Enhancing Human Learning using Stochastic Optimal Control and RL

HUMAN-CENTERED MACHINE LEARNING

http://courses.mpi-sws.org/hcml-ws18/
What is learning?

- Declarative versus Procedural learning

Learning a new language’s vocabulary

Learning how to ride a bike
How do humans *learn*?

- Declarative versus Procedural learning
- Repetition is important!
How do humans learn?

- Declarative versus Procedural learning
- Spaced Repetition is important!

Leitner System for Flash cards

[Leitner 1974]

[Lebbinghaus 1885]
How do humans *learn* in today’s world?

- Computer assisted learning:
  - Anki
  - Duolingo
  - Synap

The platforms decide *when to schedule* reviews based on the user’s history and item.
Strategy to optimize spaced repetition

Data representation

Student feedback and memory dynamics

Optimizing recall

Experiments

Temporal point processes

System of stochastic equations with jumps

Optimal control of jumps

Duolingo
Optimizing spacing between repetition

Agent

Online learning platform

Environment

Learner

Review & successful recall

Review & unsuccessful recall

Marks

When to review to maximize recall probability?

\[ \lambda_i(t) \rightarrow N_i(t) \]

Design (optimal) reviewing intensities
Memory Model: Intuition

- Memory strength decays with time
- Resets to *maximum* immediately after review
- Recall is probabilistic
Memory Model: A Mathematical Model

\[ m(t) = e^{-n(t) \times \eta} \]

\[ n(t) = \begin{cases} 
(1 - \alpha) \times n(t^-) & \text{if recalled} \\
(1 + \beta) \times n(t^-) & \text{if forgotten} 
\end{cases} \]

- \( m(t) \): Probability of recall
- \( n(t) \): Memory decay rate.
- \( \eta \): Time since last review.
Memory Model: SDE with jumps

\[ m(t) = e^{-n(t) \times \eta} \]

- \( m(t) \): Probability of recall
- \( n(t) \): Memory decay rate.
- \( \eta \): Time since last review.

\[ n(t) = \begin{cases} 
(1 - \alpha) \times n(t^-) & \text{if recalled} \\
(1 + \beta) \times n(t^-) & \text{if forgotten} 
\end{cases} \]

\[ dm(t) = -m(t)n(t)dt + (1 - m(t))dN(t) \]
\[ dn(t) = [-\alpha r(t)n(t) + \beta (1 - r(t))n(t)]dN(t) \]
Memory Model: Inferring parameters

\[ m(t) = e^{-n(t) \times \eta} \]

- \( m(t) \): Probability of recall
- \( n(t) \): Memory decay rate.
- \( \eta \): Time since last review.

\[ n(t) = \begin{cases} 
(1 - \alpha) \times n(t^-) & \text{if recalled} \\
(1 + \beta) \times n(t^-) & \text{if forgotten} 
\end{cases} \]

\[ dm(t) = -m(t)n(t)dt + (1 - m(t))dN(t) \]
\[ dn(t) = [-\alpha r(t)n(t) + \beta (1 - r(t))n(t)]dN(t) \]

We can estimate parameters \( \alpha \) and \( \beta \) from data. [Settles et al. 2016]
Optimizing spaced repetition: The Scheduler

Agent

Online learning platform

Environment

When to review to maximize recall probability?

\[ \lambda_i(t) \rightarrow N_i(t) \]

Design (optimal) reviewing intensities

Marks

Review & successful recall

Review & unsuccessful recall
Representing actions of the teacher as MTPPs

Agent

Environment

Online learning platform

- We will control the rate of reviewing \( u(t) \)

- For simplicity, we will consider the problem for just one item.
Spaced repetition: Uniform baseline

Agent

Environment

Does not exploit spacing effect at all.

Online learning platform

$u(t) = \mu$
Spaced repetition: Threshold Heuristic

Agent

Online learning platform

“desirable difficulty”

[Bjork, 1994]

\[ m_{th} = m(s) \]

Environment

Learner

\[ u(t) \]

\[ N(t) \]

1.0

0.0

\[ m_{th} \]

S

15
Spaced repetition: Threshold Heuristic

Agent

Online learning platform

“desirable difficulty”
[Bjork, 1994]

\[ m_{th} = m(s) \]
Optimizing spacing between repetition

Agent

Online learning platform

Environment

When to review to maximize recall probability?

\[ \lambda_i(t) \rightarrow N_i(t) \]

Design (optimal) reviewing intensities

Marks

Review & successful recall

Review & unsuccessful recall
Optimization Objective

Objective trades off high recall and high reviewing rate.

\[
\begin{aligned}
\min_{u(t_0,t_f)} & \quad \mathbb{E}_{(N,r)(t_0,t_f)} \left[ \int_{t_0}^{t_f} \left( \frac{1}{2} (1 - m(\tau))^2 + \frac{1}{2} qu^2(\tau) \right) d\tau \right] \\
\text{subject to} & \quad u(t) \geq 0 \ \forall t \in (t_0, t_f)
\end{aligned}
\]

