



A Stochastic Shortest Path Algorithm for Optimizing Spaced Repetition Scheduling

Junyao Ye

Jingyong Su*

jarrett.ye@outlook.com

sujingyong@hit.edu.cn

Harbin Institute of Technology, Shenzhen
Shenzhen, Guangdong, China

Yilong Cao

MaiMemo Inc.

Qingyuan, Guangdong, China

yl.cao@aimemo.com

ABSTRACT

Spaced repetition is a mnemonic technique where long-term memory can be efficiently formed by following review schedules. For greater memorization efficiency, spaced repetition schedulers need to model students' long-term memory and optimize the review cost. We have collected 220 million students' memory behavior logs with time-series features and built a memory model with Markov property. Based on the model, we design a spaced repetition scheduler guaranteed to minimize the review cost by a stochastic shortest path algorithm. Experimental results have shown a 12.6% performance improvement over the state-of-the-art methods. The scheduler has been successfully deployed in the online language-learning app MaiMemo to help millions of students.

CCS CONCEPTS

- Applied computing → E-learning; • Computing methodologies → Dynamic programming for Markov decision processes.

KEYWORDS

spaced repetition, optimal control, language learning

ACM Reference Format:

Junyao Ye, Jingyong Su, and Yilong Cao. 2022. A Stochastic Shortest Path Algorithm for Optimizing Spaced Repetition Scheduling. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA*. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3534678.3539081>

1 INTRODUCTION

To form long-term memory, students should repeatedly review learned materials. According to the primary memory experiment by Ebbinghaus[6], the number of repetitions, the interval between

*Also with Chongqing Research Institute of Harbin Institute of Technology, Industrial Vision Research Center.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

KDD '22, August 14–18, 2022, Washington, DC, USA

© 2022 Association for Computing Machinery.

ACM ISBN 978-1-4503-9385-0/22/08...\$15.00

<https://doi.org/10.1145/3534678.3539081>

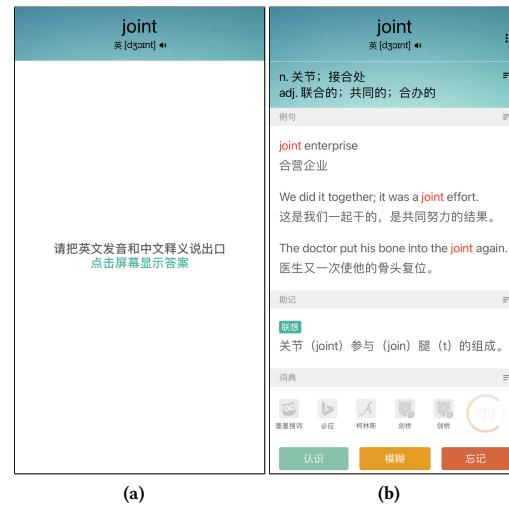


Figure 1: The vocabulary flashcard in MaiMemo

repetitions, and other factors during the review affect one's memories. These factors have been investigated in a large number of psychological experiments[4] and summarized into some effects, such as *spacing effect*[5] and *lag effect*[10]. *Spaced repetition* is a mnemonic technique employing these effects to improve memory efficiency. Spacing out each review can effectively enhance long-term memory. Spaced repetition has a significant effect on second language acquisition and many studies show that spaced repetition can also work in the fields of medicine[18], statistics[9], history[3], etc.

In most spaced repetition practices, the materials are presented in flashcards, as shown in Figure 1, with simple schedules determining when each flashcard will be reviewed. The problem of designing a more efficient spaced repetition scheduling algorithm has a rich history. From Leitner's box[8] to SuperMemo[20] which is the first spaced repetition software, they use simple rule-based heuristics with the designers' experience and personal experiments. Many spaced repetition software (Anki, Mnemosyne, etc.) still use these algorithms. Due to the hard-coded parameters and the lack of theoretical proof, these algorithms cannot adjust to different learners and materials. Meanwhile, their performance cannot be guaranteed.

With the popularity of e-learning platforms, vast amounts of data about students' reviews could be collected. It enables researchers

to design trainable, adaptive, and guaranteed algorithms. An efficient spaced repetition scheduling algorithm can save millions of users' time and help them remember more words in MaiMemo, a language learning application that supports learners in memorizing vocabulary. Recently, some works employed machine learning to determine the optimal review times. However, due to the following three reasons, the methods do not fit our conditions:

- Lack of time-series information: Several works, such as HLR (Half-life Regression)[14] model and EFC (Exponential Forgetting Curve)[12] model, use the forgetting curve to relate the probability of recall with the time elapsed since the last review. But they ignore the effect of interval on memory strength. In their models, memory strength is a function of the number of repetitions. According to the spacing effect[5] and the data we collected, the interval between reviews has a significant impact on the formation of long-term memory.
- Lack of user-perceived optimization goals: HLR [14] and the improved C-HLR[22] model based on it, aim to accurately predict the recall probability of memory. Reducing the pressure of review, maximizing memory, and other relevant indicators to users are not considered.
- Lack of explainability: Some algorithms[13, 15, 19, 21] based on deep reinforcement learning are a black box for designers and lack explainability. Providing explanation during the model scheduling the review of students is helpful for educational research[16].

In this paper, we have established the DHP (Difficulty-Halflife-P(recall)) model for simulating the user's long-term memory based on the collected millions of memory data. We set a clear optimization goal with practical significance: to minimize the memorization cost for users to form long-term memory. To achieve this goal, we propose a novel spaced repetition scheduling algorithm, namely SSP-MMC (Stochastic-Shortest-Path-Minimize-Memorization-Cost).

Our work provides a long-term memory model more approximate to the natural environment for spaced repetition schedulers, tested by data from the real world. We find a possible optimization problem for spaced repetition scheduling and the corresponding optimal method. We make the data and software tools publicly available to facilitate reproduction and follow-up research (see Section 7). To summarize, the main contributions of this paper are:

- Build and publicly release our spaced repetition log dataset, which is the first to contain time-series information.
- This is the first work to employ the time-series feature to model long-term memory to the best of our knowledge.
- The empirical results demonstrate that SSP-MMC outperforms state-of-the-art baselines in minimizing the memorization cost.

2 RELATED WORK

There is already a large amount of literature in terms of optimizing spaced repetition, from modeling and predicting learners' long-term memory to designing optimization scheduling algorithms based on related memory models.

