Online learning for spiking neural networks with relative information rate

Shaowei Lin
Topos Institute

(joint work with Tenzin Chan, Chris Hillar and Sarah Marzen)

20231211

IMSI Workshop on
Bayesian Statistics and Statistical Learning

Part I

SPIKING NEURAL NETWORKS

Spiking Neural Networks

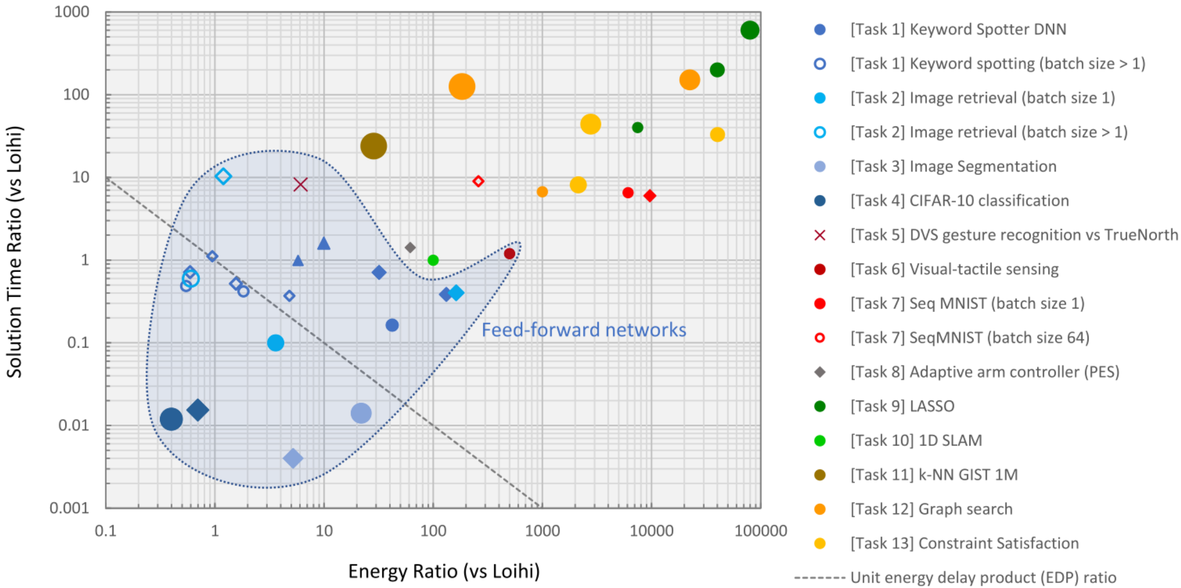
- Spiking neural networks (SNNs) are artificial neural networks (ANNs) that mimic biological neural networks (BNNs) more closely than feedforward neural networks (FNNs).

- **Event-driven.** Neurons communicate only when there is a spike.

- **Energy-efficient.**
  - Human brain $\sim$20W
  - Training GPT4[1] $\sim$55GWh $> 6 \times 10^6$ human years.
  - Training cost? Inference cost?

[1] https://towardsdatascience.com/the-carbon-footprint-of-gpt-4-d6c676eb21ae

Energy Consumption



[2] Davies M, Wild A, Orchard G, Sandamirskaya Y, Guerra GA, Joshi P, Plank P, Risbud SR. Advancing neuromorphic computing with loihi: A survey of results and outlook. Proceedings of the IEEE. 2021 Apr 6;109(5):911-34.

Spike Response Model

- For clarity, we study a simplified (stochastic) *spike response model* (SRM)[3] for a directed network (V, E) where V is the set of neurons and E is the set of synapses.

- Our SRM is a continuous-time Markov chain where at time t, each neuron j has a membrane potential u_{jt} and an instantaneous spiking rate

$$\rho_{jt} = \rho_0 \exp(\beta u_{jt})$$

for some fixed rate constant ρ_0 and fixed inverse temperature β.

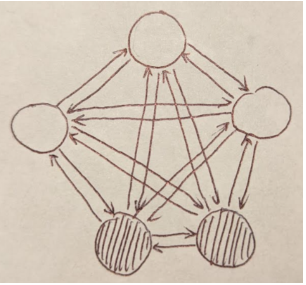
- The membrane potential u_{jt} of neuron j at time t is given by

$$u_{jt} = u_0 + \sum_{ij \in E} w_{ij} c_{ijt}$$

where u_0 is a fixed reset potential, w_{ij} is the synaptic weight from neuron i to neuron j, and c_{ijt} counts the spikes from neuron i since the last spike from neuron j (up to some max L).

