Termination*
Initialize the environment parameters;
while *not terminated* **do**
  Get the current item due today: $I_t = \mathcal{M}.get()$ with $I_t.due\_day == day$;
  if $I_t.type$ == *REVIEW* **then**
    Calculate recall probability $p_t = \mathcal{M}.prob(I_t)$; `/* The item type is review item */`
    $I_t.attempts \leftarrow I_t.attempts + 1$;
    $I_t.intervals$.append(a_t); `/* Record interval history */`
    if *random*$() < p_t$ **then**
      $r_t \leftarrow 1$; `/* Successful recall */`
      $cost \leftarrow cost + cost_{r=1}$;
    else
      $r_t \leftarrow 0$Unsuccessful recall $cost \leftarrow cost + cost_{r=0}$;
    Update interval states of Memory $\mathcal{M}$;
    $I_t.recalls$.append(r_t); `/* Record recall history */`
    $I_t.probabilities$.append(p_t); `/* Record probability history */`
  else
    $cost \leftarrow cost + cost_{\text{new}}$; `/* The item type is new item */`
    $I_t.type \leftarrow$ REVIEW;
  $I_t.due\_day \leftarrow I_t.due\_day + a_t$; `/* Set the next day to review */`
  $I_t.last\_day \leftarrow day$;
  Put the item back to review $\mathcal{M}.put(I_t)$;
  $r \leftarrow p_t$; `/* Reward is equal to recall probability */`
  if $cost \geq cost_{limit}$ **then**
    $cost \leftarrow 0$;
    next_day$()$; `/* The learning quota of today is running out */`
  if *no next item or session ends* **then**
    Terminate the session; Set reward to 0;
  Get observation and other information;
Close the environment;

4.4. Reinforcement-Learning-Based Spaced Repetition Optimization

The third component of our framework is the RL-based spaced repetition policy. In general, we can use any off-the-shelf RL algorithm. In this paper, we assume that the policy cannot access the internal states of the learner except the observations. In addition, the action space is discrete for our environment. Based on the above considerations, we choose the model-free, off-policy Deep Q-Network [28] for the policy. In addition, to capture the temporal dynamics of memory, we adopt Long Short-term Memory [29] for the policy network and train the policy using a recurrent style.

4.4.1. DQN Algorithm

The main idea behind DQN is that if we had a Q function $Q^* : \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ to compute the reward if the policy to take an action a in a given state s, to maximize the reward, we have the equation:

$$\pi^*(s) = \arg\max_a Q^*(s, a). \tag{8}$$

Since the optimal Q function is unknown, we use a neural network Q^π to approximate it. For each training step, we have:

$$Q^\pi(s,a) = r + \gamma Q^\pi(s', \pi(s')), \tag{9}$$

where r is the step reward, and s' is the previous state. The difference between two consecutive steps is the temporal difference error δ:

$$\delta = Q(s,a) - (r + \max_a Q(s',a)). \tag{10}$$

To minimize the temporal difference error, DQN optimizes the Huber loss $\mathcal{L}$ over a batch of data sampled from the replay buffer:

$$\mathcal{L} = \frac{1}{|B|} \sum_{(s,a,s',r)\in B} \mathcal{L}(\delta), \tag{11}$$

where:

$$\mathcal{L}(\delta) = \begin{cases} \frac{1}{2}\delta^2 & \text{for} |\delta| \leq 1 \\ |\delta| - \frac{1}{2} & \text{otherwise} \end{cases}. \tag{12}$$

4.4.2. Recurrent-Style Planning

Previous works use features of current time while ignoring the temporal relations [14]. Considering the importance of temporal relations between learning events, we adopt LSTM for the policy network and train the policy using a recurrent style [30]. The policy $\pi(\cdot,\cdot)$ takes not only the current state but also hidden features that encode previous states as inputs.

5. Experiments

In the experiment section, we assess the performance of our framework according to two aspects: memory prediction and schedule optimization. The goal of memory prediction assessment is to evaluate the accuracy of recall probability prediction. The goal of schedule optimization assessment is to assess the effectiveness of the scheduling.

