Optimizing Spaced Repetition Schedule by Capturing the Dynamics of Memory

Jingyong Su, Junyao Ye, Liqiang Nie, *Senior Member, IEEE*, Yilong Cao, and Yongyong Chen

***Abstract*—Spaced repetition, namely, learners review items in a given schedule, has been proven powerful for memorization and practice of skills. Most current spaced repetition methods focus on either predicting student recall or designing an optimal review schedule, thus omitting the integrity of the spaced repetition system. In this work, we propose a novel spaced repetition schedule framework by capturing the dynamics of memory, which alternates memory prediction and schedule optimization to improve the efficiency of learners' reviews. First, the framework collects logs from students' reviews and builds memory models with Markov property to capture the dynamics of memory. Then, the spaced repetition optimization is transformed a stochastic shortest path problem and solved via the value iteration method. We also construct a new benchmark dataset for spaced repetition, which is the first to contain time-series information during learners' memorization. Experimental results on the collected data from the real world and the simulated environment demonstrate that the proposed approach reduces 64% error and 17% cost in predicting recall rates and optimizing schedules compared to several baselines. We have publicly released the dataset containing 220 million rows and codes used in this paper at: https://github.com/maimemo/SSP-MMC-Plus.**

***Index Terms*—Language learning, Markov decision process, recurrent neural networks, time-series features, spaced repetition.**

I. Introduction

MEMORY plays an important role in learning. To preserve long-term memory efficiently, students need to regularly review what they have learned over a lengthy period of time, a technique known as spaced repetition. The spacing effect and forgetting curve, which were identified in the basic memory experiment of [1], are the inspiration for spaced repetition. Researchers have worked extensively on optimizing spaced repetition to predict learners' memory and schedule efficient review tasks. There is an optimal review schedule, according to meta-analyses of review intervals in [2], [3]. Additionally, a number of studies verify that the students of medicine [4], statistics [5], history [6] all benefit significantly from the use of spaced repetition.

With more students studying online on e-learning platforms, it is feasible to collect extensive learning data. Based on that, many efforts have been dedicated to how to create intelligent tutoring systems via data mining and machine learning techniques [7], [8]. One of the system's main functions is to schedule learning and review tasks for learners. In particular, Settle and Meeder [9] develop a trainable memory prediction model that aids learners in deciding which skills require review. Rafferty et al. [10] formulate instruction as a partially observable Markov decision process (POMDP) planning problem, which is often used in sequential optimization [11]. Meanwhile, they explore the optimal strategies with deep reinforcement learning (DRL) methods, which are powerful for making sequential decision [12], [13]. These techniques increase learners' effectiveness and engagement, and therefore, are quite practical in real-world scenarios.

Previous research, however, has either focused on predicting learners' memory or designing optimal scheduling. The lack of optimal scheduling in predicting memory prevents it from improving the learners' efficiency directly; the lack of predicting memory in optimal scheduling makes it challenging to fit the real learners. Furthermore, the works of predicting memory [9], [14], [15] focus more on the statistical than the time-series features of learners' memory behaviors. Many samples that are noticeably different in time-series cannot be differentiated. Therefore, it hampers these memory models from correctly simulating learner memory (see "Related Work" at Section II-A). The accuracy of several algorithms used to schedule reviews based on such models is similarly constrained by the lack of time-series features. Additionally, the action spaces of previous DRL approaches [16], [17], [18] are rigid, making it difficult for students to supplement new learning stuff.

This paper investigates how to predict learners' memory based on time-series behavioral data during spaced repetition and uses it as a basis to build a scheduling algorithm to optimize the spaced repetition schedule. We collect a dataset containing time-series features of learners' memory behaviors and propose a novel framework for spaced repetition, as shown in Fig. 1, by processing memory prediction and schedule optimization alternately to improve the efficiency of learners' review. In terms of memory prediction, we establish DHP-HLR (Difficulty Halflife P(recall) Halflife-Regression) model inspired by the

Manuscript received 19 July 2022; revised 10 January 2023; accepted 21 February 2023. Date of publication 6 March 2023; date of current version 15 September 2023. This work was supported in part by the GuangDong Basic and Applied Basic Research Foundation under Grants 2022A1515010800 and 2022A1515010819, in part by the Shenzhen Science and Technology Innovation Program under Grants JCYJ20220818102414031 and RCBS20210609103708013, in part by the National Natural Science Foundation of China under Grant 62106063, and in part by the Shenzhen College Stability Support Plan under Grant GXWD20201230155427003-20200824113231001. Recommended for acceptance by B. He. *(Jingyong Su and Junyao Ye are co-first authors.) (Corresponding authors: Yilong Cao and Yongyong Chen.)*

Jingyong Su, Liqiang Nie, and Yongyong Chen are with the Harbin Institute of Technology, Shenzhen 518055, China (e-mail: sujingyong@hit.edu.cn; nieliqiang@hit.edu.cn; yongyongchen.cn@gmail.com).

Junyao Ye and Yilong Cao are with the MaiMemo Inc., Qingyuan, Guangdong 511500, China (e-mail: jy.ye@maimemo.com; yl.cao@maimemo.com).

Digital Object Identifier 10.1109/TKDE.2023.3251721

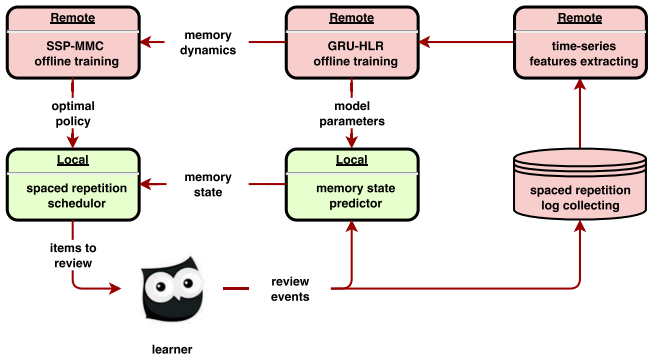


Fig. 1. The framework of our method is separated into two main parts: local in green and remote in red. Each time a user examines a word on the remote, the review events are locally recorded. After the learner has finished all of the day's reviews, the log is submitted to the spaced repetition log collection on the remote. Then, logs will be collectively processed to extract the time-series features for training the memory prediction model. Then the optimizing algorithm searches for the optimal spaced repetition schedule based on the dynamics of memory captured by the memory model. Periodically, the local will be updated with the optimal policy and model parameters. The local spaced repetition scheduler determines the next review date for words, while the local memory state predictor determines the memory state for each review.

two-component model of long-term memory [19] and GRU-HLR combining gated recurrent unit (GRU) [20] network and halflife regression (HLR) model [9]. Meanwhile, the state variables of DHP-HLR and the hidden layer variables of the GRU network capture the potential memory states and dynamics. We optimize the spaced repetition schedule based on memory models and a stochastic dynamic programming method to minimize learners' memory costs on each item. It is found that for predicting memory, the use of time-series features effectively reduces prediction error while capturing memory dynamics. For review schedule, the proposed scheduling algorithm SSP-MMC based on the time-series model, combined with stochastic dynamic programming to minimize the memory cost, outperforms other state-of-the-art methods. To summarize, the main contributions of this paper are:

- We propose a novel spaced repetition schedule framework by capturing the dynamics of memory, which associates memory prediction with optimal scheduling to improve the efficiency of learners' reviews.
- To the best of our knowledge, this is the first study to apply the time-series features to model long-term memory, making the model trained directly from the memory behaviors of learners.
- We build and publicly release our spaced repetition log dataset with 220 million rows, the first to include time-series data.

This paper is a substantial extension of our previous conference paper *A Stochastic Shortest Path Algorithm for Optimizing Spaced Repetition Scheduling* [21], where we proposed DHP-HLR model to predict memory and SSP-MMC algorithm to optimize the schedule. Compared with the conference version, we add GRU-HLR to capture memory dynamics and reduce memory prediction error while streamlining manual intervention in adjusting model parameters. The extended experiments compare more benchmark memory models and analyze the causes of error prediction. Furthermore, we discover that the proposed SSP-MMC algorithm is well compatible with the hidden layer state of GRU, which may be employed as state spaces in the Markov decision-making process.

The rest of this paper is organized as follows. A brief review of related work is reported in Section II. The proposed DHP-HLR and GRU-HLR are elaborated in Section III. The proposed SSP-MMC algorithm is presented in Section IV. Extensive experimental results and discussions are reported in Section V, and a conclusion is given in Section VI.

II. Related Work

Relevant prior work includes studies of memory models and optimizing schedules.

A. Human Memory Models

There are many studies on modeling human memory to improve teaching. Ebbinghaus [1] first proposed the forgetting curve to illustrate memory decay if no review. Anderson [14] proposed ACT-R theory, whose declarative memory module assumes that each review will produce a forgetting curve. These models do not distinguish the results of review (remembered or forgotten), and only consider the number of reviews and the interval between reviews [22]. Mozer et al. [15] introduced the multiscale context model, which combines two cognitive theories and divides unsuccessful and successful recall, where some weights are hand-picked. Settle and Meeder [9] used the machine learning technique and exponential forgetting curve to predict student recall rates. Their feature sets include the number of times a student correctly and incorrectly recall but dismiss the time-series information in the history of review.