[3] https://neuronaldynamics.epfl.ch/online/Ch9.S1.html

Transition Rates



- The SRM is a continuous-time Markov chain with states $c_t = (c_{ijt})_{ij \in E} \in \{0, \ldots, L\}^E$ and weights $w = (w_{ij})_{ij \in E} \in \mathbb{R}^E$. Let Γ be the transition rate matrix.

- The entry $\Gamma_{cc'}$ is nonzero only when c' is derived from c by the spiking of some neuron $j \in V$, i.e. c_{ji} increases by 1 (up to L) for all $ji \in E$ but c_{ij} resets to 0 for all $ij \in E$. Here,

$$\Gamma_{cc'} = \rho_0 \exp(\beta u_j), \quad \text{where } u_j = u_0 + \textstyle\sum_{ij \in E} w_{ij} c_{ij},$$

$$\Gamma_{cc} = -\textstyle\sum_{c' \neq c} \Gamma_{cc'}.$$

Path Distribution

- We sample *paths* from the SRM by *uniformization*. Given $\gamma > |\Gamma_{cc}|$ for all c, sample from a Poisson process with rate γ where at each firing, state c jumps to state c' with probability

$$P_{cc'} = \begin{cases} \Gamma_{cc'}/\gamma, & \text{if } c' \neq c, \\ 1 - \Gamma_{cc}/\gamma, & \text{otherwise.} \end{cases}$$

- A path $x_{0\ldots T} : [0, T] \to \{0, \ldots, L\}^E$ of time-length T with jumps $c_0, c_1, \ldots, c_n$ has probability

$$p(x_{0\ldots T}) = \pi(c_0) \frac{(\gamma T)^n}{n!} e^{-\gamma T} \prod_{i=0}^{n-1} P_{c_i c_{i+1}}$$

where $\pi(\cdot)$ is the initial distribution on the states.

Discrete Time Approximation

- Allow multiple spikes in one time step. Good for simulating on GPUs.
- Discretize time into small intervals of size δ.
- Fix a neuron j. Let c be the current state of the network. Let ρ_j be the resulting spiking rate.

$$p(\,\text{neuron } j \text{ unchanged} \,|\, c) = e^{-\delta\rho_j}$$
$$p(\,\text{neuron } j \text{ spikes} \,|\, c) = 1 - e^{-\delta\rho_j} \approx \delta\rho_j\, e^{-\delta\rho_j}$$

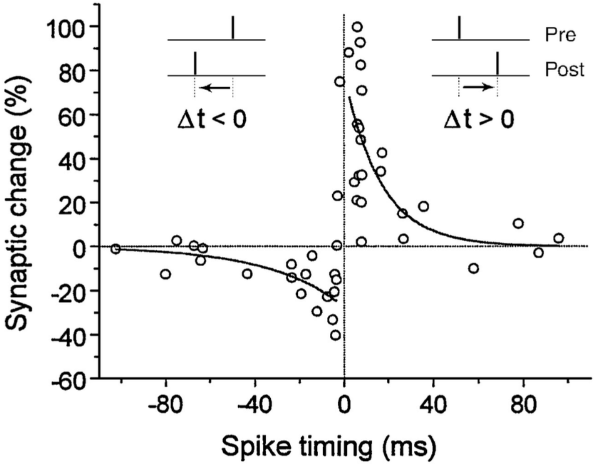
- Let $V_s \subset V$ be the set of neurons that spiked. Let c' be the resulting state of the network.

$$p(c'|c) = \prod_{j\in V} e^{-\delta\rho_j} \prod_{j\in V_s} \delta\rho_j$$

- The limit as $\delta \to 0$ of the above process is our continuous-time Markov chain.

Spike-Timing-Dependent Plasticity

- How do we train the weights w_{ij}? An experimentally-observed method is STDP (Bi & Poo 1998). Update depends on temporal order of and interval between pre-spike and post-spike.
- How do we train the weights in a model with *hidden variables*?