5.1. Environments

The simulation environment is implemented using OpenAI Gymnasium (v1.0.0 alpha1) [31]. The spaced repetition policy is implemented by Tianshou (v0.5.0) [32] and PyTorch (v2.2) [33]. All the experiments are performed on a computer with an RTX4090 GPU.

5.2. Memory Prediction

5.2.1. Baselines

We compared THLR with the following baselines:

- Pimsleur [4] is a pioneer work in introducing an initial scheduling system that utilized a geometric progression, characterized by a common ratio of five.
- Leitner [5] is a method using tangible containers, which manages the frequency of flashcard reviews by transferring them between boxes of differing dimensions.
- HLR [8] proposes a parameter to measure the storage strength of memory and gives the recall probability according to the half-life and the time interval.
- DHP-HLR [9] is a variant of HLR that considers the item difficulty.
- GRU-HLR [10] is an improved version of DHP-HLR that uses recurrent neural networks to update internal parameters.

5.2.2. Evaluation Metrics and Experimental Settings

We use mean absolute error (MAE) and mean absolute percentage error (MAPE) to evaluate the performance of our model and baselines. In this experiment, we use the dataset collected by [9], which contains 220 million review event logs from the online e-learning platform MaiMemo, processed into 71,697 groups.

5.2.3. Results and Analysis

Table 3 shows the experimental results for memory prediction. We can see that our method outperforms all the baselines in predicting recall probabilities with the lowest MAE and MAPE, which illustrates that THLR has remarkable effectiveness in predicting recall probability. To show the significance of our method, we also conduct a Wilcoxon Signed-Rank Test [34] over baselines w.r.t. the two measurements. The p-value is 0.03125, which shows that we should reject the H0 assumption (the algorithms are equal) at the 95% confidence level.

Table 3. Experimental results for memory prediction. The bold item indicates the best performance.

	MAE	**MAPE**
Pimsleur	0.3169	165.69%
Leitner	0.4535	133.92%
HLR	0.1070	76.65%
DHP	0.0779	46.35%
GRU-HLR	0.0307	18.88%
THLR (ours)	**0.0274**	**16.70%**

Among all baselines, Pimsleur and Leitner are traditional, rule-based models that do not adapt to individual learning patterns, resulting in the highest MAE and MAPE values. This verifies that it is difficult to learn a good memory prediction model using handcrafted rules from real-world data. HLR introduces a parameter to measure memory strength and provides recall probabilities based on the half-life and time interval, which improves the prediction accuracy but still falls short compared to more sophisticated models. DHP and GRU-HLR are more advanced models that consider more environment variables. DHP takes half-life, recall probability, recall result, and item difficulty into consideration and utilizes handcrafted transition equations. GRU-HLR utilizes recurrent neural networks to update internal parameters, which improves the flexibility compared to DHP. Thus, GRU-HLR shows a slightly better result compared to DHP. The two models both show significant improvements over the traditional models.

THLR, the proposed model in this paper, outperforms all baselines by a considerable margin. The use of the Transformer network allows the effective capture of temporal dynamics in learning trajectories, which is crucial for accurate recall probability estimation. The results suggest that the THLR model's ability to flexibly adapt to various memory patterns and its effectiveness in time-series prediction contribute to its enhanced performance. The lower MAPE and MAPE values signify that the THLR model is better at estimating recall probabilities.

5.3. Schedule Optimization

5.3.1. Baselines

We compared our framework with the following baselines:

- RANDOM is a baseline that chooses a random interval from $[1, \text{halflife}]$ to schedule the next review.
- ANKI [7] is a variant of the SM-2 algorithm, which is used in a popular learning application.
- MEMORIZE [20] is a spaced repetition algorithm based on optimal control that is trained to determine the parameter to minimize the expectation of review cost.

- HLR [8] is a baseline to schedule the next review interval equal to the half-life.