B. Optimization Schedules

Hand-Crafted Methods. Prior to the mass adoption of e-learning, traditional spaced repetition schedules were the mainstream. One of the first schedules was a geometric progression with a common ratio of five, introduced by Pimsleur [23]. Leitner [24] proposed a heuristic schedule based on physical boxes, where it controls the reviews' frequency of different flashcards by moving them to boxes of various sizes. By contrast, SuperMemo was the original digital spaced repetition algorithm, receiving users' interactions to update its schedule. Its program aims to keep users' forgetting rate at 5% [25], but there is no proof that it is optimal. These schedules rely on hand-crafted rules to determine spacing intervals for review and have less adaptability and theoretical guarantees.

Stochastic Control. Recently, Reddy et al. [26] proposed a queueing network model to maximize the learning speed for the Leitner system, which was constrained and not tested for accuracy. Tabibian et al. [27] introduced marked time-series point processes to represent review events in spaced repetition and designed an algorithm named MEMORIZE based on stochastic optimal control to make a tradeoff between recall

probability and the number of reviews. In their memory model, the forgetting rate dynamics were not time-varying. Upadhyay et al. [28] validated MEMORIZE in an actual interventional experiment. Hunziker et al. [29] used a greedy algorithm to maximize the average recall probability during the learning process, where they assumed recall and forgetting had the same impact on memory. These methods imposed strict conditions on their human memory models constraining their generality.

Deep Reinforcement Learning. Reddy et al. [16] proposed a model-free DRL method to maximize the expected number of items recalled. Sinha [17] improved the DRL method by using Long Short Term Memory (LSTM) [30] neural network to predict reward. These studies assumed that the intervals between each adjacent review are constant and oversimplified. To consider the varying internals in the real world, Yang et al. [18] utilized Time-LSTM to estimate the recall probabilities with time interval input. Nioche et al. [31] proposed a model-based planning approach at the level of individual learners and items. These approaches formulated scheduling of spaced repetition as a POMDP, encoding the memory of all material into one single state variable, making it hard to introduce new stuff during the learning period. Furthermore, it is impractical that the agent could only learn one item per session in their simulation environments. Upadhyay et al. [32] introduced a deep reinforcement learning algorithm based on the policy gradient that encoded the history of all items' reviews into a hidden state and allowed multiple items per session. But their method required determining the set of items before training, which is not convenient for students who need to import new items during the learning process.

III. Memory Model Based on Time-Series

The time-series data in spaced repetition and the memory model known as halflife regression (HLR) [9] are briefly introduced in this section. Then, we apply a manually created time-series model and a recurrent neural network to combine time-series features with the HLR.

A. Time-Series Data in Spaced Repetition

Inspired by work of [27], we use a quadruplet to represent each review event:

$$e := (u, w, \Delta t, r), \tag{1}$$

where e is the review event that the learner u recalls item w successfully ($r = 1$) or unsuccessfully ($r = 0$) at interval Δt since last review. Based on that, we can concatenate $(\Delta t, r)$ of each review to obtain sequential features:

$$e_i := (u, w, \boldsymbol{\Delta t}_{1:i-1}, \boldsymbol{r}_{1:i-1}, \Delta t_i, r_i) \tag{2}$$

where $\boldsymbol{\Delta t}_{1:i-1}$ denotes the intervals between each review before the ith review and $\boldsymbol{r}_{1:i-1}$ is the historical responses of reviews. The samples are shown in Table I.

The review event is defined to include the whole reviewing history of any student for any item. However, in the aforementioned review event, recall is binary (i.e., a user either recalls or forgets a word). The recall probability needs to be obtained to capture memory dynamics. The recall rate is defined by [9] as the percentage of times a word is correctly recalled throughout a review session, implying that different memory actions for the same word during a session are independent. In practice, the first recall event has a considerable impact on a learner's memory state and subsequent memories throughout the day. We, therefore, propose a more appropriate measure for recall ratio. We use $n_{r=1}/N$ in a group of N individuals learning word w as the recall probability p:

$$e_i := (w, \boldsymbol{\Delta t}_{1:i-1}, \boldsymbol{r}_{1:i-1}, \Delta t_i, p_i, N). \tag{3}$$

By controlling the w, $\boldsymbol{\Delta t}_{1:i-1}$ and $\boldsymbol{r}_{1:i-1}$, we can plot the p for each Δt to obtain the forgetting curve. When N is big enough, the ratio $n_{r=1}/N$ gets close to the recall probability. However, there are almost 100,000 words in MaiMemo, and the behavior events collected for each word in different time-series are sparse. We need to group words to make a tradeoff between distinguishing different words and alleviating data sparsity. Since we are interested in the forgetting curve, words' difficulties significantly influence the forgetting slope. As a result, we try to use the recall ratio the next day after learning them for the first time as a criterion for classifying the difficulties of words. The distribution of the recall ratio is shown in Fig. 2(a).

We can see from the data distribution that the recall ratio is mostly between 0.45 and 0.85. The words are divided into ten difficulty groups for the balance and density of grouping data, shown in Fig. 2(b) and Table II, respectively. The symbol d indicates the difficulty; the higher the number, the greater the difficulty. Then the exponential forgetting curve function $p_i =$

TABLE I
Dataset Samples

u	w	$\boldsymbol{\Delta t}_{1:i-1}$	$\boldsymbol{r}_{1:i-1}$	Δt_i	r_i
23af1d	solemn	0,1,3,1,3,6,10	0,1,0,1,1,1,0	1	0
23af1d	dominate	0	0	1	1
e9654e	nursery	0,1,1,3,1,3	0,0,1,0,1,1	1	1
948c7c	nursery	0,1	0,1	3	1

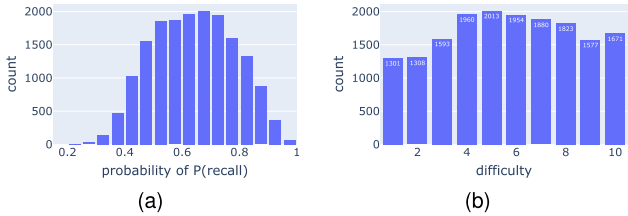


Fig. 2. The distributions of P(recall) and difficulty. The easiest words with P(recall) > 0.85 are assigned $d = 1$ and the hardest words P(recall) ≤ 0.45 are assigned $d = 10$. The difficulties of the remaining words are assigned from 2 to 9 by dividing remaining interval of P(recall) equaly into eight parts.

TABLE II
Illustration of Words' Difficulty

w	d	$\boldsymbol{\Delta t}_{1:i-1}$	$\boldsymbol{r}_{1:i-1}$	Δt_i	p_i	N
automobile	2	0	0	1	0.8379	7501
emission	8	0	0	1	0.5094	7497
multiply	6	0	0	1	0.635	7495
hatch	10	0	0	1	0.4371	7492

TABLE III
ILLUSTRATION OF GROUPING

d	$\Delta \boldsymbol{t}_{1:i-1}$	$\boldsymbol{r}_{1:i-1}$	$\boldsymbol{p}_{1:i-1}$	Δt_i	p_i	h_{i-1}	N
1	0,1	0,1	0.84,0.86	3	0.92	25.6	674825
5	0,1	0,1	0.63,0.61	4	0.74	8.9	426235
9	0,1	0,1	0.34,0.39	2	0.73	5	493590

$2^{-\Delta t_i/h_{i-1}}$ can be used to fit the halflife h_i of memory and add history of recall probabilities:

$$e_i := (d, \Delta \boldsymbol{t}_{1:i-1}, \boldsymbol{r}_{1:i-1}, \boldsymbol{p}_{1:i-1}, \Delta t_i, p_i, h_{i-1}, N), \quad (4)$$

The samples of (4) are shown in Table III. The $\Delta \boldsymbol{t}_{1:i-1}$, $\boldsymbol{r}_{1:i-1}$ and $\boldsymbol{p}_{1:i-1}$ are the times-series features which will be integrated into the HLR.

B. Halflife Regression Model (HLR)

Settle and Meeder [9] define the halflife regression model as follows:

$$p = 2^{-\Delta/h}, \quad (5)$$

where p denotes the probability of recall, Δ is the lag time since the item is last reviewed, and h is the halflife or strength of the learner's memory of the item.

Let $\hat{h}_{\boldsymbol{\Theta}}$ denote the estimated halflife, defined as:

$$\hat{h}_{\boldsymbol{\Theta}} = 2^{\boldsymbol{\Theta} \cdot \boldsymbol{x}}$$
$$\boldsymbol{x} = (x_{\oplus}, x_{\ominus}, \boldsymbol{lex}), \quad (6)$$

where $\boldsymbol{\Theta}$ is a weight vector for the feature vector $\boldsymbol{x}$, which consists of the times a word is correctly recalled $x_{\oplus}$, the times incorrectly recalled $x_{\ominus}$ and the lexeme tag $\boldsymbol{lex}$. These features capture the statistical information in each student's review history with each item.