[4] Asl, Mojtaba Madadi. "Propagation delays determine the effects of synaptic plasticity on the structure and dynamics of neuronal networks." (2018).

Part II

Relative Information

Hidden Variables

Setup

- Observed variable X
- Hidden variable Z
- True distribution $q(X)$
- Model distribution $p_\theta(X, Z)$ parametrized by θ
- Marginal distribution $p_\theta(X) = \int p_\theta(X, Z)dZ$

Goal

- Find θ minimizing the *(relative) information* or *Kullback-Leibler divergence* of X from p_θ to q.

$$I_{q\|p_\theta}(X) = \int q(X) \log \frac{q(X)}{p_\theta(X)} dX$$

Variational Inference

Trick.

- Introduce distribution $q(Z|X)$ as extra parameter for optimization
- *Discriminative* distribution $q(X,Z) = q(Z|X)q(X)$
- *Generative* distribution $p_\theta(X,Z) = p_\theta(X|Z)p_\theta(Z)$
- Minimize
$$I_{q||p_\theta}(X,Z) = \int q(X,Z) \log \frac{q(X,Z)}{p_\theta(X,Z)} dXdZ$$
by alternatingly varying $q(Z|X)$ while holding $p_\theta(X,Z)$ fixed, and vice versa.

Variants.

- *EM algorithm* (Dempster-Laird-Rubin)[5]. Let $q(Z|X)$ be $p_\theta(Z|X)$ at each step of the optimization.
- *em algorithm* (Amari)[6]. Let $q(Z|X)$ be parametrized $q_\lambda(Z|X)$ and alternatingly optimize θ and λ.
- Amari's *em* algorithm is biologically more plausible because **Bayesian inversion is hard**!

[5] Dempster, A.P., N.M. Laird, and D.B. Rubin. "Maximum likelihood from incomplete data via the EM algorithm." JRSS 39, no. 1 (1977): 1-22.

[6] Amari, Shun-ichi. "Information geometry of the EM and em algorithms for neural networks." Neural networks 8, no. 9 (1995): 1379-1408.

Conditional Relative Information

- A powerful concept is the *(conditional relative) information* of Z given X from p_θ to q.

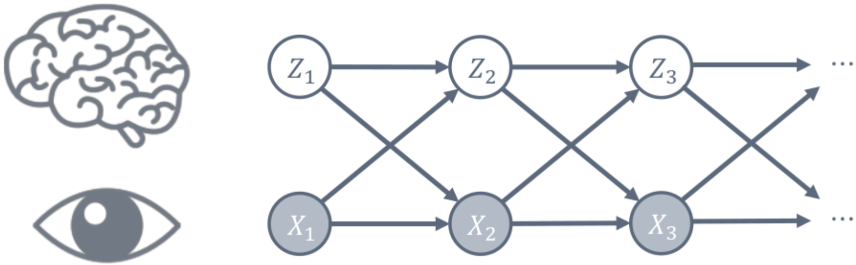
$$I_{q\|p_\theta}(Z|X) = \int q(X) \left(\int q(Z|X) \log \frac{q(Z|X)}{p_\theta(Z|X)}\, dZ \right) dX$$

- It satisfies a fundamental lemma, the *Chain Rule*.

$$I_{q\|p_\theta}(Z,X) = I_{q\|p_\theta}(Z|X) + I_{q\|p_\theta}(X)$$

- In the EM/*em* algorithms,
  - $I_{q\|p_\theta}(Z|X)$ is minimized in the E/*e*-step, and
  - $I_{q\|p_\theta}(Z,X)$ is minimized in the M/*m*-step.

Time Series with Memory

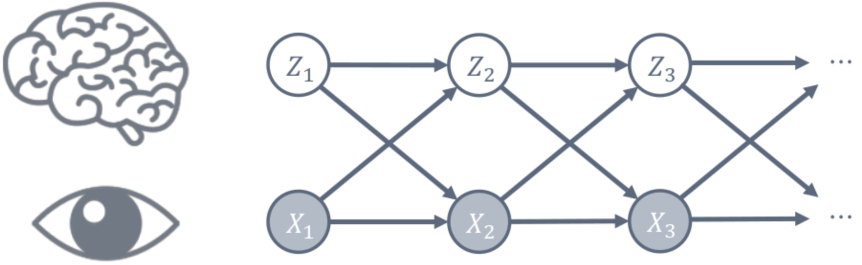


- **Time.** Assume discrete time for simplicity.
- **Environment**. $X_1, X_2, \ldots$. Immutable. Possibly partially hidden.
- **Memory**. $Z_1, Z_2, \ldots$. Mutable. Not latent/hidden variables!