To test the effectiveness of spaced repetition algorithms in different environments, we conduct experiments on the following memory models:

- EFC [3] is one of the most recognized memory models that illustrates the relationship between the forgetting curve and the time decay.
- DHP [9] is a memory model with the Markov property for explainability and simplicity and handcrafted state-transition equations. It takes half-life, recall probability, recall result, and item difficulty as state variables.
- THLR is the simulation environment proposed in this work.

5.3.2. Evaluation Metrics and Experimental Settings

As in [14,17], we evaluate the performance by using the average recall probability (ARP) over all items after the last learning event.

We set the limit of daily learning cost as 600 and the number of items as 1000. The learning cost of a new item, a recalled review item, and a forgotten review item are 12, 3, and 9, respectively. The maximum duration of the simulation is set as 1000 days.

5.3.3. Results and Analysis

Table 4 shows the experimental results for spaced repetition optimization, where the columns represent different methods and the rows represent different simulation environments.

Table 4. Experimental results for spaced repetition optimization. The bold item indicates the best performance.

	EFC		**DHP**		**THLR**	
	SUM	**MEAN**	**SUM**	**MEAN**	**SUM**	**MEAN**
RANDOM	52,439.40	0.867	28,803.55	0.475	13,482.70	0.178
ANKI	13,264.31	0.637	13,702.83	0.584	12,425.18	0.185
MEMORIZE	-	-	35,350.10	0.817	12,694.96	0.169
HLR	-	-	10,516.85	0.468	25,169.26	0.355
Ours	88,548.67	**0.920**	162,482.57	**0.942**	18,572.39	**0.372**

In general, the results indicate that the proposed framework (referred to as "Ours") outperforms all baselines in terms of ARP across the three memory models, suggesting that it effectively optimizes the scheduling of review intervals to enhance long-term memory retention. To investigate the significance of our proposed method, we perform a Friedman test [35] w.r.t. different simulation environments over baseline methods. The p-value is 0.0094, which suggests that we reject the H0 assumption (the algorithms are equal) at the 0.99% confidence level, i.e., our method significantly outperforms the baselines.

For the EFC environment, MEMORIZE and HLR cannot be applied in this environment because they require the half-life parameter, which is not contained in the environment. ANKI shows a worse result than RANDOM. Our method achieves a moderate improvement compared to RANDOM and ANKI. For the DHP environment, ANKI shows moderate performance and is slightly better than RANDOM. MEMORIZE shows better performance than the other baseline methods. However, HLR's performance is worse than that of RANDOM. For the THLR environment, ANKI and HLR show better results than RANDOM. However, the performance of MEMORIZE is even worse than that of RANDOM.

Across all environments, ANKI shows a moderate performance. It performs slightly better than RANDOM in DHP and THLR but does not adapt as effectively to the EFC environment. MEMORIZE illustrates a significant improvement compared with RANDOM in DHP but fails in THLR. HLR shows a worse performance than that of RANDOM in DHP but gets a better result in THLR. The experimental results illustrate that none of the baselines show consistent performance across all environments. In contrast, our proposed

framework beats all competing methods over all environments. The experimental results show our framework's superior performance for spaced repetition optimization utilizing Deep Q-Networks augmented with a Long Short-term Memory mechanism for learning optimal review intervals.

6. Conclusions

This paper introduces a novel deep reinforcement learning framework, DRL-SRS, aimed at optimizing spaced repetition scheduling to enhance long-term memory retention. This framework is particularly relevant for applications in online learning platforms and cognitive science research, where the efficiency of memory retention is paramount. Our approach addresses the limitations of traditional rule-based methods and previous DRL-based models by introducing three key innovations. Firstly, we developed a THLR model to more accurately estimate the recall probability of learning items, capturing the temporal dynamics of a learner's memory trajectory. Secondly, we constructed a simulation environment that emulates a learner's daily review process, considering both inter-day and intra-day time dynamics. Thirdly, we employed a DQN augmented with an LSTM mechanism to learn the optimal review intervals for learning items. For future works, there are two valuable directions to explore. First, incorporating personal features to enhance personalized experiences for learners, in that, the scheduler can be further refined with personalized data to output individually optimized review intervals, e.g., relaxed scheduling for learners who are performing well. Second, incorporating multi-modal learning data, such as visual, auditory, and kinesthetic inputs, can improve the accuracy of describing the difficulty of learning items. However, most current research works only consider language learning, i.e., text inputs.