The HLR model is trained by the following loss function:

$$loss = (p - \hat{p})^2 + \alpha(h - \hat{h})^2 + \lambda ||\boldsymbol{\theta}||^2, \quad (7)$$

to optimize both p and h in the loss function.

C. DHP-HLR Model

We manually develop *D*ifficulty-*H*alflife-*P*(recall)-HLR with the Markov property for explainability and simplicity to enhance HLR. In DHP-HLR, the dimensionalities of time-series are decomposed into state variables and state-transition equations. We take into account the following four factors:

- Halflife. It measures the storage strength of memory.
- P(recall). It measures the retrieval strength [33] of memory. According to the spacing effect [3], the interval between each review affects the halflife. When h is fixed, Δt and p are mapped one-to-one. We use $p = 2^{-\Delta t/h}$ instead of Δt as a state variable for normalization.
- Result of recall. The halflife increases after recall and decreases after forgetting.
- Difficulty. Intuitively the higher the difficulty, the harder the memory to be consolidated.

The last halflife, recall probability, and halflife are used to project the time-series data into a three-dimensional space. As illustrated in Fig. 3, the color denotes the degree of difficulty. Observing the projection of the data, we notice two phenomena: $h_i > h_{i-1}$ when $r_i = 1$ and $h_i < h_{i-1}$ when $r_i = 0$. They imply that a word's halflife lengthens if a student remembers it throughout a review. In turn, the halflife shortens if the learner forgets. To further explore the dynamics of halflife during reviews, we extract a group of adjacent review events shown in Fig. 4. Obviously, as the time between reviews increases, the probability of recall decreases (blue line). And the halflife following a successful review lengthens (red, green, and orange lines) as the probability of memory declines, a phenomenon known as the lag effect [34]. According to the lag effect, recall after long intervals between learning sessions performs than recall after short intervals.

Considering the above observations, the state-transition equation can be formulated as:

$$h_i = [h_{i-1} \cdot (e^{\boldsymbol{\theta}_1 \cdot \boldsymbol{x}_{i-1}} + 1), e^{\boldsymbol{\theta}_2 \cdot \boldsymbol{x}_{i-1}}] \cdot [r_{i-1}, 1 - r_{i-1}]^T, \quad (8)$$

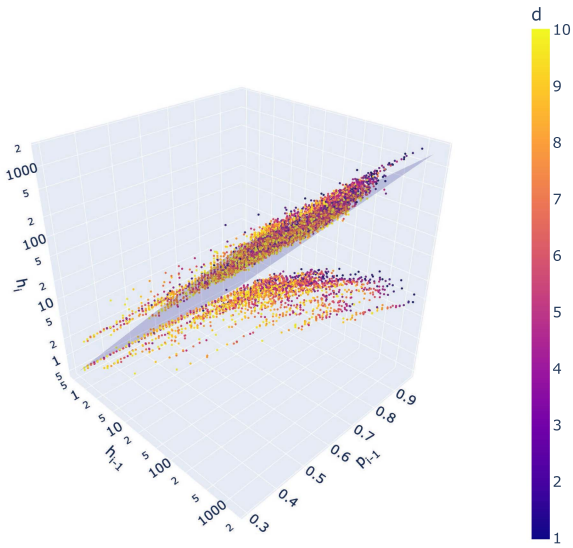


Fig. 3. The projection in Difficulty, Halflife, and P(recall). The h_{i-1} and h_i denote last halflife and halflife. Similiarly, the p_{i-1} denotes P(recall).

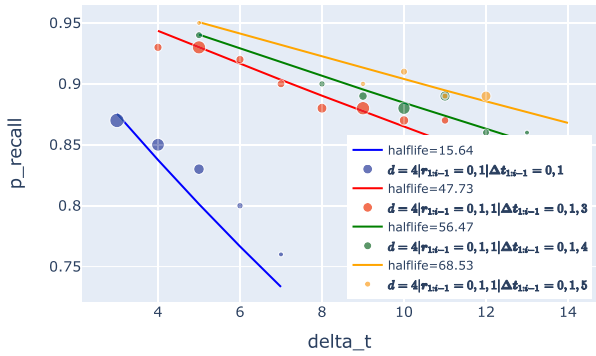


Fig. 4. Forgetting curves of a set of related review events. The blue line shows the forgetting curve after two reviews. And the red, green, and orange lines depict the forgetting curves after three reviews which follow the successful review in the blue line with three, four, and five days.

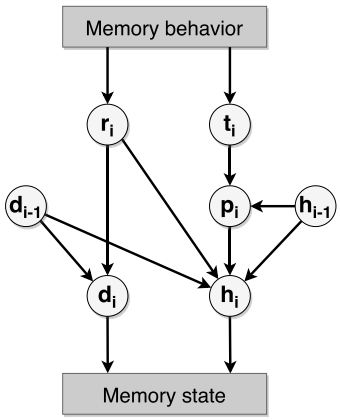


Fig. 5. The structure of DHP-HLR. d_i and h_i represent the memory state during the spaced repetition. The current input of memory behavior and the last memory state determine the new memory state. Therefore, DHP-HLR has the Markov property.

where $\boldsymbol{x}_i = [\log d_{i-1}, \log h_{i-1}, \log(1-p_i)]$ is the feature vector. $e^{\boldsymbol{\theta}_1 \cdot \boldsymbol{x}_i} + 1$ guarantees the h_i after a recall greater than the h_{i-1}. And $e^{\boldsymbol{\theta}_2 \cdot \boldsymbol{x}_i}$ means that the halflife after a forgetting is constantly positive.

The data shows that if learners forget during the review, the halflife of future successful recall will be shorter than those have not been forgotten, even under the same recall probability and last halflife conditions. It is explained by the fact that harder word is more likely to be forgotten. The word that has been forgotten is, therefore, considered more difficult. As a result, the difficulty also has a state-transition equation:

$$d_i = [d_{i-1}, d_{i-1} + \theta_3] \cdot [r_{i-1}, 1 - r_{i-1}]^T, \tag{9}$$

where θ_3 is greater than zero to keep the difficulty increasing in each forgetting event. We set an upper limit to prevent the difficulty from increasing indefinitely.

Finally, we formulate the memory state-transition equation set of DHP-HLR:

$$\begin{aligned}
\begin{bmatrix} h_i \\ d_i \end{bmatrix} &= \begin{bmatrix} h_{i-1}\left(e^{\boldsymbol{\theta}_1 \cdot \boldsymbol{x}_{i-1}} + 1\right) & e^{\boldsymbol{\theta}_2 \cdot \boldsymbol{x}_{i-1}} \\ d_{i-1} & d_{i-1} + \theta_3 \end{bmatrix} \begin{bmatrix} r_{i-1} \\ 1 - r_{i-1} \end{bmatrix} \\
\boldsymbol{x}_{i-1} &= [\log d_{i-1}, \log h_{i-1}, \log(1-p_{i-1})] \\
r_{i-1} &\sim Bernoulli(p_{i-1}) \\
p_{i-1} &= 2^{-\Delta t_{i-1}/h_{i-1}} \\
h_1 &= -1/\log_2(0.925 - 0.05 \cdot d_0),
\end{aligned} \tag{10}$$

where the parameters for h_1 are derived from the grouping of difficulty.

Based on (10) and the initial values of difficulty d_0, the halflife h_i of any memory behaviors can be calculated. The calculation process is displayed in Fig. 5.

D. GRU-HLR Model

Explainability is a benefit of DHP-HLR. However, manually designing the state transition equation is its drawback. Therefore, GRU network, a type of recurrent neural network, is introduced to facilitate the process.

Based on (4), GRU-HLR can be desribed as follows:

$$\hat{h}_i = \text{GRU}(\boldsymbol{\Delta t}_{1:i-1}, \boldsymbol{r}_{1:i-1}, \boldsymbol{p}_{1:i-1}). \tag{11}$$

To reduce the error of predicted recall probability, we improve the loss function in (7) as:

$$l(e_i, \boldsymbol{\theta}) = \left| \frac{h_i - \hat{h}_i}{h_i + \hat{h}_i} \right| + C||\boldsymbol{\theta}||_2^2, \tag{12}$$

where we replace the mean squared error (MSE) with the symmetric mean absolute percentage error (sMAPE) to minimize the mean absolute error (MAE) of p.

With GRU-HLR, it is possible to not only predict the halflife and recall probability, but also capture the dynamics of memory:

$$\boldsymbol{h}_i, \boldsymbol{s}_i = \text{GRU-HLR}(\boldsymbol{s}_{i-1}, \Delta t_{i-1}, r_{i-1}, p_{i-1}), \tag{13}$$

where $\boldsymbol{s}$ is the hidden state in GRU, corresponding to the memory state of a word in the learner's mind. The GRU describes how the memory state is updated from the $i-1$th review event to ith review event. Moreover, the memory state in spaced repetition can be formulated as the Markov decision process (MDP), where the action and cost are formulated in Section IV-B.

IV. Spaced Repetition Schedule Optimization

In this section, we set up a practical goal for spaced repetition and formulate it as a stochastic shortest path problem, which can be solved by stochastic dynamic programming.