- **Goal**. Optimize use of limited memory for predicting environment.
- **Objective.** Minimize

$$\lim_{T\to\infty} \frac{1}{T} I_{q\|p_\theta}(X_{1\ldots T}, Z_{1\ldots T})$$

Information Constraints



Put different constraints on structure of $q(Z_{1\dots T}|X_{1\dots T}) = \prod_k q(Z_{k+1}|Z_{1\dots k}, X_{1\dots T})$.
Let p^* denote the resulting p_θ that minimizes $I_{q\|p_\theta}(X_{1\dots T}, Z_{1\dots T})$.

- **No constraints.** $q(Z_{1\dots T}|X_{1\dots T}) = \prod_k q(Z_{k+1}|Z_{1\dots k}, X_{1\dots T})$. Optimal $I_{\text{free}} = I_{q\|p^*}(X_{1\dots T})$.
- **Online learning.** $q(Z_{1\dots T}|X_{1\dots T}) = \prod_k q(Z_{k+1}|Z_{1\dots k}, X_{1\dots k})$. Optimal $I_{\text{online}} > I_{\text{free}}$.
- **Limited memory.** $q(Z_{1\dots T}|X_{1\dots T}) = \prod_k q(Z_{k+1}|Z_k, X_k)$. Optimal $I_{\text{mem}} > I_{\text{online}}$.

Relative Information Rate

- Assume limited memory, i.e. Markov process $q(Z_{1\ldots T}|X_{1\ldots T}) = \prod_k q(Z_{k+1}|Z_k, X_k)$.
- Assume q has stationary distribution $\bar{\pi}$.
- Let $\bar{q}$ be same Markov process but with initial distribution $\bar{\pi}$.

- Using Kingman's subadditive ergodic theory[7] and under mild regularity conditions[8],

$$\lim_{T\to\infty} \frac{1}{T} I_{q\|p}(X_{1\ldots T}, Z_{1\ldots T}) = I_{\bar{q}\|p}(Z_2, X_2|Z_1, X_1).$$

- In continuous-time, we get the *(relative) information rate*

$$\lim_{T\to\infty} \frac{1}{T} I_{q\|p}(X_{1\ldots T}, Z_{1\ldots T}) = \frac{d}{dt} I_{\bar{q}\|p}(X_{1\ldots 1+t}, Z_{1\ldots 1+t})\Big|_{t=0}.$$

[7] https://en.wikipedia.org/wiki/Kingman%27s_subadditive_ergodic_theorem

[8] Brian G Leroux. "Maximum-likelihood estimation for hidden markov models." Stochastic processes and their applications, 40(1):127–143, 1992.

Stochastic Approximation

Setup. Parametric models $q_\lambda(Z_{n+1}|Z_n, X_n)$ and $p_\theta(Z_{n+1}, X_{n+1}|Z_n, X_n)$.

Goal. Minimize information $I_{\bar{q}\|p}(Z_2, X_2|Z_1, X_1)$.

Stochastic Approximation.[9]

1. Sample environment X_{n+1} from true distribution $q(X_{n+1}|X_n)$.
2. Sample memory Z_{n+1} from discriminatory distribution $q_\lambda(Z_{n+1}|Z_n, X_n)$.
3. Sample the generator gradient $\nabla_\theta I_{\bar{q}\|p}(Z_2, X_2|Z_1, X_1)$ using Z_{n+1}, X_{n+1}.
4. Sample the discriminator gradient $\nabla_\lambda I_{\bar{q}\|p}(Z_2, X_2|Z_1, X_1)$ using Z_{n+1}, X_{n+1}.
5. Update parameters θ, λ and repeat until convergence.

[9] Robbins, Herbert, and Sutton Monro. "A stochastic approximation method." The annals of mathematical statistics (1951): 400-407.