Author Contributions: Conceptualization, Q.X. and J.W.; methodology, Q.X.; software, Q.X.; validation, Q.X. and J.W.; formal analysis, J.W.; investigation, J.W.; resources, J.W.; data curation, J.W.; writing—original draft preparation, Q.X.; writing—review and editing, J.W.; visualization, Q.X.; supervision, J.W.; project administration, J.W.; funding acquisition, J.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Innovation Fund of the Engineering Research Center of the Ministry of Education for the Integration and Application of Digital Learning Technology (grant number 1221031).

Data Availability Statement: The original data presented in the study are openly available via the Harvard Dataverse at https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/VAGUL0 (accessed on 19 February 2024).

Acknowledgments: We are immensely grateful to everyone who has supported us during this project.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Cepeda, N.J.; Vul, E.; Rohrer, D.; Wixted, J.T.; Pashler, H. Spacing effects in learning: A temporal ridgeline of optimal retention. *Psychol. Sci.* **2008**, *19*, 1095–1102. [CrossRef] [PubMed]
2. Finke, L. *Markov Models for Spaced Repetition Learning*; Mathematisches Institut der Georg-August-Universität Göttingen: Göttingen, Germany, 2023.
3. Ebbinghaus, H. Memory: A contribution to experimental psychology. *Ann. Neurosci.* **2013**, *20*, 155. [CrossRef] [PubMed]
4. Pimsleur, P. A memory schedule. *Mod. Lang. J.* **1967**, *51*, 73–75. [CrossRef]
5. Leitner, S.; Totter, R. *So Lernt Man Lernen*; Angewandte Lernpsychologie ein Weg zum Erfolg; Herder: Freiburg, Germany, 1972.
6. Woźniak, P.; Gorzelańczyk, E. Optimization of repetition spacing in the practice of learning. *Acta Neurobiol. Exp.* **1994**, *54*, 59–62. [CrossRef]
7. Lu, M.; Farhat, J.H.; Beck Dallaghan, G.L. Enhanced learning and retention of medical knowledge using the mobile flash card application Anki. *Med. Sci. Educ.* **2021**, *31*, 1975–1981. [CrossRef] [PubMed]
8. Settles, B.; Meeder, B. A trainable spaced repetition model for language learning. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Berlin, Germany, 7–12 August 2016; pp. 1848–1858.
9. Ye, J.; Su, J.; Cao, Y. A stochastic shortest path algorithm for optimizing spaced repetition scheduling. In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 14–18 August 2022; pp. 4381–4390.