A. Problem Setup

Learners can effectively establish long-term memory via spaced repetition. The number of repetitions and the time spent on each repetition represent the cost of memory, whereas the memory halflife assesses the long-term memory's storage strength. Therefore, the goal of spaced repetition schedule optimization is to achieve a particular quantity of learned content with the target halflife at the lowest possible memory cost or to consolidate additional learning materials to the target halflife within a certain memory cost limitation. The latter may be reduced to making a memory material achieve the desired halflife at the minimized memory cost (MMC).

The long-term memory model we construct in Section III-D satisfies the Markov property. In DHP-HLR and GRU-HLR, the state of each memory depends only on the last state, the current review interval, and the result of the recall, which follows a distribution that relies on the review interval. Due to the randomness of halflife state-transition, the number of reviews required for memorizing material to reach the target halflife is uncertain. Therefore, the spaced repetition scheduling problem can be regarded as a problem of infinite-time stochastic dynamic programming. Since it has a termination state, which is the target halflife, in the case of long-term memory formation. It could be transformed into a stochastic shortest path (SSP) problem [35], of which the goal is to control an agent, who dynamically evolves in a system consisting of finite states, to

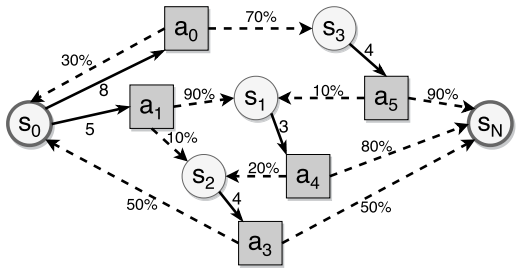


Fig. 6. The stochastic shortest path problem in spaced repetition. The percentage above the dashed arrows is the probability of successful or forgetting if the learner executes the corresponding review interval. The number attached to solid arrows is the cost when the learner chooses the review interval.

converge on a predetermined objective. Actions attached with costs are scheduled for the agent in each time period. Transitions in the system are regulated by probability distributions, which depend only on the last action. The policy's goal is to select an action for each state to minimize the total expected cost incurred by the agent before reaching the target state beginning from a given initial state. In the case of spaced repetition optimization, the states are the halflife of memory and other hidden states; the actions are the intervals for the next reviews; the cost is the time of each review; the target state is a long halflife, which means the memory is stable enough that no need to review again.

Combining with the optimization goal, we name the algorithm as SSP-MMC.

As shown in Fig. 6, circles are memory states, squares are review action (i.e., the interval after the current review), dashed arrows indicate state transitions for a given review interval, and black edges represent review intervals available in a given memory state. The stochastic shortest path problem in spaced repetition is to find the optimal review interval to minimize the expected review cost of reaching the target state.

B. Formulation

To solve the problem, our proposed method is to model the reviewing process for a word as an MDP with a set of states $\mathcal{S}$, actions $\mathcal{A}$, state-transition probability $\mathcal{P}$, and cost function $\mathcal{J}$. The agent's goal is to find a policy π that minimizes the expected review cost to achieving the target state s_N:

$$\pi^* = \underset{\pi\in\Pi}{\operatorname{argmin}} \lim_{N\to\infty} \mathbb{E}_{s_0,a_0,\ldots}\left[\sum_{t=0}^{N} \mathcal{J}\left(s_t, a_t\right) \mid \pi\right]. \tag{14}$$

The state-space $\mathcal{S}$ depends on the state size of the memory model. The DHP-HLR only has two state variables so that the state can be formulated as $s = (d, h)$. For GRU-HLR, the state relies on the hidden layers, which use $\tanh$ as the activate function with range of $(-1, 1)$. It can be discretized as follows:

$$S \xrightarrow{(\lfloor \frac{s}{\varepsilon} \rfloor | s\in\mathcal{S})} \mathrm{S}, \tag{15}$$

where ε is the step width of discretization.

The action space $\mathcal{A} = \{\Delta t_1, \Delta t_2, \ldots, \Delta t_n\}$ consists of N intervals that the agent can schedule for the item. We discretize the intervals to days because most users prefer to review at a specific time block, instead of the entire day. And it is infeasible to control the specific timing of reviews when the actual user has other tasks to do. The state-transition probability $\mathcal{P}_{s,a}(s')$ is the probability item recalled at state s and action a, described in (5). The cost function $\mathcal{J}$ is defined as:

$$\mathcal{J}(s_0) = \lim_{N\to\infty} E\left\{\sum_{t=0}^{N-1} g_t(s_t, a_t(s_t), r_t)\right\}$$
$$r_t \sim Bernoulli(p_t), \tag{16}$$

where the g_t is the cost per stage and the r_t is the result of recall which follows the Bernoulli distribution. The target state s_N corresponds to a halflife bigger than h_N, which is the target halflife.

C. Algorithm

We solve the MDP($\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{J}$) using value iteration with DHP-HLR and GRU-HLR to capture the dynamic of the memory state. The Bellman equation is:

$$\mathcal{J}^*(s) = \min_{a\in\mathcal{A}(s)} \left\{\sum_{s'} \mathcal{P}_{s,a}(s')(g(r) + \mathcal{J}^*(s'))\right\}$$
$$s' = \mathcal{F}(s, a, r, p), \tag{17}$$

where the $\mathcal{J}^*$ is the optimal cost function, and $\mathcal{F}$ is the state-transfrom function including DHP-HLR and GRU-HLR. For simplicity, we only consider the response of recall r: $g(r) = a \cdot r + b \cdot (1 - r)$, a is the cost of recall and b is the cost of forgetting.

Based on (17), the value iteration algorithm, as described in Algorithm 1, uses a cost matrix to record the optimal cost and a policy matrix to save the optimal action for each state during the iteration.

In addition to the Algorithm 1, if it uses GRU-HLR as the state-transfrom function, the halflife h can be transformed from memory state s via the fully-connected layer of GRU-HLR.

V. Experiment

This section evaluates our framework in two aspects: memory predicting and schedule optimizing. To obtain deeper insight, we also analyze the model weights and the policy derived from SSP-MMC.

A. Memory Predicting

1) Experimental Setting: Dataset We collected 220 million review event logs formulated in (1) from the online language-learning APP MaiMemo, and preprocessed them into 71,697 grouping samples formulated in (4).

Baselines We compared DHP-HLR and GRU-HLR with Pimsleur [23], Leitner [24], HLR and its variant [9]. To understand the contribution of the time-series features to GRU-HLR, we set ablation experiments, considering four variants of GRU-HLR: with and without $\boldsymbol{\Delta t}_{1:i-1}$ features (-t), and with and without the $\boldsymbol{p}_{1:i-1}$ feature (-p).

Algorithm 1: SSP-MMC.

Data: a, b, h_N
Result: $\pi, \mathcal{J}$
$\mathcal{J} = \inf$;
$\mathcal{J}(s_N) = 0$;
while $\Delta J < 0.1$ **do**
  $\mathcal{J}_0 = \mathcal{J}(s_0)$;
  for $s \leftarrow s_0$ **to** s_{N-1} **do**
    foreach $a \in A(s)$ **do**
      $p \leftarrow 2^{-\frac{a}{h}}$;
      $s_{r=1} \leftarrow \mathcal{F}(s, a, 1, p)$;
      $s_{r=0} \leftarrow \mathcal{F}(s, a, 0, p)$;
      $\mathcal{J} \leftarrow p \cdot (a + \mathcal{J}(s_{r=1})) + (1-p) \cdot (b + \mathcal{J}(s_{r=0}))$;
      if $\mathcal{J} < \mathcal{J}(s)$ **then**
        $\mathcal{J}(s) = \mathcal{J}$;
        $\pi(s) = a$;
      end
    end
  end
  $\Delta\mathcal{J} = \mathcal{J}_0 - \mathcal{J}(s_0)$;
end

TABLE IV
PERFORMANCE OF EACH MEMORY MODEL ON TWO METRICS (BOLD FONT FOR THE BEST)

Model	MAE(p)	sMAPE(h)
GRU-HLR	**0.0307**	**18.88%**
GRU-HLR -t	0.0309	**18.88%**
GRU-HLR -p	0.0328	20.61%
GRU-HLR -t -p	0.0376	22.39%
DHP-HLR	0.0779	46.35%
HLR	0.1070	76.65%
HLR -lex	0.1152	80.94%
Pimsleur	0.3169	165.69%
Leitner	0.4535	133.92%

Metrics We considered two different criteria to assess the performance. The first metric is MAE(p), the absolute error between the predicted recall probability and the actual recall probability. The recall probability is a value between 0 and 1. The smaller the MAE, the more accurate the prediction. The second metric is sMAPE(h), the relative error between the predicted and actual halflife.

Implements The size of the hidden layer of GRU-HLR determines the dimensions of state space. For a fair comparison between DHP-HLP and GRU-HLR, we set the hidden layer size to 2. We randomly selected 20% of the data for tuning the hyperparameters of GRU-HLR and finally selected the followings: iterations=1,000,000, learning_rate=0.001, weight_decay=0.00001. For the remaining 80% of the data, 5×2-fold cross-validation was used for evaluation.

