Neural Optimal-Controlled Memory

October 31, 2019

1 Motivations

Similar to other long-term credit assignment problems, memory control is difficult due to:

1. Lack of a principled motivation that drive the control besides error-driven motivation
2. Gradient vanishing across long timesteps
3. Local optima introduced by backpropagation

We aim to overcome these challenges with:

- Introduce maximum contribution principle (1)
- Neural optimal control framework: hopefully solve (2)(3) and support (1)

2 Related ideas

Self-supervision is a classical concept. The current literature is based on generative principles. (1) Helmholtz machine [5] relies on a hypothesis that we need a generation process to support recognition process. In particular, a generator creates fantasy of hidden activations, from which the recognizer is trained after. Alternatively, the generator is trained towards hidden activations driven by the recognizer. The mechanism happens inside the neural network. (2) World model [4] also aims to generate an imagination of the world, from which an agent can be trained on in sequential setting. Both approaches take advantages of the training signals from the fantasy created by the generator, which motivates the system to learn even when task-driven training signal is not available. Both minimize the discrepancy between fantasy and real activities.

We aim for the same goal, yet with different principle. Our principle is based on contribution. We maximize the contribution, from which a fantasy of optimal actions is imagined. The controller is trained towards these fantasy. The learning in our approach alternates between backpropagation and optimal control.

3 Formulate memory control as optimal control problem

Recap universal memory control:

- The memory is controlled by the controller (parameterized as θ)
- At each timesteps, the memory controller generates control signal ξ_t (partially using external (assume stochastic) inputs a_t), update memory state $\mathbf{M}_t$ then repeat for future timesteps
- The memory control system of dynamics for an instance of a:

$$\mathbf{M}_{t+1} = f_{write}\left(\mathbf{M}_t, \xi_t\right) \tag{1}$$
$$\xi_{t+1} = f_\theta\left(f_{read}\left(\mathbf{M}_{t+1}, \xi_t\right), a_t\right) \tag{2}$$

and we normally want to minimize some loss function which is conditioned on the final state of the dynamic marginalized over external inputs:

$$\min_{\theta\in\Theta}\sum_a \Phi_a\left(\mathbf{M}_{T,a}\right) \tag{3}$$

If we want to add motivation loss, the final loss function becomes,

$$\min_{\theta\in\Theta}\sum_a \Phi_a\left(\mathbf{M}_{T,a}\right) + \sum_a\sum_t L_t\left(\xi_{t,a}, \mathbf{M}_{t,a}, a_t\right) \tag{4}$$

where $\Phi_a\left(\mathbf{M}_{T,a}\right)$ is the task (delayed, long-term) loss and $L_t\left(\xi_{t,a}, \mathbf{M}_{t,a}, a_t\right)$ is the cost rate or motivation loss (intrinsic, energy,...) associated with external variable a. The cost rate represents an internal motivation to minimize some expected cost the agent is interested in. It should depends locally only the current state and action.

For the sake of simplicity, the writing process is defined by the write-weight $w_t^w = \mathrm{D}_w\xi_t \in \mathbb{R}^N$, the update value $v_t = \mathrm{D}_v\xi_t \in \mathbb{R}^D$ and erase value $e_t = \mathrm{D}_e\xi_t \in [0,1]^D$ as follows:

$$\mathbf{M}_{t+1} = \mathbf{M}_t \circ \left(E - w_t^w e_t^\top\right) + w_t^w v_t^\top \tag{5}$$

where $\circ$ is element-wise product, E is an $N \times D$ matrix of ones and the constant binary matrices D_w,D_v, and D_e indicate the elements of ξ_t that will be allocated to form w_t^w, v_t and e_t. We can rewrite the first dynamic in full form:

$$\mathbf{M}_{t+1} = f_t\left(\mathbf{M}_t, \xi_t\right) = \mathbf{M}_t \circ \left(E - \mathrm{D}_w\xi_t\left(\mathrm{D}_e\xi_t\right)^\top\right) + \mathrm{D}_w\xi_t\left(\mathrm{D}_v\xi_t\right)^\top \tag{6}$$
$$= \mathbf{M}_t \circ \left(E - \mathrm{D}_w\xi_t\xi_t^\top\mathrm{D}_e^\top\right) + \mathrm{D}_w\xi_t\xi_t^\top\mathrm{D}_v^\top \tag{7}$$

Basically, f_t is convex over ξ_t.