Stochastic Gradients

Generator.

$$\nabla_\theta I_{\bar{q}\|p}(Z_2, X_2|Z_1, X_1) = \lim_{T\to\infty} \mathbb{E}_q \left[\nabla_\theta \log p_\theta(Z_{T+1}, X_{T+1}|Z_T, X_T)\right]$$

Discriminator.

$$\nabla_\lambda I_{\bar{q}\|p}(Z_2, X_2|Z_1, X_1) = \lim_{T\to\infty} \mathbb{E}_q \left[\underbrace{\left(\sum_{i=1}^{T} \nabla_\lambda \log q_\lambda(Z_{i+1}|Z_i, X_i) \right)}_{\text{momentum}} \underbrace{\log \frac{q_\lambda(Z_{T+1}, X_{T+1}|Z_T, X_T)}{p_\theta(Z_{T+1}, X_{T+1}|Z_T, X_T)}}_{\text{surprise}} \right]$$

- Use discounted momentum (scale summands by some τ^{T-i} with $\tau<1$) for numerical stability.
- Same as reinforcement learning with surprise as reward (policy gradient for average reward)[10].

[10] Karimi, Belhal, Blazej Miasojedow, Eric Moulines, and Hoi-To Wai. "Non-asymptotic analysis of biased stochastic approximation scheme." PMLR 2019 pp. 1944-1974.

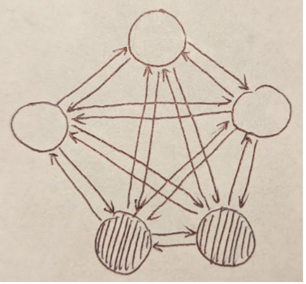
Online Learning

1. Get next observation.
$$X_{n+1} \sim q(X_{n+1}|X_n)$$
2. Sample next memory state.
$$Z_{n+1} \sim q_{\lambda_n}(Z_{n+1}|Z_n, X_n)$$
3. Update generator.
$$\theta_{n+1} = \theta_n + \eta_{n+1} \frac{d}{d\theta} \log p_\theta(Z_{n+1}, X_{n+1}|Z_n, X_n)\bigg|_{\theta=\theta_n}$$
4. Update momentum.
$$\alpha_{n+1} = \tau\alpha_n + \frac{d}{d\lambda} \log q_\lambda(Z_{n+1}|Z_n, X_n)\bigg|_{\lambda=\lambda_n}$$
5. Update surprise.
$$\gamma_{n+1} = \xi(X_{n+1}|X_n) + \log \frac{q_{\lambda_n}(Z_{n+1}|Z_n, X_n)}{p_{\theta_n}(Z_{n+1}, X_{n+1}|Z_n, X_n)}$$
$\xi(X_{n+1}|X_n)$ is any estimate of $\log q(X_{n+1}|X_n)$
6. Update discriminator.
$$\lambda_{n+1} = \lambda_n - \eta_{n+1}\alpha_{n+1}\gamma_{n+1}$$

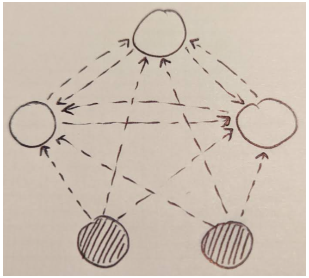
Part III

Online Spike Learning

Discriminator Networks



(a) Generator Network



(b) Discriminator Network

- To train our *generative* spiking network with Amari's *em* algorithm, we introduce a *discriminator* network — it has the same neurons but a different set of synapses.
- Let the weights and transition rates of the generative network be $w_{ij}^{(p)}$ and $\Gamma_{cc'}^{(p)}$ respectively.
- Let the weights and transition rates of the discriminative network be $w_{ij}^{(q)}$ and $\Gamma_{cc'}^{(q)}$ respectively.
- Idea of introducing discriminator networks is not new — see Rezende & Gerstner 2014.[11]

[11] Rezende, D. J., and W. Gerstner. "Stochastic variational learning in recurrent neural networks." Frontiers Comput. Neurosci., 8:38, 2014.