10. Su, J.; Ye, J.; Nie, L.; Cao, Y.; Chen, Y. Optimizing spaced repetition schedule by capturing the dynamics of memory. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 10085–10097. [CrossRef]
11. Pinto, J.D.; Paquette, L. Deep Learning for Educational Data Science. *arXiv* **2024**, arXiv:2404.19675.
12. Taherisadr, M.; Stavroulakis, S.A.; Elmalaki, S. adaPARL: Adaptive privacy-aware reinforcement learning for sequential decision making human-in-the-loop systems. In Proceedings of the 8th ACM/IEEE Conference on Internet of Things Design and Implementation, San Antonio, TX, USA, 9–12 May 2023; pp. 262–274.
13. Gharbi, H.; Elaachak, L.; Fennan, A. Reinforcement Learning Algorithms and Their Applications in Education Field: A Systematic Review. In *Proceedings of the International Conference on Smart City Applications*; Springer: Berlin/Heidelberg, Germany, 2023; pp. 410–418.
14. Reddy, S.; Levine, S.; Dragan, A. Accelerating human learning with deep reinforcement learning. In Proceedings of the NIPS Workshop: Teaching Machines, Robots, and Humans, Long Beach, CA, USA, 9 December 2017.
15. Sinha, S. Sinha, S. Using Deep Reinforcement Learning for Personalizing Review Sessions on E-Learning Platforms with Spaced Repetition. Master's Thesis, KTH, School of Electrical Engineering and Computer Science (EECS), Stockholm, Sweeden, 2019.
16. Upadhyay, U.; De, A.; Gomez Rodriguez, M. Deep reinforcement learning of marked temporal point processes. In Proceedings of the 32nd Conference on Neural Information Processing Systems, Montréal, QC, Canada, 3–8 December 2018; p. 31.
17. Yang, Z.; Shen, J.; Liu, Y.; Yang, Y.; Zhang, W.; Yu, Y. TADS: Learning Time-Aware Scheduling Policy with Dyna-Style Planning for Spaced Repetition. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), New York, NY, USA, 25–30 July 2020; pp. 1917–1920. [CrossRef]
18. Sutton, R.S.; Szepesvari, C.; Geramifard, A.; Bowling, M.P. Dyna-Style Planning with Linear Function Approximation and Prioritized Sweeping. *arXiv* **2012**, arXiv:1206.3285 .
19. Zhao, G.; Huang, Z.; Zhuang, Y.; Liu, J.; Liu, Q.; Liu, Z.; Wu, J.; Chen, E. Simulating Student Interactions with Two-stage Imitation Learning for Intelligent Educational Systems. In Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, Birmingham, UK, 21–25 October 2023; pp. 3423–3432.
20. Tabibian, B.; Upadhyay, U.; De, A.; Zarezade, A.; Schölkopf, B.; Gomez-Rodriguez, M. Enhancing human learning via spaced repetition optimization. *Proc. Natl. Acad. Sci. USA* **2019**, *116*, 3988–3993. [CrossRef] [PubMed]
21. Hunziker, A.; Chen, Y.; Mac Aodha, O.; Gomez Rodriguez, M.; Krause, A.; Perona, P.; Yue, Y.; Singla, A. Teaching multiple concepts to a forgetful learner. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 4048–4058.
22. Reddy, S.; Labutov, I.; Banerjee, S.; Joachims, T. Unbounded human learning: Optimal scheduling for spaced repetition. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 1815–1824.
23. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 2018.
24. Pokrywka, J.; Biedalak, M.; Gralinski, F.; Biedalak, K. Modeling Spaced Repetition with LSTMs. In Proceedings of the 15th International Conference on Computer Supported Education CSEDU (2), Prague, Czech, 21–23 April 2023; pp. 88–95.
25. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*; Neural Information Processing Systems Foundation, Inc.: La Jolla, CA, USA, 2017; Volume 30.
26. Wen, Q.; Zhou, T.; Zhang, C.; Chen, W.; Ma, Z.; Yan, J.; Sun, L. Transformers in time series: A survey. *arXiv* **2022**, arXiv:2202.07125.
27. Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv* **2014**, arXiv:1406.1078.
28. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef] [PubMed]
29. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
30. Schäfer, A.M. Reinforcement Learning with Recurrent Neural Networks. Ph.D. Thesis, Osnabrück University, Osnabrück, Germany, 2008.
31. Towers, M.; Terry, J.K.; Kwiatkowski, A.; Balis, J.U.; Cola, G.d.; Deleu, T.; Goulão, M.; Kallinteris, A.; KG, A.; Krimmel, M.; et al. Gymnasium. 2023. Available online: https://github.com/Farama-Foundation/Gymnasium (accessed on 16 May 2024).
32. Weng, J.; Chen, H.; Yan, D.; You, K.; Duburcq, A.; Zhang, M.; Su, Y.; Su, H.; Zhu, J. Tianshou: A Highly Modularized Deep Reinforcement Learning Library. *J. Mach. Learn. Res.* **2022**, *23*, 1–6.
33. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. Pytorch: An imperative style, high-performance deep learning library. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 8026.
34. Wilcoxon, F. Individual comparisons by ranking methods. In *Breakthroughs in Statistics: Methodology and Distribution*; Springer: New York, NY, USA, 1992; pp. 196–202.
35. Demšar, J. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **2006**, *7*, 1–30.