Spaced repetition and memory models. ACT-R (Adaptive Character of Thought-Rational model)[1] is cognitive architecture

including a declarative memory module. It assumes that each exercise produces a power forgetting curve and that multiple power functions approximate the forgetting curve in spaced repetition. MCM (Multiscale Context Model)[11] integrates two theories of spacing effect and uses multiple exponential functions to approximate the forgetting curve. HLR[14] is a trainable spaced repetition model to predict the half-life of memory. A work[22] improves the model, introduces neural networks, and includes more psychological and linguistic features. However, these models ignore time-series information like the order of historic recall result sequence and the interval between every successive review.

Optimal scheduling with spaced repetition models. To balance the learning of new material and the reviewing of learned material, Reddy et al.[12] propose a queueing network model for the Leitner System[8] and designed a heuristic algorithm for scheduling review. However, the algorithm is based on the EFC model taking memory strength as a function of the number of reviews and position of Leitner Box instead of the interval between repetitions. Tabibian et al.[17] introduce marked temporal point processes to represent spaced repetition and regard the scheduling as an optimal control problem. They come up with the tradeoff between recall probability and the number of reviews. Due to the limit of the dataset, the forgetting rate in their model is affected only by the recall result. In the work of Hunziker et al.[7], optimizing spaced repetition scheduling boiled down to stochastic sequence optimization problems. They designed a greedy algorithm to achieve high performance in maximizing learners' retention. But the sufficient conditions for the algorithm to achieve a high utility are strict and may not be applicable in most cases. Recently, reinforcement learning has also been applied to optimize spaced repetition scheduling. A number of papers[13, 15, 19, 21] used RL to maximize learners' memory expectations by designing rewards and training in a simulated environment. However, the environments on which their algorithms were based are oversimplified, making them not well suited to the real world. For example, in [13]'s work, they assumed that the student could only learn one item per step and the intervals between steps are constant. In addition to that, these algorithms lack explainability, and their performance varies in different simulation environments and is not always superior to heuristic algorithms.

3 LONG-TERM MEMORY MODEL

To design a high-efficiency, guaranteed, interpretable spaced repetition scheduling algorithm, we collect learners' memory behavior data and then use the data to validate several psychological effects. Using these effects, we model memory to understand how scheduling in spaced repetition impacts the learner's memory state.

3.1 Building Dataset

We have collected one month of MaiMemo's logs containing 220 million memory behavior data from 828 thousand students to build a model to simulate learners' memory. For the following reasons, we do not use Duolingo's open-source dataset:

- It lacks time-series aspects such as sequences of feedback and interval, and our analysis of the data shows that historical features have a substantial impact on memory state.

- Its recall probability definition is questionable. Duolingo defines the recall probability as the proportion of times a word is correctly recalled in a given session, implying that multiple memory behaviors of the same word in a given session are independent. However, the first recall will influence the learners' memory state and subsequent memories throughout the day.

The historical information of learners' word memory is recorded in MaiMemo's logs. We use a quad to describe a memory behavior event:

$$e := (u, w, t, r), \quad (1)$$

which means that the user u reviewed the word w at time t and either recalled it ($r = 1$) or forgot it ($r = 0$).

To consider the times-series of memory behavior events, we include historical features:

$$e_i := (u, w, \Delta t_{1:i-1}, r_{1:i-1}, \Delta t_i, r_i), \quad (2)$$

where e_i is the i th memory behavior event that the user u reviewed the word w and Δt is the time elapsed from the last review. $\Delta t_{1:i-1}$ and $r_{1:i-1}$ are the interval sequence and recall sequence from 1st to $i - 1$ th.

We filter out the logs of which the first feedback is $r = 1$ to exclude the influence of memories that learners have formed before using spaced repetition. The sample logs of memory behavior events of learners are shown in Table 1.

Table 1: Samples of dataset

u	w	$\Delta t_{1:i-1}$	$r_{1:i-1}$	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

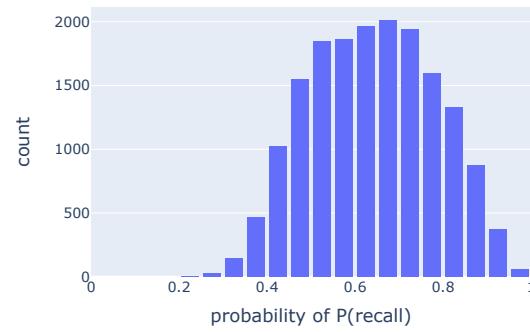
Based on the above dataset, we verify two psychological phenomena that are significant in spaced repetition.

3.2 Forgetting Curve and Spacing Effect

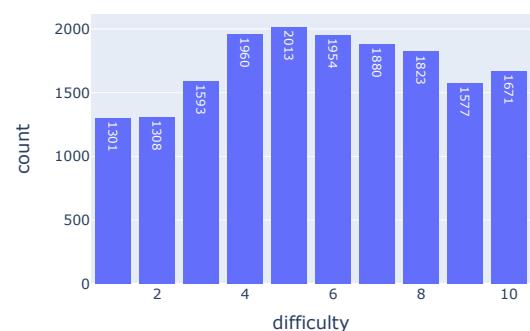
The term "forgetting curve" describes how memory decay over time once a learner stops reviewing. In the above memory behavior events, recall is binary (i.e., a user either recalls or forgets a word). To capture memory decay, we need to get the probability underlying recall. We use the recall ratio $n_{r=1}/N$ in a group of N individuals learning words as the recall probability p :

$$e_i := (w, \Delta t_{1:i-1}, r_{1:i-1}, \Delta t_i, p_i, N). \quad (3)$$

By controlling the w , $\Delta t_{1:i-1}$ and $r_{1:i-1}$, we can plot the p for each Δt to obtain the forgetting curve. The ratio $n_{r=1}/N$ approaches the recall probability with N large enough. 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. Given that we're interested in the forgetting curve, material difficulty significantly influences forgetting speed. 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 difficulty of words. Figure 2(a) depicts the recall ratio distribution.



(a)



(b)

Figure 2: The distributions of P(recall) and difficulty. The easiest words with $P(\text{recall}) > 0.85$ are assigned $d = 1$ and the hardest words $P(\text{recall}) \leq 0.45$ are assigned $d = 10$. The difficulties of the remaining words are assigned from 2 to 9 by dividing remaining interval of $P(\text{recall})$ equally into eight parts.