The read vector at timestep t-th can be written as $r_t = \mathbf{M}_{t+1}^{\top} w_t^r = \mathbf{M}_{t+1}^{\top} D_r \xi_t$, where $\mathbf{M}$, w_t^r are the data memory and the read weight, respectively. The constant binary matrix D_r indicates the elements of ξ_t that will be allocated to form w_i^r. Hence, the output of the RNN controller reads

$$\begin{aligned} c_{t+1} &= W_o \sigma \left(W x_{t+1} + U h_t + V r_t \right) && (8) \\ &= W_o \sigma \left(W x_{t+1} + U h_t + V \mathbf{M}_{t+1}^{\top} D_r \xi_t \right) && (9) \end{aligned}$$

This leads to a recursive equation 2 rewritten as,

$$\begin{aligned} \xi_{t+1} &= W^c \xi_t && (10) \\ &= W^c W_o \sigma \left(W x_{t+1} + U h_t + V \mathbf{M}_{t+1}^{\top} D_r \xi_t \right) && (11) \\ &= W^c W_o \sigma \left(W x_{t+1} + U h_t + V \left(\mathbf{M}_t \circ \left(E - \mathrm{D}_w \xi_t \xi_t^{\top} \mathrm{D}_e^{\top} \right) + \mathrm{D}_w \xi_t \xi_t^{\top} \mathrm{D}_v^{\top} \right)^{\top} D_r \xi_t \right) && (12) \end{aligned}$$

where $\{x_{t+1}, h_t\} = a_t$ and $\{W^c, W_o, W, U, V\} = \theta$ (can vary through time as θ_t).

There are two ways to cast the memory control to an optimal control problem.

(1) Use single state variable $s_t = [\mathbf{M}_t.\mathit{vectorize}(), \xi_t]$, we can rewrite $\mathbf{M}_t = (D_m s_t).\mathit{reshape}()$ and $\xi_t = D_\xi s_t$ then the system of dynamics can be rewritten as single dynamic over s_t:

$$\begin{aligned} s_{t+1} = \Big[& (D_m s_t).\mathit{reshape}() \circ \left(E - \mathrm{D}_w D_\xi s_t (D_\xi s_t)^{\top} \mathrm{D}_e^{\top} \right) + \mathrm{D}_w D_\xi s_t (D_\xi s_t)^{\top} \mathrm{D}_v^{\top}, \\ & W^c W_o \sigma W x_{t+1} + U h_t + V (D_m s_t).\mathit{reshape}() \circ \left(E - \mathrm{D}_w D_\xi s_t (D_\xi s_t)^{\top} \mathrm{D}_e^{\top} \right) \\ & + \mathrm{D}_w D_\xi s_t (D_\xi s_t)^{\top} \mathrm{D}_v^{\top} D_r D_\xi s_t \Big] \\ = {} & f(s_t, \theta_t) \end{aligned}$$

Then the control signal is θ_t to be optimized and the dynamic can be controlled by optimal control theory [8]. However, it poses several challenges such as θ_t is very high dimensional and $f(s_t, \theta_t)$ is non-convex, hard to find global optimum and θ should vary through time as θ_t (not recurrent anymore, can use NSM or other fast-weight to generate θ_t).

(2) Only optimally control the first dynamic Eq. (1) and use neural network to learn θ that satisfies the second dynamic Eq. (2). It seems that no one has tried this approach in deep learning before. Thus, we will propose to use this as a new framework to learn memory optimal control.