Dynamics defined by Jump SDEs

\[
\begin{align*}
    dm(t) &= -m(t)n(t)dt + (1 - m(t))dN(t) \\
    dn(t) &= [-\alpha r(t)n(t) + \beta (1 - r(t))n(t)]dN(t)
\end{align*}
\]
Stochastic Optimal Control: Cost-to-go

\[
\text{minimize} \quad \mathbb{E}_{(N,r)(t_0,t_f)} \left[ \int_{t_0}^{t_f} \left( \frac{1}{2} (1 - m(\tau))^2 + \frac{1}{2} qu^2(\tau) \right) d\tau \right] \\
\text{subject to} \quad u(t) \geq 0 \quad \forall t \in (t_0, t_f)
\]

Dynamics defined by Jump SDEs

\[
\begin{align*}
\frac{dm(t)}{dt} &= -m(t)n(t)dt + (1 - m(t))dN(t) \\
\frac{dn(t)}{dt} &= [-\alpha r(t)n(t) + \beta (1 - r(t))n(t)]dN(t)
\end{align*}
\]

\[
J(n(t), m(t), t) = \min_{u(t,t_f)} \mathbb{E}_{(N(s),r(s))|_{s=t}} \left[ \phi(m(t_f), n(t_f)) + \int_{t}^{t_f} \ell(m(\tau), u(\tau)) d\tau \right].
\]

\[
\ell(m(t), n(t), u(t)) = \frac{1}{2} (1 - m(t))^2 + \frac{1}{2} qu^2(t),
\]
Lemma. The optimal cost-to-go satisfies Bellman’s Principle of Optimality

\[ J(n(t), m(t), t) = \min_{u(t,t+dt)} \mathbb{E}[J(n(t + dt), m(t + dt), t + dt)] + \ell(n(t), m(t), u(t)) \]
Stochastic Optimal Control: Solution

\[
J(m(t), n(t), t) = \min_{u(t, t+dt)} \mathbb{E}[J(m(t+dt), n(t+dt), t+dt)] \\
+ \ell(m(t), n(t), u(t))dt
\]

\[
0 = \min_{u(t, t+dt)} \mathbb{E}[dJ(m(t), n(t), t)] + \ell(m(t), n(t), u(t))dt.
\]

\[
dJ(m, n, t) = J_t(m, n, t) - nmJ_m(m, n, t) + [J(1, (1 - \alpha)n, t)r(t) + J(1, (1 + \beta)n, t)(1 - r) \\
- J(m, n, t)]dN(t).
\]

\[
u^*_d(t) = q^{-1} [J_d(m(t), n(t), t) - J_d(1, (1 - \alpha)n(t), t)m(t) - J_d(1, (1 + \beta)n(t), t)(1 - m(t))]_+
\]

Optimal solution (MEMORIZE): \[u(t) = q^{-\frac{1}{2}} (1 - m(t))\]
Evaluating Memorize: Dataset

- **Natural experiment** on Duolingo:
  - 12 million sessions
  - 5.3 million unique (user, word) pairs

[Settles & Meeder, 2016]

http://learning.mpi-sws.org/memorize/
Evaluating Memorize: Metric

- **Natural experiment** on Duolingo:
  - 12 million sessions
  - 5.3 million unique (user, word) pairs

- Find (user, item) pairs closest to each scheduler using top-quantile by likelihood.
Evaluating Memorize: Metric

- **Natural experiment** on Duolingo:
  - 12 million sessions
  - 5.3 million unique (user, word) pairs

- Find (user, item) pairs closest to each scheduler using top-quantile by likelihood.

- **Relative empirical forgetting rate** as metric:
  - Treat first \( n - 1 \) sessions as “study”
  - Treat last attempt as the “test”, calculate forgetting rate

\[
\hat{n} = - \log(\hat{m}(t_n))/(t_n - t_{n-1}).
\]
Evaluating Memorize: Results

Control for:

- Number of reviews: $n$
- Duration of study: $T = t_{n-1} - t_1$

[Tabibian et. al. 2019]

http://learning.mpi-sws.org/memorize/
Evaluating Memorize: Results

Control for:

- Number of reviews: \( n \)
- Duration of study: \( T = t_{n-1} - t_1 \)

Lower is better

http://learning.mpi-sws.org/memorize/

[Tabibian et. al. 2019]
Evaluating Memorize: Results

Control for:
- Number of reviews: $n$
- Duration of study: $T = t_{n-1} - t_1$

http://learning.mpi-sws.org/memorize/

[Tabibian et. al. 2019]
Evaluating Memorize: Results

Control for:

- Number of reviews: \( n \)
- Duration of study: \( T = t_{n-1} - t_1 \)

![Graph showing correlation between LL of following Memorize and \( \hat{n} \)]

Correlation between LL of following Memorize and \( \hat{n} \)

Lower is better

\( T = 8 \pm 3.2 \) days

http://learning.mpi-sws.org/memorize/

[Tabibian et. al. 2019]
The Memory Model is actually complicated:

- Massed repetition
- Dependence between items
- Multiscale Context Model
Case for Reinforcement Learning

Agent

Online learning platform

Environment

Learner

Review & successful recall

Review & unsuccessful recall

When to review to maximize recall probability?