Table 2: Samples of words' difficulty

w	d	$\Delta t_{1:i-1}$	$r_{1:i-1}$	Δt_i	p_i	N
automobile	2	0	0	1	0.8379	7501
multiply	6	0	0	1	0.635	7495
linger	9	0	0	1	0.4925	7493
hatch	10	0	0	1	0.4371	7492

We can see from the data distribution that the recall ratio is mostly between 0.45 and 0.85. For the balance and density of grouping data, the words are divided into ten difficulty groups. The results of grouping is shown in Figure 2(b) and Table 2. The symbol d indicates the difficulty, with the larger the number, the greater the difficulty.

Thus, the grouping memory behavior event can be represented as:

$$e_i := (d, \Delta t_{1:i-1}, r_{1:i-1}, \Delta t_i, p_i, N). \quad (4)$$

By grouping, we can draw forgetting curves with enough data. We use the exponential forgetting curve function $p_i = 2^{-\Delta t_i/h_i}$ to

fit them and calculate the half-life h_i of memory:

$$e_i := (d, \Delta t_{1:i-1}, r_{1:i-1}, h_i, N). \quad (5)$$

The spacing effect states that as the time between review sessions rises, so does memory consolidation[5]. In our dataset, we can use the ratio of the half-life before and after a review to indicate the consolidation degree of memory.

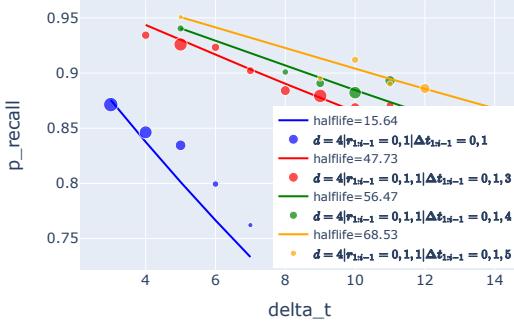


Figure 3: Forgetting curves of different time series.

The scatter points in Figure 3 reflect the proportion of recall corresponding with each interval, and the curves are the results of fitting these points using an exponential function. The various colors denote the different control situations. From the various memory behavior events, we can observe that a successful recall lengthens the memory half-life. The effect of memory consolidation rises as the review interval extends.

Based on the above observations, we propose the DHP (Difficulty-Halflife-P (Recall)) Model to fit and interpret the existing data.

3.3 Difficulty-Halflife-P(recall) Model

Current advanced approaches for processing time-series data are based on recurrent neural networks like LSTM and GRU, which can predict the half-life using time-series features. However, introducing the neural network will make our algorithm unexplainable. Instead, We develop a time-series model with the Markov property for explainability and simplicity. We reduce the dimensionality of time-series into state variables and state-transition equations in this model. We consider the following four variables:

- Half-life. It measures the storage strength of memory.
- Recall probability. It measures the retrieval strength[2] of memory. According to the spacing effect, the interval between each review affects the half-life. When h is fixed, Δt and p are mapped one-to-one, and for normalization, we use $p = 2^{-\Delta t/h}$ instead of it as a state variable.
- Result of recall. The half-life increases after recall and decreases after forgetting.
- Difficulty, the higher the difficulty, the harder the memory to consolidate.

The time series are projected into 3D space with the last half-life, recall probability, and half-life as the XYZ axis, and the difficulty is represented by the color, as illustrated in Figure 4.

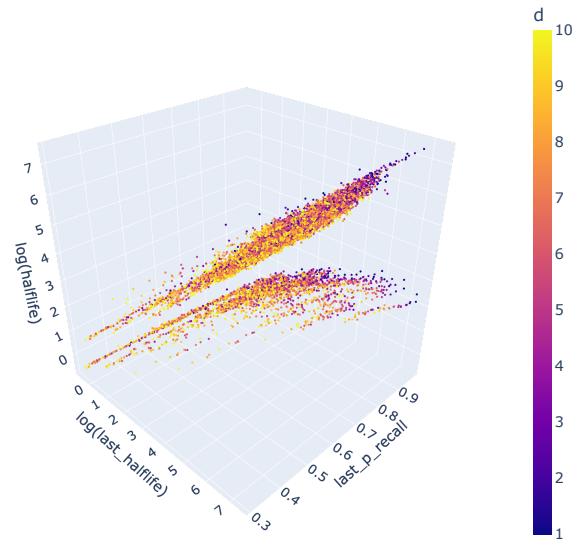


Figure 4: The projection in Difficulty, Half-life, and P(recall). The `last_halflife` and `halflife` denote h_{i-1} and h_i . Similarly, `last_p_recall` denotes p_{i-1} .

By observing the projection of the data, we notice two phenomena: $h_i > h_{i-1}$ when $r_i = 1$ and $h_i > 0$ when $r_i = 0$. In the state-transition equation, we account for these two constraints:

$$h_i = [h_{i-1} \cdot (\exp(\theta_1 \cdot \mathbf{x}_i) + 1), \exp(\theta_2 \cdot \mathbf{x}_i)] \cdot [r_i, 1 - r_i]^T, \quad (6)$$

where $\mathbf{x}_i = [\log d_{i-1}, \log h_{i-1}, \log(1 - p_i)]$ is the feature vector.

Even under the identical recall probability and last half-life conditions, we find that once learners forget during the review, the half-life of subsequent recall success is shorter. We attribute this to the fact that more difficult material is more likely to be forgotten. So the forgotten material is more difficult than remembered material. As a result, a new state-transition equation for the difficulty is added:

$$d_i = [d_{i-1}, d_{i-1} + \theta_3^1] \cdot [r_i, 1 - r_i]^T. \quad (7)$$

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

$$\begin{aligned} \begin{bmatrix} h_i \\ d_i \end{bmatrix} &= \begin{bmatrix} h_{i-1} \cdot (\exp(\theta_1 \cdot \mathbf{x}_i) + 1) & \exp(\theta_2 \cdot \mathbf{x}_i) \\ d_{i-1} & d_{i-1} + \theta_3 \end{bmatrix} \begin{bmatrix} r_i \\ 1 - r_i \end{bmatrix} \\ \mathbf{x}_i &= [\log d_{i-1}, \log h_{i-1}, \log(1 - p_i)] \\ r_i &\sim \text{Bernoulli}(p_i) \\ p_i &= 2^{-\Delta t_i/h_{i-1}} \\ h_1 &= -1/\log_2(0.925 - 0.05 \cdot d_0), \end{aligned} \quad (8)$$

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

¹To prevent the difficulty from increasing indefinitely, we set an upper limit.