2) Result and Analysis: Table IV reports the results of each memory model on two metrics. We can see GRU-HLR with both $\boldsymbol{p}_{1:i-1}$ and $\boldsymbol{\Delta t}_{1:i-1}$ features performs the best. Moreover, all GRU-HLR variants achieve lower MAE by at least 64% when compared to HLR. The DHP-HLR outperforms the original HLR. We drew the distributions of errors depicted in Fig. 7 in order to analyze the benefits of our models. The recall probability

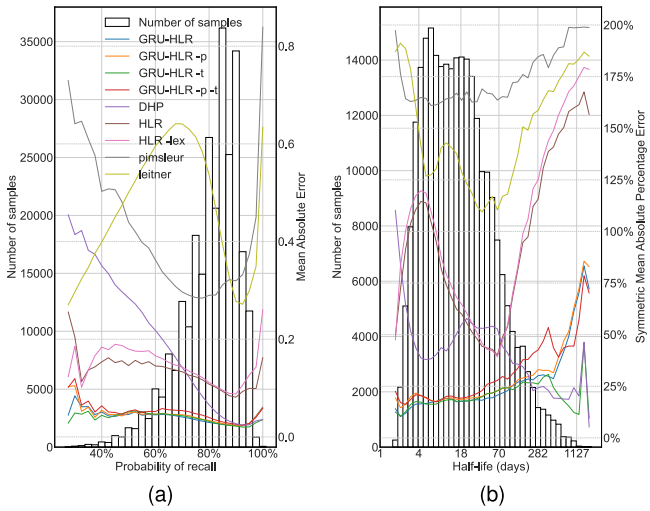


Fig. 7. Distribution of actual recall probability, halflife and errors.

distribution for the dataset is shown by the bars in Fig. 7(a). The majority of samples has recall probabilities between 80% and 90%. The MAE(p) of GRU-HLRs is less than 0.05 in $(40\%, 100\%)$. However, since the recall probability depends on the memory's halflife and the review interval, it is more feasible to directly analyze the distribution of sMAPE(h) shown in Fig. 7(b).

Most halflife values, in Fig. 7(b), fall between 4 to 32 days. The HLRs perform worse when the halflife is less than 16 days or more than 129 days. It is because statistical features, e.g., the accumulative historical number of recall and the number of forgetting, cannot distinguish the difference between sequences like $\boldsymbol{r}_{1:2} = (1, 0)$ and $\boldsymbol{r}_{1:2} = (0, 1)$. The difference is significant in our dataset shown in Fig. 3. The halflife after forgetting is significantly less than the halflife after recall, but the HLRs can only yield compromised predictions during training, inducing large errors. Another noteworthy observation is that most models have large errors in the interval with higher halflife and fewer samples, but HLRs also have greater errors in the interval with a halflife of 4 to 8 days. In addition to the effect caused by the statistical characteristics, we argued that the loss function in (7) also contributes negatively. The item $\alpha(h - \hat{h})^2$ punishes the error of any halflife interval indiscriminately so that the percentage error of the low halflife interval is substantial. The GRU-HLRs overcome this problem by improving the loss function with sMAPE in (12). As for the higher halflife interval greater than 141 days, most models perform unsatisfactorily, probably because the noise from the real world increases. From a practical point of view, when the memory halflife becomes longer, the possibility of learners reviewing outside the APP before the next review is increasing, and the memory behavior data collected will also deviate from the learner's real situation. For these partial cases, making accurate predictions is, arguably, less important.

Returning to Table IV, the results of the ablation experiments are that GRU-HLR -p with additional feature $\boldsymbol{\Delta t}_{1:i-1}$ and GRU-HLR -t with additional feature $\boldsymbol{p}_{1:i-1}$ are better than GRU-HLR -p -t with only feature $\boldsymbol{r}_{1:i-1}$. Among them, the feature $\boldsymbol{p}_{1:i-1}$ is

the most helpful in reducing the error. Moreover, the difference between GRU-HLR with both $\boldsymbol{\Delta t}_{1:i-1}$ and $\boldsymbol{p}_{1:i-1}$ features and GRU-HLR -t is smaller. As to why features can reduce prediction errors, we assumed it is related to the memory's retrieval strength and storage strength in psychology. Storage strength represents how well learned something is; retrieval strength is how accessible (or retrievable) something is. Memories that are more difficult to retrieve tend to be strengthened more by recall [33]. In the proposed models, the historical recall probability represents the retrieval strength of each recall, and the halflife is equivalent to the storage strength. Therefore, the recall probability history is useful for predicting the halflife. The interval between reviews is also important information. Reviewing at intervals (0-2-4-6-8) and reviewing at intervals (0-5-5-5-5) have significantly different results [36], as verified by the results of GRU-HLR -p and GRU-HLR -t -p. However, the difference between GRU-HLR and GRU-HLR -t is negligible. We argued that the feature $\boldsymbol{p}_{1:i-1}$ holds information of the feature $\boldsymbol{\Delta t}_{1:i-1}$. According to (5), p_i relies on Δt_i and h_{i-1} which is already contained in the hidden layer. Therefore, the review interval can be considered as a surrogate variable of the recall probability. It is worth noting that the review interval is truncated to the nearest day. Some information may have been lost, whereas the recall probability is not subject to such limitation. This may explain why the contribution of recall probability features is greater than that of review interval features.

B. Model Analysis

Interpretability is a feature of DHP-HLR, and we visualized DHP-HLR in 3D plots. At the same time, we performed a similar analysis on GRU-HLR to explain whether the neural network learns similar patterns.

1) DHP-HLR Model: According to the parameters obtained by fitting the dataset and the equations of the model, we obtained the recurrence formula of the halflife after a successful recall:

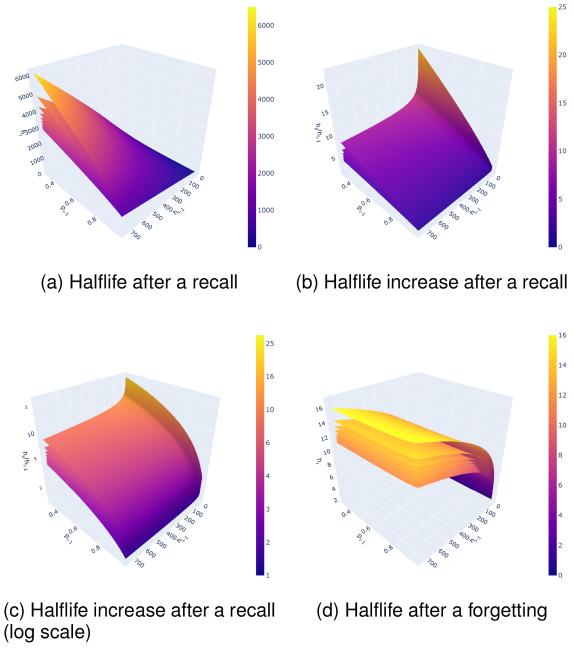
$$h_i = h_{i-1} \cdot (e^{3.25} \cdot d_{i-1}^{-0.386} \cdot h_{i-1}^{-0.147} \cdot (1 - p_{i-1})^{0.821} + 1), \tag{18}$$

where the exponent of base d is negative, the h_i decreases with the growth of d. Fig. 8(a) illustrates that the h_i after successful recall increases as p decreases, which verifies the existence of the spacing effect [3]. Fig. 8 shows that the growth of the h_i decreases as the h_{i-1} increases, which may imply that the potential of learns' memory consolidation decreases as the memory storage strength increases, i.e., there is a marginal effect.

Similarly, the recurrence formula of the halflife after a forgetting is:

$$h_i = e^{1.003} \cdot d_{i-1}^{-0.152} \cdot h_{i-1}^{0.264} \cdot (1 - p_{i-1})^{-0.017}, \tag{19}$$

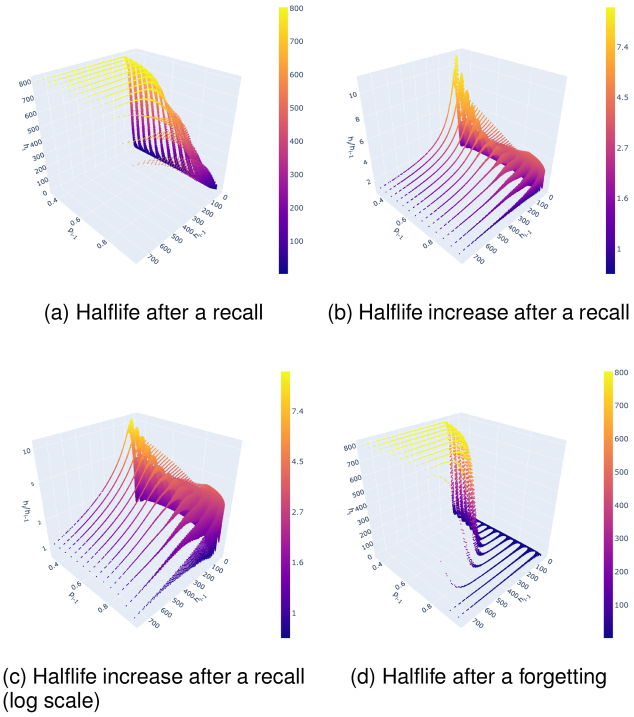
where the exponent of bases d is less than its corresponding, which means the difficulty has a weak impact on the memory after a forgetting. Fig. 8(d) shows that the longer the h_{i-1}, the longer its h_i after a forgetting, which may be because the memory is not entirely lost in forgetting. Moreover, as p_{i-1} decreases, the h_i after a forgetting also decreases, possibly due to the fact that the memory is forgotten more entirely over time.