4 Memory Optimal Control Framework

We wish to solve the following optimal control problem

$$\min_{\xi_{t,a}\in\Xi} J\left(\xi_{t,a}\right)=\sum_{a}\Phi_{a}\left(\mathbf{M}_{T,a}\right)+\sum_{a}\sum_{t}L_{t}\left(\xi_{t,a},\mathbf{M}_{t,a},a_{t}\right)$$

subject to:

$$\mathbf{M}_{t+1,a}=f_{t}\left(\mathbf{M}_{t,a},\xi_{t,a}\right),t=\overline{1,T},a\in\mathcal{A} \tag{13}$$

For each instance of a, we can solve this optimal control problem using Pontryagin's Maximum Principle.

Theorem 1. *Assume f_t and Φ be sufficiently smooth in $\mathbf{M}$. Assume further that for each t, the set $\left\{f_t\left(\mathbf{M}_t,\xi\right):\xi\in\Xi_t\right\}$ and $\left\{L_t\left(\mathbf{M}_t,\xi,a_t\right):\xi\in\Xi_t\right\}$ are convex. Then, there exists co-state processes $\boldsymbol{p^*}=\left\{p_t^*:t=\overline{1,T}\right\}$ and optimal solution $\boldsymbol{\xi^*}=\left\{\xi_t^*:t=\overline{1,T}\right\}$ such that the following holds for $t=\overline{1,T}$:*

$$\mathbf{M}_{t+1}^{*}=\nabla_{p}H_{t}\left(\mathbf{M}_{t}^{*},p_{t+1}^{*},\xi_{t}^{*}\right),\mathbf{M}_{0}^{*}=\mathbf{M}_{0} \tag{14}$$

$$p_{t+1}^{*}=\nabla_{\mathbf{M}}H_{t}\left(\mathbf{M}_{t}^{*},p_{t+1}^{*},\xi_{t}^{*}\right),p_{T}^{*}=-\nabla\Phi\left(\mathbf{M}_{T}\right) \tag{15}$$

$$H_{t}\left(\mathbf{M}_{t}^{*},p_{t+1}^{*},\xi_{t}^{*}\right)\geq H_{t}\left(\mathbf{M}_{t}^{*},p_{t+1}^{*},\xi_{t}\right):\forall\xi\in\Xi_{t} \tag{16}$$

where H is the Hamiltonian function: $H_t:\mathbb{R}^{N\times D}\times\mathbb{R}^{N\times D}\times\Xi_t\rightarrow\mathbb{R}$ by

$$H_{t}\left(\mathbf{M},p,\xi\right)=p\cdot f_{t}\left(\mathbf{M},\xi\right)-L_{t}\left(\xi,\mathbf{M},a_{t}\right) \tag{17}$$

We can use method of successive approximations (MSA) to implement Pontryagin's Maximum Principle. Let assume given θ, a, we can find a possible optimal set of control signal $\boldsymbol{\xi^*}=\left\{\xi_t^*:t=\overline{1,T}\right\}$ thanks to MSA, we need to learn θ to be able to generate $\boldsymbol{\xi^*}$while satisfying Eq. (2). It maybe possible if a true optimal set of control signal exists as θ is basically a Turing Machine (it can predict any sequence given strong supervision). The importance is θ is given with ground truth output $\boldsymbol{\xi^*}$ every timestep (no more delayed error signal), and thus strongly supervised. We combine the MSA and Turing Machine training into one algorithm (mini-batch version) 1.

The idea of the algorithm 1 is to alternatively find the control signal $\boldsymbol{\xi^*}$ to minimize the loss using global optimizer tool (Pontryagin's Maximum Principle is globally optimal given convexity) and find the parameter θ to satisfy the dynamics given $\boldsymbol{\xi^*}$and external data. In other words, we want the Turing Machine learn the optimal control solution. This may introduce some challenges:

- The computation cost of $arg\max$ operator. The cheapest way is to use gradient ascent: $\xi_{t,a}^{k+1}=\xi_{t,a}^{k}+\beta\frac{\partial H_t\left(\mathbf{M}_{t,a}^{k},p_{t+1,a}^{k},\xi_{t,a}^{k}\right)}{\partial\xi_{t,a}^{k}}$