Based on Equation 8 and the initial values of difficulty d_0 , the half-life h_i of any memory behaviors can be calculated. The calculation process is shown in Figure 5.

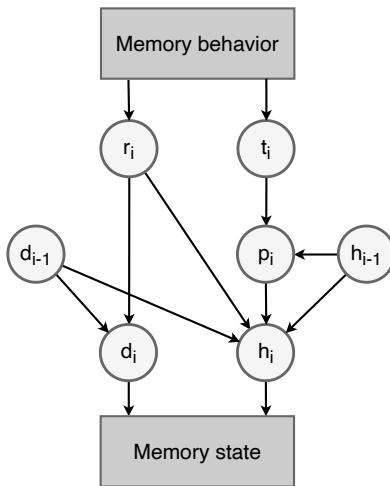


Figure 5: The structure of the DHP model

4 OPTIMAL SCHEDULING

With the DHP model describing learners' long-term memory, we can use it to simulate the performance of spaced repetition scheduling algorithms. Before designing optimal scheduling, we need to set a goal.

4.1 Problem Setup

The purpose of the spaced repetition method is to help learners form long-term memory efficiently. Whereas the memory half-life measures the storage strength of long-term memory, the number of repetitions and the time spent per repetition reflect the cost of memory. Therefore, the goal of spaced repetition scheduling optimization can be formulated as either getting as much material as possible to reach the target half-life within a given memory cost constraint or getting a certain amount of memorized material to reach the target half-life at minimal memory cost. Among them, the latter problem can be simplified as how to make one memory material reach the target half-life at the minimum memory cost (MMC).

The long-term memory model we constructed in Section 3.3 satisfies the Markov property. In the DHP model, the state of each memory depends only on the last half-life, the difficulty, the current review interval, and the result of the recall, which follows a random distribution relying on the review interval. Due to the randomness of half-life state-transition, the number of reviews required for memorizing material to reach the target half-life is uncertain. Therefore, the spaced repetition scheduling problem can be regarded as an infinite-time stochastic dynamic programming problem. In the case of forming the long-term memory, the problem has a termination state which is the target half-life. So it is a Stochastic Shortest Path (SSP) problem. Combining with the optimization goal, we name the algorithm as SSP-MMC.

4.2 Stochastic Shortest Path Algorithm

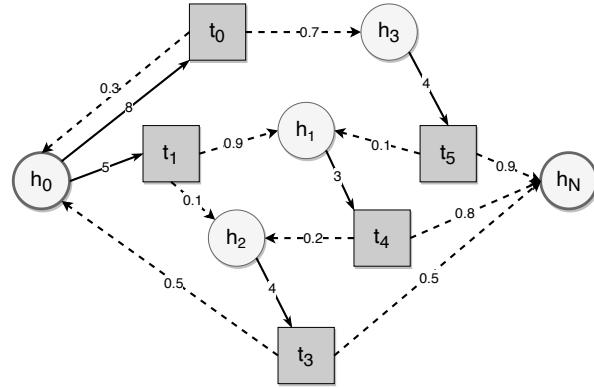


Figure 6: A graphical representation of SSP

As shown in Figure 6, without considering the state of difficulty, circles are half-life states, squares are review intervals, dashed arrows indicate half-life state transitions for a given review interval, and black edges represent review intervals available in a given memory state with the corresponding cost of review. The stochastic shortest path problem in spaced repetition is how the scheduler can suggest the optimal review interval to minimize the expected review cost for reaching the target half-life.

Symbol definition:

- h_0 - Initial half-life
 - h_N - target half-life
 - $h_{k+1}, d_{k+1} = f(h_k, d_k, \Delta t_k, r_k)$ - memory state-transition equation
 - $g(h_k, d_k, \Delta t_k, r_k)$ - review cost²
 - $\Delta T(h_k, d_k)$ - the set of optional review intervals corresponding to the current memory state.
 - $J_\pi(h_0, d)$ - Total review cost for each difficulty.

The Bellman's equation for SSP-MMC:

$$J(h_k, d_k) = \min_{\Delta t_k \in \Delta T(h_k, d_k)} E[g(r_k) + J(f(h_k, d_k, \Delta t_k, r_k))]. \quad (9)$$

Based on this equation, it can be solved iteratively by stochastic dynamic programming to obtain the optimal review interval Δt_k corresponding to each h_k, d_k . Since h is a continuous value, which is not conducive to recording states, it can be discretized as follows:

$$h_{index} = \lfloor \log(h)/\log(\text{base}) \rfloor, \quad (10)$$

where base is used to control the size of the h bins. In the state-transition equation, h is approximately exponentially growing. So the function log is used to reduce the sparsity in the range of high h .

Since there is an upper difficulty limit d_N and a termination half-life h_N , we can build a cost matrix $J[d_N][h_N]$ and initialize it to inf. Set $J[d_i][h_N] = 0$ and start iterating through each (d_k, h_k) from

²For simplicity, we only consider the r_k : $g(r_k) = a \cdot r_k + b \cdot (1 - r_k)$, a is the cost of recall success and b is the cost of recall failure.

$\Delta T(h_k, d_k)$ to each Δt_k and compute $p_k, h_{r_k=1}, h_{r_k=0}, d_{r_k=1}, d_{r_k=0}$, and then use the following equation

$$\begin{aligned} J[h_k][d_k] = \min_{p_k} & [p_k \cdot (g(r_k = 1) + J[h_{r_k=1}][d_{r_k=1}]) \\ & + (1 - p_k) \cdot (g(r_k = 0) + J[h_{r_k=0}][d_{r_k=0}])] \end{aligned} \quad (11)$$

to iteratively update the optimal cost corresponding to each memory state. A strategy matrix is used to record the optimal review interval in each state. The optimal strategy converges finally over iterations. SSP-MMC is given in Algorithm 1.