(a) Halflife after a recall (b) Halflife increase after a recall

(c) Halflife increase after a recall (log scale) (d) Halflife after a forgetting

Fig. 8. The projection of DHP-HLR.



(a) Halflife after a recall (b) Halflife increase after a recall

(c) Halflife increase after a recall (log scale) (d) Halflife after a forgetting

Fig. 9. The projection of GRU-HLR.

2) GRU-HLR Model: GRU-HLR is visualized in the similar manner. By traversing each memory state and review interval, we computed the h_{i-1}, p_{i-1} and h_i and mapped them to a 3-dimensional space in Fig. 9. We found that the pattern of GRU-HLR prediction is very similar to that of DHP-HLR. For example, Fig. 9(a) shows that the h_i increases with the growth of the h_{i-1} and the decline of the p_{i-1}. One of the key differences

between Figs. 8(a) and 9(a) is that the h_i in GRU-HLR saturates when the p_{i-1} becomes small enough, e.g., around 0.45 as shown in Fig. 9(a). It is related to the halflife range in our dataset. Longer halflife data require a longer time to collect. The h_i increases after recall in Fig. 9(a) is higher at the lower h_{i-1} and also has the same pattern as DHP-HLR. However, when the h_{i-1} is close to 0, the difference is more distinguishable, possibly because more difficult words tend to produce many data with a lower halflife, and these words generally have lower halflife increases. Comparing Figs. 8(d) and 9(d), there is a huge divergence between DHP-HLR and GRU-HLR. In GRU-HLR, the predicted h_i after a forgetting is near 800 days when p_{i-1} is 50% less. But that in DHP-HLR is less than 16 days. Due to data sparsity in this area, it remains unclear that which model performs better in this experiment. Based on domain experience and prerequisite information, however, we conjectured that DHP-HLR's predictions are closer to reality. It is the disadvantage of the neural network-based model that small sample size in certain area and big hypothesis space may lead to large errors [37].

C. Schedule Optimizing

1) Environment: The environment is based on DHP-HLR and GRU-HLR trained in Section V-A. The simulation process involves two dimensions, inter-day and intra-day. To simulate the learner's preparation period and daily study time constraints, the environment limits the number of days the simulation is conducted and the time spent on review and study each day. However, due to the stochastic nature of memory, the review schedule for the day may take longer than the daily time limit. To alleviate this situation, the review is scheduled before learning. When the daily time is exhausted, the remaining review is postponed to the next day regardless of whether it is completed.

The simulation process is shown in Algorithm 2, where the $Student$ represents the learner's memory model, which updates the memory state according to the review situation. The $Schedule$ is the interval repetition algorithm scheduler that adjusts the review interval based on the learner's feedback on the memory state. The day_{limit} is the period limit. The $cost_{limit}$, the daily review cost limit, is the maximum amount of time a learner can spend on review per day. The h_N is the target halflife. When the halflife of the word exceeds this value, it will not be scheduled for review and will be remembered forever.

We set a recall halflife of 360 days (near one year) as the target halflife and set 600 s (10 min) as the upper limit of daily learning cost. We used the average time spent by learners of 3 s for recall and 9 s for forgetting. Then, we set a simulation duration of 1000 days for learning.

2) Baselines and Metrics: We compared SSP-MMC with five baseline scheduling algorithms:

- RANDOM, which chooses a random interval from $[1, \text{halflife}]$ to schedule the review.
- ANKI, a variant of SM-2 [38].
- HALF-LIFE, where the halflife is used as the review interval.

Algorithm 2: Spaced Repetition Simulator.

Data: $Student, Schedule, h_N, day_{\text{limit}}, cost_{\text{limit}}$
Result: $rewiew$
for $day \leftarrow 1$ **to** day_{limit} **do**
  $cost \leftarrow 0$;
  foreach $w \in rewiew[day]$ **do**
    if $cost \geq cost_{\text{limit}}$ **then break**;
    $w.t \leftarrow day - w.last$;
    $w.p \leftarrow 2^{-\frac{w.ivl}{w.h}}$;
    if $random() < w.p$ **then**
      $w.r, cost \leftarrow 1, cost + cost_{\text{r=1}}$;
    else
      $w.r, cost \leftarrow 0, cost + cost_{\text{r=0}}$;
    end
    $w.x \leftarrow Student.review(w.x, w.r, w.t, w.p)$;
    $w.last \leftarrow day$;
    if $w.h \geq h_N$ **then**
      $w.h \leftarrow \infty$; **continue**;
    $rewiew[day + Schedule(w.x)].add(w)$;
  end
  foreach $w \in new$ **do**
    if $cost \geq cost_{\text{limit}}$ **then break**;
    $cost \leftarrow cost + cost_{\text{new}}$;
    $w.x \leftarrow Student.new()$;
    $rewiew[day + Schedule(w.x)].add(w)$;
    $w.last \leftarrow day$;
    $new.pop(w)$;
  end
end

- THRESHOLD, review when p is less than or equal to a certain level (we adopted 90% which is default in SuperMemo [25]).
- MEMORIZE, an algorithm based on optimal control, with codes from the open-source repository of [27]. It is trained to determine the parameter for minimizing expectation of review cost.

Our evaluation metrics include:

- *THR* (target halflife reached) is the number of words that reach the target halflife.
- *SRP* (summation of recall probability) is the summation of all learned words' recall probability. The recall probability is set at 100% when the word reaches the target halflife to simulate that the learner has formed a solid long-term memory.
- *WTL* (words total learned) is the number of total learned words.

3) Result and Analysis: It is expected that SSP-MMC outperforms all baselines in the measure THR in both of the proposed memory models. The THR is consistent with the optimization goal of SSP-MMC, thus, SSP-MMC can reach the upper bound of this metric theoretically. To quantify the relative difference between the performance of each algorithm, we compared the number of days for THR $= 2000$ (i.e., the COST of total review shown in Table V). SSP-MMC saves about 12.5% cost of review compared with MEMORIZE in DHP-HLR and 16.8% compared

TABLE V
RESULTS OF SIMULATIONS IN SEVERAL SCHEDULES AND MEMORY MODELS

	DHP-HLR		GRU-HLR	
	THR	COST	THR	COST
SSP-MMC	12364	203	6742	425
THRESHOLD	11088	239	4514	511
ANKI	10886	244	3704	611
HALF-LIFE	3163	706	351	N/A
MEMORIZE	10345	232	2041	982
RANDOM	6586	399	2271	902

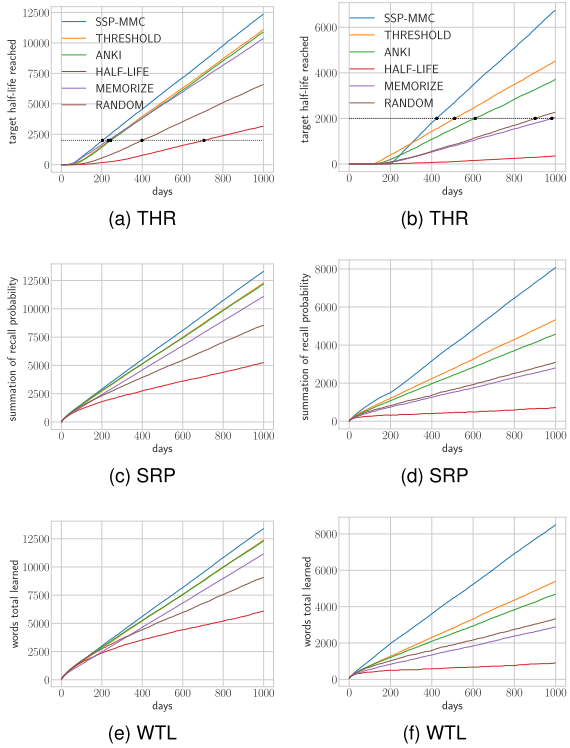


Fig. 10. Simulation results of schedule optimizing.