Algorithm 1 Memory Optimal Control Framework

```
 1: Sample B sequences of training data $\left\{x_{t,a} : t = \overline{0,T}\right\}$
 2: for $a = 1, B$ do
 3:     Initialize $\boldsymbol{\xi}_{\boldsymbol{a}}^{\mathbf{0}} = \left\{\xi_{t,a}^0 \in \Xi_t : t = \overline{0, T-1}\right\}$
 4:     Initialize $\mathbf{M}_{0,a}^0, \in \mathcal{M}$
 5:     for $k = 1, K$ do
 6:         for $t = 0, T-1$ do
 7:             $\mathbf{M}_{t+1,a}^k = f_t\left(\mathbf{M}_{t,a}^k, \xi_{t,a}^k\right)$
 8:         end for
 9:         $p_{T,a}^k = -\nabla\Phi\left(\mathbf{M}_{T,a}^k, x_{T,a}\right)$
10:         for $t = T-1, 0$ do
11:             $p_{t,a}^k = \nabla_{\mathbf{M}} H_t\left(\mathbf{M}_{t,a}^k, p_{t+1,a}^k, \xi_{t,a}^k\right)$
12:         end for
13:         for $t = 0, T-1$ do
14:             $\xi_{t,a}^{k+1} = arg\max_{\xi\in\Xi_t} H_t\left(\mathbf{M}_{t,a}^k, p_{t+1,a}^k, \xi_{t,a}^k\right)$
15:         end for
16:     end for
17:     for $t = 0, T-1$ do
18:         $\hat{\xi}_{t+1,a}^K = f_\theta\left(f_{read}\left(\mathbf{M}_{t,a}^K, \xi_{t,a}^K\right), x_{t,a}\right)$
19:     end for
20:     $\mathcal{L}_\xi = \sum_{t=1}^{T-1}\left\|\hat{\xi}_{t,a}^K - \xi_{t,a}^K\right\|$
21:     $\theta = \theta - \alpha\frac{\partial\mathcal{L}_\xi}{\partial\theta}$
22: end for
```

- Initializing a set of control signals (hundreds of timesteps) then optimizing seems very hard. We can ease it with a mix strategy:
  - Optimize using optimal control for only a random subset $\boldsymbol{\xi}^{\blacklozenge} \subset \boldsymbol{\xi}^{*}$, the other control signals are generated using Turing Machine
  - Let's Turing Machine optimize the loss function too
  - If $\boldsymbol{\xi}^{\blacklozenge} = \emptyset$, it becomes normal training using backpropagation

5 Maximum Contribution Principle

Some motivations:

- It seems to have hidden relation to Free Energy Principle [2] and Least-action Principle.
- The idea is simple: a rationale action is the one that on average contribute to the final action (the one that associates with the error signal)
- Examples: we avoid to act randomly (contribution mean =0) or act no thing (contribution = 0). Every action counts. It must direct toward the final action. By doing so, the "energy" seems minimized.

- Contribution analysis is used recently:
  - Contribution is used to measure memory capacity (first order derivative [7], second order derivative Fisher Information [3])
  - Contribution is used to explain deep learning (how some visual feature affects output)
  - Contribution is used in reward redistribution (TVT [6] uses memory read weight as contribution, Rudder [1] uses Integrated Gradients as contribution)

In memory control, the inner motivation is assumed to be Maximum Contribution Principle (MCP). The cost rate is defined as:

$$L_t\left(\xi_{t,a}, \mathbf{M}_{t,a}, a_t\right) = -\mathcal{C}\left(\xi_{t,a}, \xi_{T,a}\right) \tag{18}$$

where $\mathcal{C}\left(\xi_{t,a}, \xi_{T,a}\right)$ measures the contribution of $\xi_{t,a}$ towards $\xi_{T,a}$. Some obvious bad controls that violate MCP:

- Do not use memory: $\xi_{t,a} = 0$, $\mathcal{C}\left(\xi_{t,a}, \xi_{T,a}\right) = 0$
- Overwritten: $\xi_{t,a}$ is overwritten by some $\xi_{t',a}$ with $t' > t$, $\mathcal{C}\left(\xi_{t,a}, \xi_{T,a}\right) = 0$. When the memory is finite, overwriting is unavoidable. We need to choose the least contribution to overwrite, this is aromatic when minimizing $\sum\limits_a \sum\limits_t -\mathcal{C}\left(\xi_{t,a}, \xi_{T,a}\right)$. Current overwriting such as least-used is heuristic not optimal. Actually, overwriting to least-used memory slot loose rare events.