Algorithm 1: SSP-MMC

```

Data:  $a, b, d_N, h_N$ 
Result:  $\pi^*[d_N][h_N], J[d_N][h_N]$ 
1 initialization;
2 for  $d \leftarrow d_N$  to  $d_0$  do
3    $J[d][h_0 : h_{N-1}] = \inf;$ 
4    $J[d][h_N] = 0;$ 
5   while  $\Delta J < 0.1$  do
6      $J_{h_0} = J[d][h_0];$ 
7     for  $h \leftarrow h_{N-1}$  to  $h_0$  do
8       foreach  $\Delta t \in \Delta T(d, h)$  do
9          $p \leftarrow 2^{-\frac{\Delta t}{h}};$ 
10         $h_{r=1} \leftarrow f(h, p, d, 1);$ 
11         $h_{r=0} \leftarrow f(h, p, d, 0);$ 
12         $J \leftarrow p \cdot (a + J[d][h_{r=1}])$ 
13         $+ (1 - p) \cdot (b + J[d+2][h_{r=0}]);$ 
14        if  $J < J[d][h]$  then
15           $J[d][h] = J;$ 
16           $\pi^*[d][h] = \Delta t;$ 
17        end
18      end
19       $\Delta J = J_{h_0} - J[d][h_1];$ 
20    end
21 end

```

5 EXPERIMENT

Our experiments are designed to answer the following questions:

- How well does the DHP model simulate long-term memory?
- What is the practical significance of the weight parameters of the DHP model?
- What are the patterns of optimal review intervals given by the SSP-MMC algorithm?
- How does the SSP-MMC algorithm improve on different metrics compared to baselines?

In this section, we train the DHP model first based on the dataset collected in Section 3.3 and compare it with the HLR model. The parameters of the DHP model are also visualized to obtain an intuitive interpretation of the memory. Then, we use the DHP model as a training environment for the SSP-MMC algorithm to get the optimal policy and visualize it. Finally, we compare the performance of the SSP-MMC with different baselines in a simulation environment composed of the DHP model.

5.1 DHP Model Weight Analysis

To have a better understanding of the DHP model, we compare it with the HLR[14] model in terms of predicted half-life and analyze the model weights.

Table 3: Evaluation results

Method	Recall ¹		Forget ²	
	MAE ³	MAPE ⁴	MAE	MAPE
DHP	11.93	26.16%	0.44	31.99%
HLR	41.70	45.75%	7.08	112.7%

¹ Recall: half-life after recall.

² Forget: half-life after forget.

³ MAE: mean absolute error.

⁴ MAPE: mean absolute percentage error.

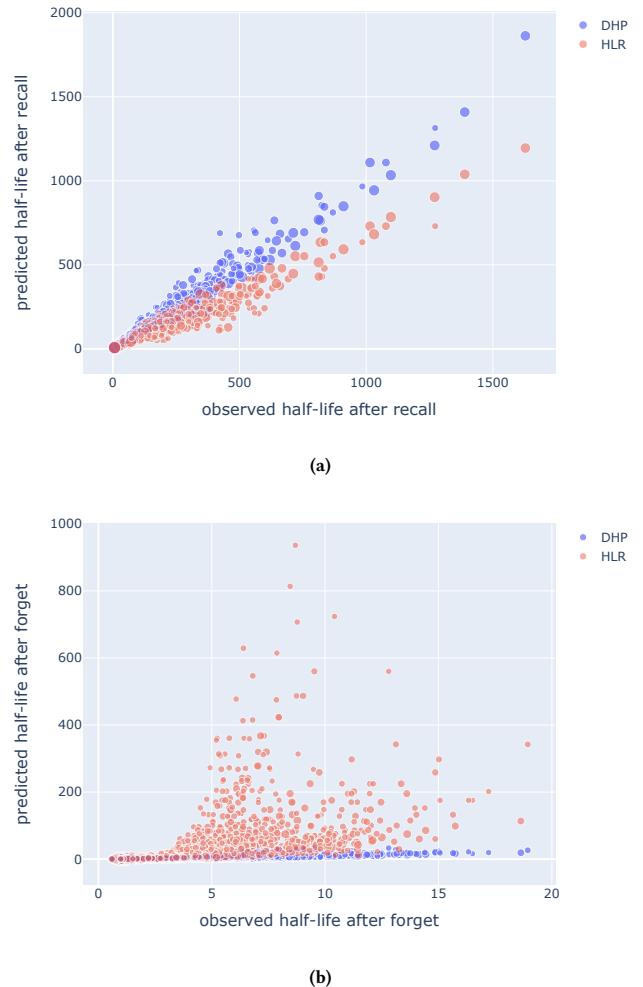


Figure 7: The results of fitting

The results of fitting are shown in Figure 7. From the comparison in Figure 7(a) and 7(b), we can find that HLR is under-fitted in