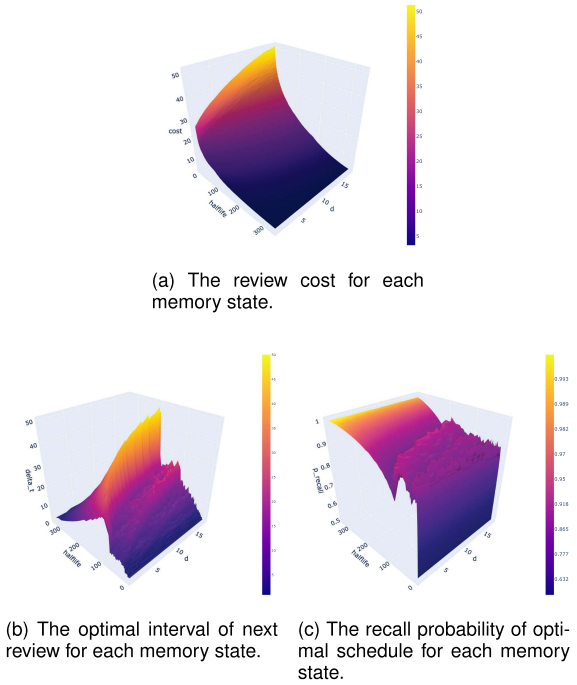


Fig. 11. The optimal policy on DHP-HLR.

with THRESHOLD in GRU-HLR. The trend of THR during the simulation is approximately linear, as shown in Fig. 10(a) and (b). It means that the speed of memorizing words is constant. Hence, the advantage of SSP-MMC over other baselines is almost independent of the learning time.

As shown in Fig. 10(c) and (d), the comparison among schedules in SRP is similar to that in THR. So the learner following the schedules of the SSP-MMC will remember the most. In the metric WTL shown in Fig. 10(e) and (f), the SSP-MMC outperforms other baselines because it minimizes the cost of memorization and gives learners more time to learn new words.

D. Policy Analysis

1) DHP-HLR Model: By training Algorithm 1 in the environment of the DHP model, we obtained the expected review cost and the optimal review interval for each memory state.

The review cost decreases as the halflife increases and increases as the difficulty increases, as shown in Fig. 11(a). Memories with a high halflife reach the target halflife at a lower expected cost. In addition, memories with greater difficulty have a higher expected cost because they have a lower halflife growth, as shown in Fig. 8(c), and require more reviews to reach the target halflife.

The interval increases with difficulty for the same level of memory halflife, as shown in Fig. 11(b). The reason could be that forgetting raises the difficulty of simple memories and decreases their halflife, leading to higher review costs. The scheduling algorithm tends to give shorter intervals to simple memories and reduce their probability of forgetting, even if it sacrifices a small amount of halflife boost. The interval reaches its peak in the midrange of halflife. It is necessary to compare Fig. 11(b) with 11(c) to explain the peak.

The recall probability corresponding to the optimal review interval increases with halflife and decreases with difficulty, as shown in Fig. 11(c). It means that the scheduler will instruct learners to review at a lower retrieval strength in the early stages of memorization, which may reflect "desirable difficulties"[33]. As the halflife increases to the target value, the recall probability approaches 100%. According to the equation $\Delta t = -h \cdot \log_2 p$ and the trend of p on h, optimal review interval Δt first increases and then decreases where the peak emerges.

2) GRU-HLR Model: To analyze the optimal policy derived by the SSP-MMC on GRU-HLR, we visualized the state space of GRU-HLR and its corresponding halflife, cost, interval, and recall probability.

Fig. 12(a) illustrates the halflife in each state $(s1, s2)$. The maximun and minimum values of the halflife are at $(1, -1)$ and $(-1, 1)$. Intuitively, the maximum of cost corresponds to the minimum of halflife. However, in Fig. 12(b), the maximum value of the cost is not located at the coordinates $(-1, 1)$. Also, the

(a) The halflife for each memory state.

(b) The review cost for each memory state.

(c) The optimal interval of next review for each memory state.

(d) The recall probability of optimal schedule for each memory state.

Fig. 12. The optimal policy on GRU-HLR.

costs are not equal on the contour lines of halflife. This suggests that, in addition to halflife, it requries more features to better represent the memory state, where difficulty is not negligible. Fig. 12(c) and (a) show that the optimal interval of review increases first and then decreases as the halflife increases, which is similar to the pattern shown in Fig. 11(b). The same pattern can also be found by comparing Fig. 11(d) and (c). Therefore, the optimal policy derived by the SSP-MMC algorithm also reflects the inherent similarity of the two memory models.

E. Generality of SSP-MMC

To further validate the generality of our optimization algorithm SSP-MMC, we port it into the experiment and dataset in [32]. The training and testing procedures are in line with their open-source code. The optimal policy of SSP-MMC is trained in their student model, the original HLR [9], described in (6). We compare the performance of SSP-MMC with two alternatives: (i) a state-of-the-art method called TPPRL [32], which does not make any assumptions on the model of memory, and (ii) a baseline schedule which chooses items uniformly at random with a constant reviewing rate.

The results are shown in Fig. 13, where the cost of review by each method is the same. Fig. 13(a) shows that our SSP-MMC is on par with TPPRL in maximizing the empirical recall probability at time $T + \tau$. Fig. 13(b) compares the average fraction of review cost per day across all items for SSP-MMC and TPPRL. Both SSP-MMC and TPPRL have a constant load over time. To summarize, SSP-MMC still has a good performance on the dataset of [32], which proves the generality of SSP-MMC. Besides, the time cost of SSP-MMC's training process, which

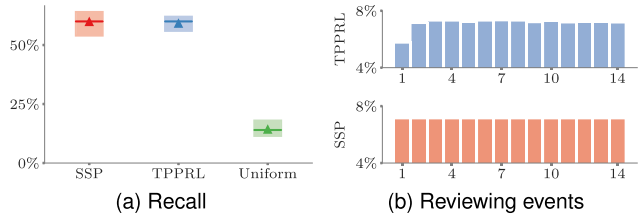


Fig. 13. Performance of SSP-MMC against TPPRL and a uniform baseline.

linearly depends on the size of memory states, is less than TPPRL, which requires thousands of iterations with given items and learning time. In this experiment, training SSP-MMC only costs several seconds, but training TPPRL takes at least one hour. And SSP-MMC is more scalable than TPPRL. If the number of items or the length of the learning period was changed, TPPRL needs to train from scratch. In contrast, SSP-MMC only needs training once if the memory model doesn't change. In summary, our SSP-MMC not only has achieved promising performance, but also costs less time over TPPRL.

VI. Conclusion and Future Work

We designed a long-term memory model based on time-series information that can well fit the existing data, and provided a solid foundation for optimizing spaced repetition scheduling. The memory cost of learners is minimized as the goal of spaced repetition software based on stochastic optimal control theory. We derived a mathematically guaranteed scheduling algorithm for minimizing memory cost. SSP-MMC combines psychologically proven theories (e.g., forgetting curve and spacing effect) with modern machine learning techniques to reduce the cost of learners in forming long-term memory. Compared with the HLR, memory models based on time-series are significantly more accurate in predicting users' long-term memory. The stochastic dynamic programming-based spaced repetition scheduling algorithm SSP-MMC outperforms the previous algorithm in all metrics. Experiment results verify the hypothesis that time-series features are very effective in predicting long-term memory. It suggests that the spaced repetition scheduling algorithm based on the time-series model and stochastic optimal control method can effectively predict learners' long-term memory state and improve memory efficiency.

Further work is certainly encouraged to improve time-series-based models by considering the effect of user features on memory state and validating these models beyond language learning applications. In addition, the scenarios where learners use spaced repetition methods are rather diverse. Designing optimization metrics that better help learners archive their goals is yet another area worth further investigation.

Acknowledgments

Thanks to our collaborators at MaiMemo, especially Jun Huang, Jie Mao, and Zhen Zhang, who helped us build the log collecting and processing system.

REFERENCES

[1] H. Ebbinghaus, *Memory: A Contribution to Experimental Psychology*. New York, NY, USA: Teachers College Press, 1913.

[2] N. J. Cepeda, H. Pashler, E. Vul, J. T. Wixted, and D. Rohrer, "Distributed practice in verbal recall tasks: A review and quantitative synthesis," *Psychol. Bull.*, vol. 132, no. 3, pp. 354–380, 2006.

[3] N. J. Cepeda, E. Vul, D. Rohrer, J. T. Wixted, and H. Pashler, "Spacing effects in learning: A temporal ridgeline of optimal retention," *Psychol. Sci.*, vol. 19, no. 11, pp. 1095–1102, Nov. 2008.

[4] S. TJ et al., "Impact of online education on intern behaviour around joint commission national patient safety goals: A randomised trial," *BMJ Qual. Saf.*, vol. 21, no. 10, pp. 819–825, Oct. 2012.

[5] J. K. Maass, P. I. Pavlik, and H. Hua, "How spacing and variable retrieval practice affect the learning of statistics concepts," in *Artif. Intell. Educ.*, 2015, pp. 247–256.

[6] S. K. Carpenter, H. Pashler, and N. J. Cepeda, "Using tests to enhance 8th grade students' retention of u.s. history facts," *Appl. Cogn. Psychol.*, vol. 23, no. 6, pp. 760–771, Sep. 2009.

[7] S. Wan and Z. Niu, "A hybrid E-learning recommendation approach based on learners' influence propagation," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 05, pp. 827–840, May 2020.

[8] A. Cully and Y. Demiris, "Online knowledge level tracking with data-driven student models and collaborative filtering," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 10, pp. 2000–2013, Oct. 2020.