However, $\mathcal{C}\left(\xi_{t,a}, \xi_{T,a}\right)$ is highly non-convex, hard to compute. We derive a lower bound (easier to compute) for a class of contribution $\mathcal{C}\left(\xi_t, \xi_T\right) = \left\| \frac{\partial^n \xi_T}{\partial \xi_t^n} \right\|$ for $n = 1, 2$. It is easy to see that for $n = 1$,

$$\begin{aligned}
\mathcal{C}\left(\xi_t, \xi_T\right) &= \left\| \frac{\partial \xi_T}{\partial \xi_t} \right\| && (19)\\
&= \left\| \frac{\partial \xi_T}{\partial \xi_{t+k}} \frac{\partial \xi_{t+k}}{\partial \xi_t} \right\| && (20)\\
&\geq \sigma_{min}\left(\frac{\partial \xi_T}{\partial \xi_{t+k}} \right) \left\| \frac{\partial \xi_{t+k}}{\partial \xi_t} \right\| && (21)
\end{aligned}$$

where $k > 0$. In optimal control problem Eq. (13), $\sigma_{min}\left(\frac{\partial \xi_T}{\partial \xi_{t+k}}\right)$ is independent from ξ_t. The contribution to the last action can be maximized by maximizing the contribution to some future action. The simplest bound is:

$$L_t\left(\xi_{t,a}, \mathbf{M}_{t,a}, a_t\right) = -\left\| \frac{\partial \xi_{t+1,a}}{\partial \xi_{t,a}} \right\| \tag{22}$$

This bound is simple to compute and likely to be convex, which guarantees the global optimum of using Pontryagin's Maximum Principle.

For other classes of contribution, if we want to avoid computing $\mathcal{C}\left(\xi_{t,a},\xi_{T,a}\right)$, we may want to learn a neural approximation for this term. Given $\xi_{t,a}$, we can train a (convex) neural network $\hat{\mathcal{C}}_{\phi}$ to predict $\mathcal{C}\left(\xi_{t,a},\xi_{T,a}\right)$. As we can collect the true $\mathcal{C}\left(\xi_{t,a},\xi_{T,a}\right)$ when we observe $\xi_{T,a}$ and $\xi_{T,a}$ is actually a function of $\xi_{t,a}$, the learning is doable. The the cost rate becomes:

$$L_t\left(\xi_{t,a},\mathbf{M}_{t,a},a_t\right)=-\hat{\mathcal{C}}_{\phi}\left(\xi_{t,a}\right) \tag{23}$$

References

[1] Jose A Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. Rudder: Return decomposition for delayed rewards. *arXiv preprint arXiv:1806.07857*, 2018.

[2] Karl Friston. The free-energy principle: a rough guide to the brain? *Trends in cognitive sciences*, 13(7):293–301, 2009.

[3] Surya Ganguli, Dongsung Huh, and Haim Sompolinsky. Memory traces in dynamical systems. *Proceedings of the National Academy of Sciences*, 105(48):18970–18975, 2008.

[4] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[5] Geoffrey E Hinton, Peter Dayan, Brendan J Frey, and Radford M Neal. The" wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, 1995.

[6] Chia-Chun Hung, Timothy Lillicrap, Josh Abramson, Yan Wu, Mehdi Mirza, Federico Carnevale, Arun Ahuja, and Greg Wayne. Optimizing agent behavior over long time scales by transporting value. *arXiv preprint arXiv:1810.06721*, 2018.

[7] Hung Le, Truyen Tran, and Svetha Venkatesh. Learning to remember more with less memorization. In *International Conference on Learning Representations*, 2019.

[8] Qianxiao Li and Shuji Hao. An optimal control approach to deep learning and applications to discrete-weight neural networks. *arXiv preprint arXiv:1803.01299*, 2018.