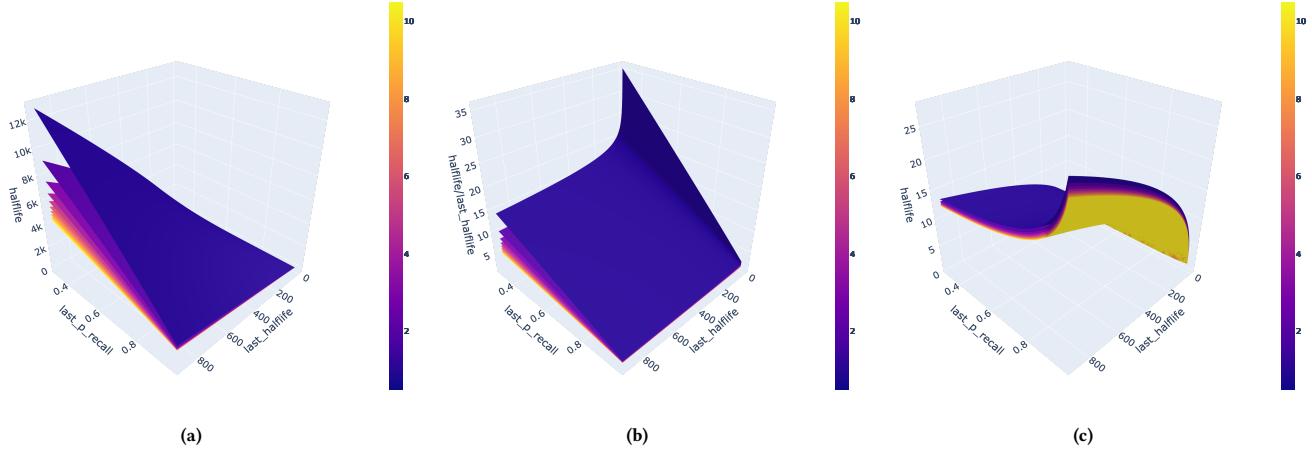


Figure 8: The projection of the DHP model

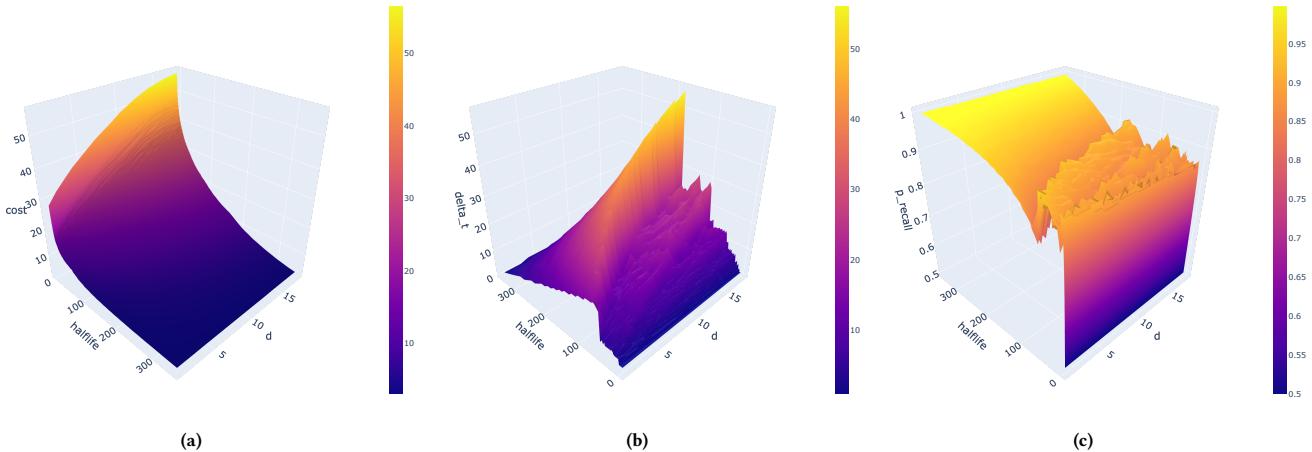


Figure 9: The optimal policy

predicting the half-life, and the prediction error (shown in Table 3) of the DHP model is significantly smaller than that of the HLR model, especially in the result of Forget. It is most likely because HLR drops the time-series information, making it impossible to distinguish memory behaviors such as $r_{1:2} = 1, 0$ and $r_{1:2} = 0, 1$. According to the parameters obtained by fitting and the equations of the model, we get the half-life after a successful recall:

$$h_{i+1} = h_i \cdot (\exp(3.81) \cdot d_i^{-0.534} \cdot h_i^{-0.127} \cdot (1 - p_i)^{0.970} + 1). \quad (12)$$

The Figure 8(a) illustrates that the half-life after successful recall increases as p_i decreases, which verifies the existence of the spacing effect (see §3.2). Figure 8(b) shows that the increasing multiplication of the half-life decrease as the last half-life increases, which may imply that our memory consolidation decreases as the memory strength increases, i.e., there is a marginal effect.

Similarly, the half-life after a failed recall is:

$$h_{i+1} = \exp(-0.041) \cdot d_i^{-0.041} \cdot h_i^{0.377} \cdot (1 - p_i)^{-0.227}. \quad (13)$$

Figure 8(c) plots that the longer the half-life of memory, the longer its half-life after a failed recall, which may be because the memory is not entirely lost in forgetting. And as recall probability decreases, the half-life after a failed recall also decreases, which could be due to the memory being forgotten more entirely over time.

5.2 SSP-MMC Optimal Policy Analysis

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

The cost of review decreases as the half-life increases and increases as the difficulty increases as shown in Figure 9(a). Memories

with a high half-life reach the target half-life at a lower expected cost. In addition, memories with greater difficulty have a higher expected cost because they have a lower half-life growth as shown in Figure 8(b) and require more review to reach the target half-life.

The interval increases with difficulty for the same level of memory half-life as shown in Figure 9(b). This may be because forgetting raises the difficulty of simple memories and decreases their half-life, 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 bit of half-life boost. The interval reaches its peak in the midrange of half-life. It is necessary to compare with Figure 9(c) to explain the peak.

The recall probability corresponding to optimal interval increases with half-life and decreases with difficulty, as shown in Figure 9(c). It means that the scheduler will instruct learners to review at a lower retrieval strength in the early stages of memorization, which may be a reflection of "desirable difficulties" [2]. As the half-life 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 , Δt is first increasing and then decreasing where the peak emerges.

5.3 Offline Simulation

We compare five baseline policies: RANDOM, ANKI, HALF-LIFE, THRESHOLD, and MEMORIZE; and three metrics: target half-life reached, summation of recall probability, and words total learned.

Environments. Our simulation environment has the following parameters: Target half-life h_N , Recall success/failure cost, Daily cost limit, and Simulation duration.

Baselines. We compare SSP-MMC with five baseline scheduling algorithms:

- RANDOM, which chooses a random interval from $[1, h_N]$ to schedule the review.
- ANKI, a variant of SM-2[20], not require environmental parameters.
- HALF-LIFE, the half-life is used as the review interval.
- THRESHOLD, review when p is less than or equal to a certain level (we use 90% which is the default in SuperMemo).
- MEMORIZE, an algorithm based on optimal control, with code from the open-source repository of Tabibian et al. [17].

Metric. Our evaluation metrics include:

- **THR** (target half-life reached) is the number of words that reach the target half-life.
- **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 half-life to simulate that the learner has formed a solid long-term memory.
- **WTL** (words total learned) is the number of total learned words.

Implementation details. We set a recall half-life of 360 days (near one year) as the target half-life, and when the half-life of the ITEM exceeds this value, it will not be scheduled for review. Then, considering that the daily learning time of learners in the actual scenario is roughly constant, we set 600s (10min) as the upper limit of daily learning cost. When the accumulated cost during each learning and review exceeds this limit, the review task is postponed to the next day regardless of whether it is completed or not, ensuring that each algorithm is compared at the same memory cost. We use