[9] B. Settles and B. Meeder, "A trainable spaced repetition model for language learning," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 1848–1858.

[10] A. N. Rafferty, E. Brunskill, T. L. Griffiths, and P. Shafto, "Faster teaching via POMDP planning," *Cogn. Sci.*, vol. 40, no. 6, pp. 1290–1332, Aug. 2016.

[11] Q. Kang and W. P. Tay, "Sequential multi-class labeling in crowdsourcing," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 11, pp. 2190–2199, Nov. 2019.

[12] Y. Zhang, P. Zhao, Q. Wu, B. Li, J. Huang, and M. Tan, "Cost-sensitive portfolio selection via deep reinforcement learning," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 1, pp. 236–248, Jan. 2022.

[13] J. Ke, F. Xiao, H. Yang, and J. Ye, "Learning to delay in ride-sourcing systems: A multi-agent deep reinforcement learning framework," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 5, pp. 2280–2292, May 2022.

[14] J. R. Anderson, "ACT: A simple theory of complex cognition," *Amer. Psychol.*, vol. 51, no. 4, pp. 355–365, 1996.

[15] M. C. Mozer, H. Pashler, N. Cepeda, R. Lindsey, and E. Vul, "Predicting the optimal spacing of study: A multiscale context model of memory," in *Proc. 22nd Int. Conf. Neural Inf. Process. Syst.*, 2009, pp. 1321–1329.

[16] S. Reddy, S. Levine, and A. Dragan, "Accelerating human learning with deep reinforcement learning," in *Proc. Workshop: Teach. Mach., Robots, Hum.*, 2017, Art. no. 9.

[17] S. Sinha, "Using deep reinforcement learning for personalizing review sessions on E-learning platforms with spaced repetition," Ph.D. dissertation, KTH, School of Electrical Engineering and Computer Science (EECS), 2019.

[18] Z. Yang, J. Shen, Y. Liu, Y. Yang, W. Zhang, and Y. Yu, "TADS: Learning time-aware scheduling policy with dyna-style planning for spaced repetition," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2020, pp. 1917–1920.

[19] P. Woźniak, E. Gorzelańczyk, and J. Murakowski, "Two components of long-term memory," *Acta Neurobiol. Exp.*, vol. 55, no. 4, pp. 301–305, 1995.

[20] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," Oct. 2014, *arXiv:1409.1259*.

[21] J. Ye, J. Su, and Y. Cao, "A stochastic shortest path algorithm for optimizing spaced repetition scheduling," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, New York, NY, USA, 2022, pp. 4381–4390. [Online]. Available: https://doi.org/10.1145/3534678.3539081

[22] P. I. Pavlik and J. R. Anderson, "An ACT-R model of the spacing effect," in *Proc. 5th Int. Conf. Cogn. Model.*, 2003, pp. 177–182.

[23] P. Pimsleur, "A memory schedule," *Modern Lang. J.*, vol. 51, no. 2, pp. 73–75, Feb. 1967.

[24] S. Leitner, *So Lernt Man Leben*, 1st ed., Munich, Germany: Droemer-Knaur, 1974.

[25] P. A. Woiniak and E. J. Gorzelanczyk, "Optimization of repetition spacing in the practice of learning," *Acta Neurobiol. Exp.*, vol. 54, no. 2, pp. 59–62, 1994.

[26] S. Reddy, I. Labutov, S. Banerjee, and T. Joachims, "Unbounded human learning: Optimal scheduling for spaced repetition," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 1815–1824.

[27] B. Tabibian, U. Upadhyay, A. De, A. Zarezade, B. Schölkopf, and M. Gomez-Rodriguez, "Enhancing human learning via spaced repetition optimization," in *Proc. Nat. Acad. Sci.*, 2019, pp. 3988–3993.

[28] U. Upadhyay, G. Lancashire, C. Moser, and M. Gomez-Rodriguez, "Large-scale randomized experiments reveals that machine learning-based instruction helps people memorize more effectively," *NPJ Sci. Learn.*, vol. 6, no. 1, pp. 1–3, Sep. 2021.

[29] A. Hunziker et al., "Teaching multiple concepts to a forgetful learner," in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 4048–4058.

[30] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[31] A. Nioche, P.-A. Murena, C. de la Torre-Ortiz, and A. Oulasvirta, "Improving artificial teachers by considering how people learn and forget," in *Proc. 26th Int. Conf. Intell. User Interfaces*, 2021, pp. 445–453.

[32] U. Upadhyay, A. De, and M. Gomez-Rodrizuez, "Deep reinforcement learning of marked temporal point processes," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 3172–3182.

[33] R. A. Bjork et al., "A new theory of disuse and an old theory of stimulus fluctuation," *Learn. Processes Cogn. Processes: Essays Honor William K. Estes*, vol. 2, pp. 35–67, 1992.

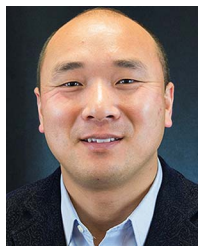
[34] A. W. Melton, "The situation with respect to the spacing of repetitions and memory," *J. Verbal Learn. Verbal Behav.*, vol. 9, no. 5, pp. 596–606, Oct. 1970.

[35] I. P. Androulakis, "Dynamic programming: Stochastic shortest path problems," in *Encyclopedia of Optimization*, Berlin, Germany: Springer, 2009, pp. 869–873.

[36] G. B. Maddox, D. A. Balota, J. H. Coane, and J. M. Duchek, "The role of forgetting rate in producing a benefit of expanded over equal spaced retrieval in young and older adults," *Psychol. Aging*, vol. 26, no. 3, pp. 661–670, 2011.

[37] D. Haussler, "Probably approximately correct learning," in *Proc. 8th Nat. Conf. Artif. Intell.*, 1990, pp. 1101–1108.

[38] P. A. Wozniak, "Optimization of learning," 1990. [Online]. Available: http://super-memory.com/english/ol.htm



Jingyong Su received the BE and MS degrees in electrical engineering from the Harbin Institute of Technology, in 2006 and 2008, respectively, and the PhD degree in statistics from Florida State University, in 2013. He is a professor of computer science with the Harbin Institute of Technology at Shenzhen, China. He joined the Department of Mathematics & Statistics, Texas Tech University, in 2013 as an Assistant Professor and became tenured in 2019. His areas of research include computer vision, medical image analysis, functional and shape data analysis.

Junyao Ye received the BE degree in computer science from the Harbin Institute of Technology (Shenzhen), in 2022. He is a research engineer with MaiMemo Inc., Qingyuan, China. His main research interests lie within educational data mining, personalized learning and adaptive systems.

Liqiang Nie (Senior Member, IEEE) received the BEng and PhD degree from the Xi'an Jiaotong University and National University of Singapore (NUS), respectively. He is a fellow of AAIA and IAPR, and currently the dean with the School of Computer Science and Technology, Harbin Institute of Technology (Shenzhen campus). His research interests lie primarily in multimedia content analysis and information retrieval. He has co-/authored more than 100 CCF-A papers and 5 books, with 18 k plus Google Scholar citations. He is an AE of *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on Multimedia*, *IEEE Transactions on Circuits and Systems for Video Technology*, *ACM Transactions on Multimedia Computing, Communications, and Applications*, and *Information Science*. Meanwhile, he is the regular area chair or SPC of ACM MM, NeurIPS, IJCAI, AAAI and ICML. He is a member of ICME steering committee. He has received many awards, like SIGMM emerging leaders in 2018, ACM MM and SIGIR best paper honorable mention in 2019, SIGMM rising star in 2020, MIT TR35 China 2020, DAMO Academy Young fellow in 2020, SIGIR best student paper in 2021, ACM MM best paper award in 2022, first price of the provincial science and technology progress award in 2021 (rank 1), provincial youth science and technology award in 2022, AI's 10 to Watch in 2022. Some of his research outputs have been integrated into the products of some listed companies.

Yilong Cao received the BE and PhD degrees in electronic engineering from the University of Sheffield, in 2007 and 2013, respectively. He is a co-founder of Maimemo Inc. His areas of research include genetic programming, loss function analysis, and educational data mining.

Yongyong Chen received the BS and MS degrees from the Shandong University of Science and Technology, Qingdao, China, in 2014 and 2017, respectively, and the PhD degree from the University of Macau, Macau, in 2020. He is currently an assistant professor with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, China. He has published more than 50 research papers in top-tier journals and conferences, including *IEEE Transactions on Image Processing*, *IEEE Transactions on Information Forensics and Security*, *IEEE Transactions on Dependable and Secure Computing*, *IEEE Transactions on Neural Networks and Learning Systems*, *IEEE Transactions on Multimedia*, *IEEE Transactions on Circuits and Systems for Video Technology*, *IEEE Transactions on Geoscience and Remote Sensing*, *IEEE Transactions on Computational Imaging*, *IEEE Journal of Selected Topics in Signal Processing*, *Pattern Recognition* and ACM MM. His research interests include image processing, data mining, and computer vision.