Improve continuous retention ✓

Improve test scores ✗
Complex Memory model and rewards

However, one may have access to test scores:

**Key idea:**
Think of the test score as rewards in a reinforcement learning setting!

[Upadhyay et al., 2018]
Teacher actions and Student feedback

\[ p_{A;\theta}^* = (\lambda_{\theta}^*, m_{\theta}^*) \]

\[ p_{F;\phi}^* = (\lambda_{\phi}^*, m_{\phi}^*) \]

We do not know the feedback distribution but we can sample from it...

...and measure test scores (rewards)

[Upadhyay et al., 2018]
What is the goal in reinforcement learning?

We aim to maximize the average reward in a time window $[0, T]$:

$$J(\theta) = \max_{p^*_A; \theta(\cdot)} \mathbb{E}_{A_T \sim p^*_A; \theta(\cdot), F_T \sim p^*_F; \phi(\cdot)} \left[ R^*(T) \right]$$

Actions asynchronous, Feedback synchronous

Connection to optimal control:

$$J(n(t), m(t), t) = \min_{u(t, t_f)} \mathbb{E}_{(N(s), r(s)) \mid s = t_f^{t_f}} \left[ \phi(m(t_f), n(t_f)) + \int_t^{t_f} \ell(m(\tau), u(\tau)) d\tau \right]$$

[Upadhyay et al., 2018]
We use gradient descent to improve the policy, i.e., the intensity, over time:

\[
\theta_{l+1} = \theta_l + \alpha_l \nabla_\theta J(\theta)\bigg|_{\theta=\theta_l}
\]

We need to compute the gradient of an average. But the average depends on the parameters!

\[
\nabla_\theta J(\theta) = \nabla_\theta \mathbb{E}_{A_T \sim p^*_A; \theta(\cdot), F_T \sim p^*_F; \phi(\cdot)} [R^*(T)]
\]
The reinforce trick allows us to overcome this implicit dependence:

\[
\nabla_{\theta} J(\theta) = \nabla_{\theta} \mathbb{E}_{A_T \sim p_A^*; \theta(\cdot), F_T \sim p_F^*; \phi(\cdot)} [R^*(T)]
\]

Appendix A in Upadhyay et al., 2018

\[
\nabla_{\theta} J(\theta) = \mathbb{E}_{A_T \sim p_A^*; \theta(\cdot), F_T \sim p_F^*; \phi(\cdot)} [R^*(T) \nabla_{\theta} \log \mathbb{P}_\theta(A_T)]
\]
Likelihood of action events

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{A_T \sim p^*_A; \theta(\cdot), F_T \sim p^*_F; \phi(\cdot)} \left[ R^*(T) \nabla_{\theta} \log \mathbb{P}_{\theta}(A_T) \right]$$

The key remaining question is how to parametrize the intensity $\lambda^*_\theta(t)$

$$\mathbb{P}(A_T) := \left( \prod_{e_i \in A_T} \lambda^*_\theta(t_i) \right) \exp \left( - \int_0^T \lambda^*_\theta(s) \, ds \right)$$

[Upadhyay et al., 2018]
Policy parametrization

Output layer:
\[ \lambda^*_\theta(t) = \exp(b_\lambda + w_t(t - t') + V_\lambda h_i) \]

Hidden layer:
\[ h_i = \tanh(W_h h_{i-1} + W_1 \tau_i + W_4 b_i + b_h) \]

Input layer:
\[ \tau_i = W_t(t_i - t_{i-1}) + b_t \]
\[ b_i = W_f e_i + b_b \]

Parameters
\[ \Pr[y_{i+1} = c] = \frac{\exp \langle V^y_{c,:}, h_i \rangle}{\sum_{l \in \mathbb{Z}} \exp \langle V^y_{l,:), h_i \rangle} \]
Sampling from the policy

\[ \lambda^*_\theta(t) = \exp(b_\lambda + w_t(t - t') + V_\lambda h_i) \]

The intensity can increase or decrease every time an event by the other broadcasters take place:

- We cannot apply just superposition
- We can use inversion sampling: The CDF is a function by parts, where each part is defined once an event by the other broadcasters happens
Results: Improved test scores

It learns the difficulty and memory model.

Higher is better

(a) Recall

TPPRL  MEMORIZE  Uniform

[Upadhyay et al., 2018]
Results: Intuition

It *learns* the difficulty and memory model

[Upadhyay et al., 2018]