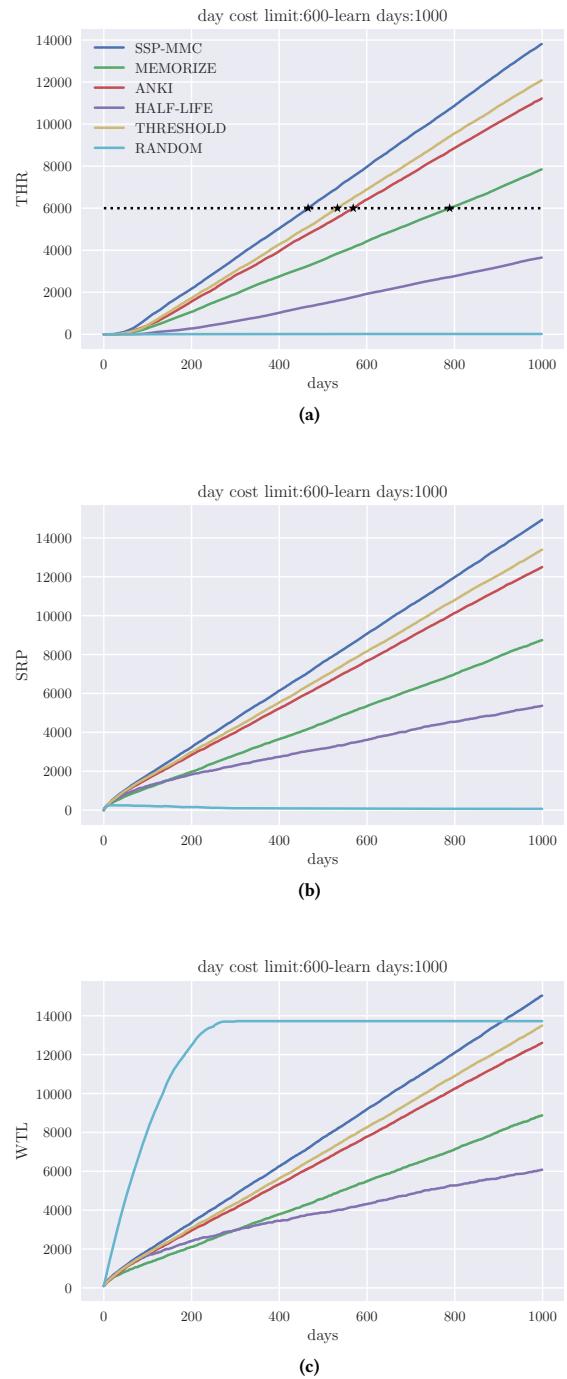


Figure 10: The results of simulation

the average time spent by learners of 3s for successful recall and 9s for failed recall. Then, language learning is a long-term process, and we set a simulation duration of 1000 days.

Analysis. The simulation results in Figure 10 show that:

According to THR, SSP-MMC performs better than all baselines, which is not surprising. THR is consistent with the optimization goal of SSP-MMC, and SSP-MMC can reach the upper bound of this metric.

To quantify the relative difference between the performance of each algorithm, we compare the number of days for $\text{THR} = 6000$ (marked as \star in Figure 10(a)): 466 days for SSP-MMC, 569 days for ANKI, 533 days for THRESHOLD, and 793 days for MEMORIZER. Compared to the THRESHOLD, SSP-MMC saves **12.6%** of the time of review.

The result on SRP is similar to that on THR. This means that the learner following the schedules of the SSP-MMC will remember the most.

On WTL, RANDOM beats all algorithms in the early stage because the learner can keep learning new words as long as the scheduling algorithm does not schedule a review, but this is at the cost of forgetting the already learned words. Besides, the SSP-MMC outperforms other baselines because it minimizes the cost of memorization and gives learners more time to learn new words.

6 CONCLUSION

We establish the first memory dataset containing complete time-series information, design a long-term memory model based on time-series information that can fit existing data well, and provide 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 derive a mathematically guaranteed scheduling algorithm for minimizing memory cost. SSP-MMC combines the psychologically proven theories of 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 model, the DHP model's accuracy is significantly improved in fitting the user's long-term memory. Moreover, the SSP-MMC scheduling algorithm outperforms the baselines. The algorithm was deployed in MaiMemo to improve long-term memory efficiency for users. We provide technical details of the design and deployment in Appendix A.

The main future work is to improve the DHP model by considering the effect of user features on memory state and validating the model in spaced repetition software beyond language learning. In addition, the scenarios in which learners use spaced repetition methods are diverse, and designing optimization metrics that match learners' goals is also an issue worth investigating.

7 DATA AND CODES

To facilitate research in this area, we have publicly released the dataset and codes used in this paper at:

<https://github.com/maimemo/SSP-MMC>

ACKNOWLEDGMENTS

This work was supported by the GuangDong Basic and Applied Basic Research Foundation under Grant 2022A1515010800. 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] John R. Anderson, Daniel Bothell, Michael D. Byrne, Scott Douglass, Christian Lebiere, and Yulin Qin. 2004. An Integrated Theory of the Mind. *Psychological Review* 111, 4 (2004), 1036–1060.
- [2] Robert A. Bjork, Elizabeth L. Bjork, et al. 1992. A New Theory of Disuse and an Old Theory of Stimulus Fluctuation. *From learning processes to cognitive processes: Essays in honor of William K. Estes* 2 (1992), 35–67.
- [3] Shana K. Carpenter, Harold Pashler, and Nicholas J. Cepeda. 2009. Using Tests to Enhance 8th Grade Students' Retention of U.S. History Facts. *Applied Cognitive Psychology* 23, 6 (2009), 760–771.
- [4] Nicholas J. Cepeda, Harold Pashler, Edward Vul, John T. Wixted, and Doug Rohrer. 2006. Distributed Practice in Verbal Recall Tasks: A Review and Quantitative Synthesis. *Psychological Bulletin* 132, 3 (2006), 354–380.
- [5] Nicholas J. Cepeda, Edward Vul, Doug Rohrer, John T. Wixted, and Harold Pashler. 2008. Spacing Effects in Learning: A Temporal Ridgeline of Optimal Retention. *Psychological Science* 19, 11 (2008), 1095–1102.
- [6] Hermann Ebbinghaus. 1913. *Memory: A Contribution to Experimental Psychology*. Teachers College Press, New York.
- [7] Anette Hunziker, Yuxin Chen, et al. 2019. Teaching Multiple Concepts to a Forgetful Learner. In *Advances in Neural Information Processing Systems*. 4050–4060.
- [8] Sebastian Leitner. 1974. *So lernt man leben*. Droemer-Knaur, München, Zürich.
- [9] Jaclyn K. Maass, Philip I. Pavlik, and Henry Hua. 2015. How Spacing and Variable Retrieval Practice Affect the Learning of Statistics Concepts. In *Artificial Intelligence in Education*. Springer, 247–256.
- [10] Arthur W. Melton. 1970. The Situation with Respect to the Spacing of Repetitions and Memory. *Journal of Verbal Learning and Verbal Behavior* 9, 5 (1970), 596–606.
- [11] Harold Pashler, Nicholas Cepeda, Robert V Lindsey, Ed Vul, and Michael C Mozer. 2009. Predicting the Optimal Spacing of Study: A Multiscale Context Model of Memory. In *Advances in Neural Information Processing Systems*. 1321–1329.
- [12] Siddharth Reddy, Igor Labutov, Siddhartha Banerjee, and Thorsten Joachims. 2016. Unbounded Human Learning: Optimal Scheduling for Spaced Repetition. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1815–1824.
- [13] Siddharth Reddy, Sergey Levine, and Anca Dragan. 2017. Accelerating Human Learning with Deep Reinforcement Learning. *University of California, Berkeley* (2017), 9.
- [14] Burr Settles and Brendan Meeder. 2016. A Trainable Spaced Repetition Model for Language Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. ACL, 1848–1858.
- [15] Sugandh Sinha. 2019. *Using Deep Reinforcement Learning for Personalizing Review Sessions on E-Learning Platforms with Spaced Repetition*. Ph.D. Dissertation. KTH Royal Institute of Technology.
- [16] Paul Smolen, Yili Zhang, and John H. Byrne. 2016. The Right Time to Learn: Mechanisms and Optimization of Spaced Learning. *Nature reviews. Neuroscience* 17, 2 (2016), 77–88.
- [17] Behzad Tabibian, Utkarsh Upadhyay, Abir De, Ali Zarezade, Bernhard Schölkopf, and Manuel Gomez-Rodriguez. 2019. Enhancing Human Learning via Spaced Repetition Optimization. *Proceedings of the National Academy of Sciences* 116, 10 (2019), 3988–3993.
- [18] Shaw TJ, Pernar LIM, et al. 2012. Impact of Online Education on Intern Behaviour around Joint Commission National Patient Safety Goals: A Randomised Trial. *BMJ quality & safety* 21, 10 (2012), 819–825.
- [19] Utkarsh Upadhyay, Abir De, and Manuel Gomez-Rodriguez. 2018. Deep Reinforcement Learning of Marked Temporal Point Processes. In *Advances in Neural Information Processing Systems*. 3172–3182.
- [20] Piotr A. Wozniak. 1990. Optimization of Learning. <http://super-memory.com/english/ol.htm>.
- [21] Zhengyu Yang, Jian Shen, Yunfei Liu, Yang Yang, Weinan Zhang, and Yong Yu. 2020. 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*. ACM, 1917–1920.
- [22] Ahmed Zaidi, Andrew Caines, Russell Moore, Paula Buttery, and Andrew Rice. 2020. Adaptive Forgetting Curves for Spaced Repetition Language Learning. In *Artificial Intelligence in Education*. Springer, 358–363.

A DEPLOYMENT DESIGN

In this Appendix, we describe how the SSP-MMC algorithm was deployed to MaiMemo for optimizing learners' spaced repetition schedules.

A.1 Architecture

Figure 11 shows the framework of the spaced repetition scheduler, which is divided into two main parts: local in green and remote in red.

Each time a user reviews a word, the memory behavior event with time-series information (see Equation 3.1 and Table 1) will be recorded in local, which we refer to as the User Review Logs. The logs are uploaded to the remote after the learner has completed all the reviews for the day.

The Full Review Logs must handle a large volume of writes, needs to be retained permanently, and will not be updated once written, so we use a streaming data service to write the log to the data warehouse. In Data Pre-process ETL, we periodically calculate the difficulty of all words and aggregate the training data needed for

the DHP Memory Model which supplies the training environment for the SSP-MMC Scheduling Algorithm.

After the model parameters and optimal policy are computed, the remote push the relevant configs to local. Then the local predictor and scheduler will load the new configs to predict the memory state and schedule the review date for each word the user has learned.

A.2 Cold Start

In the cold-start phase, It can use a simple scheduling algorithm, such as SM-2. Then the User Review Logs will be collected to the Full Review Logs in the remote and pre-processed to compute difficulty and half-life. These data are used to fit the DHP model to obtain model parameters and as an environment for the SSP-MMC to derive optimal policy. The model parameters and optimal policy in the remote will be sent to the local predictor and scheduler to replace the original simple scheduling algorithm and obtain more User Review Logs, which iteratively improve the DHP Memory Model and the optimal policy generated by the SSP-MMC Scheduling Algorithm.

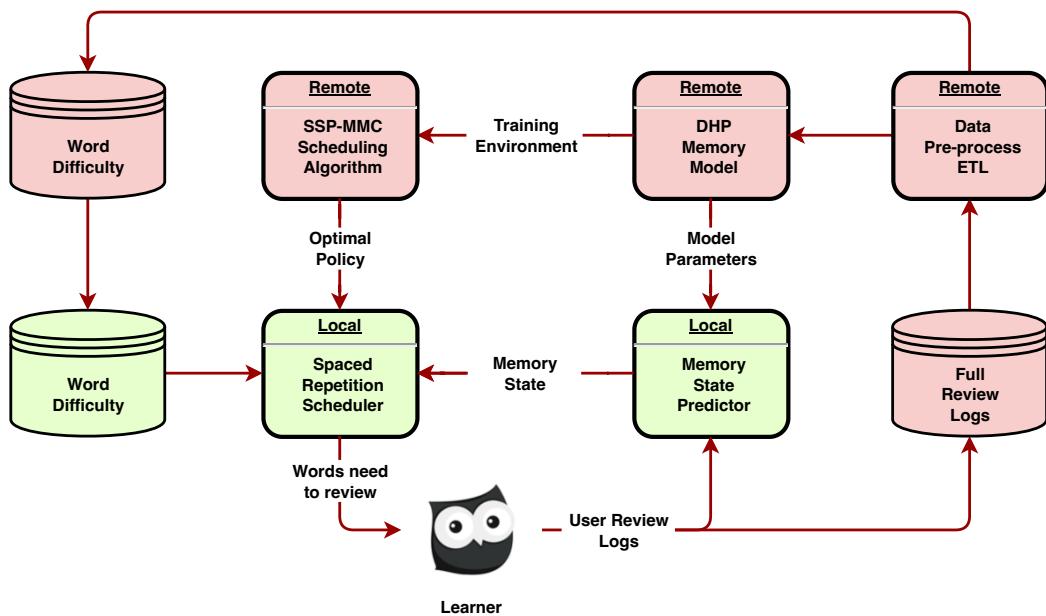


Figure 11: Framework diagram of the spaced repetition system at MaiMemo