Information Rate

- Recall that the probability of a path $x_{0\dots T}$ with jumps $c_0, c_1, \dots, c_n$ is

$$p(x_{0\dots T}) = \pi(c_0)\,\frac{(\gamma T)^n}{n!} e^{-\gamma T} \prod_{i=0}^{n-1} P_{c_i c_{i+1}}, \quad \text{where } P_{cc'} = \begin{cases} \Gamma_{cc'}/\gamma & \text{if } c' \neq c, \\ 1 - \Gamma_{cc}/\gamma & \text{otherwise.} \end{cases}$$

- Using this path distribution, we can show that the (relative) information rate is

$$\begin{aligned} \frac{d}{dt} I_{\bar{q}\|p}(X_{0\dots t})\Big|_{t=0} &= \lim_{s\to 0} \frac{1}{s} I_{\bar{q}\|p}(X_{0\dots s}|X_0 = c_0) \\ &= \lim_{s\to 0} \frac{1}{s} \sum_{c_0} \bar{\pi}(c_0) \sum_{x_{0\dots T}} q(x_{0\dots s}|c_0) \log \frac{q(x_{0\dots s}|c_0)}{p(x_{0\dots s}|c_0)} \\ &= \sum_{c} \bar{\pi}(c) \left[\Gamma_{cc}^{(q)} - \Gamma_{cc}^{(p)} + \sum_{c'\neq c} \Gamma_{cc'}^{(q)} \log \frac{\Gamma_{cc'}^{(q)}}{\Gamma_{cc'}^{(p)}} \right] \end{aligned}$$

- Using the information rate and Amari's *em* algorithm, we derive an online learning algorithm for spiking networks in continuous-time.

Continuous-Time Spike Learning

1. At all time, compute

$$\rho_{jt}^{(p)} = \rho_0 \exp(\beta u_{jt}^{(p)}), \quad u_{jt}^{(p)} = u_0 + \textstyle\sum_{ij \in E} w_{ij}^{(p)} c_{ijt}$$
$$\rho_{jt}^{(q)} = \rho_0 \exp(\beta u_{jt}^{(q)}), \quad u_{jt}^{(q)} = u_0 + \textstyle\sum_{ij \in E} w_{ij}^{(q)} c_{ijt}$$
$$\dot{w}_{ijt}^{(p)} = -\eta_t \, \beta \, \rho_{jt}^{(p)} \, c_{ijt}$$
$$\dot{w}_{ijt}^{(q)} = -\eta_t \, \alpha_{ijt} \, \gamma_t, \quad \dot{\alpha}_{ijt} = -\beta \, \rho_{jt}^{(q)} \, c_{ijt} - \epsilon \alpha_{ijt}, \quad \gamma_t = \textstyle\sum_{j \in V} \rho_{jt}^{(p)} - \rho_{jt}^{(q)}$$

2. Environmental neurons spike with unknown rate

3. Memory neurons spike with rate $\rho_{jt}^{(q)}$

4. When some neuron j (environment or memory) spikes, update

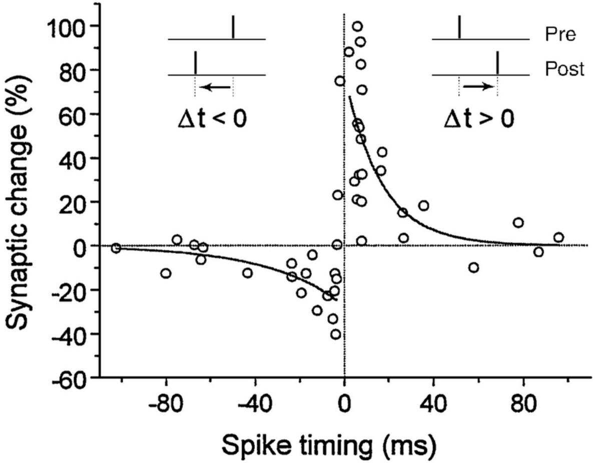
$$w_{ijt}^{(p)} \mathrel{+}= \eta_t \, \beta \, c_{ijt}$$
$$w_{ijt}^{(q)} \mathrel{+}= -\eta_t \, \alpha_{ijt} \, \gamma_t, \quad \dot{\alpha}_{ijt} = \beta \, c_{ijt}, \quad \gamma_t = \beta(u_{jt}^{(q)} - u_{jt}^{(p)})$$

Ignore $w_{ijt}^{(q)}$ update when neuron j is environmental.

Spike-Timing-Dependent Plasticity

Conjecture. Our learning algorithm explains STDP.

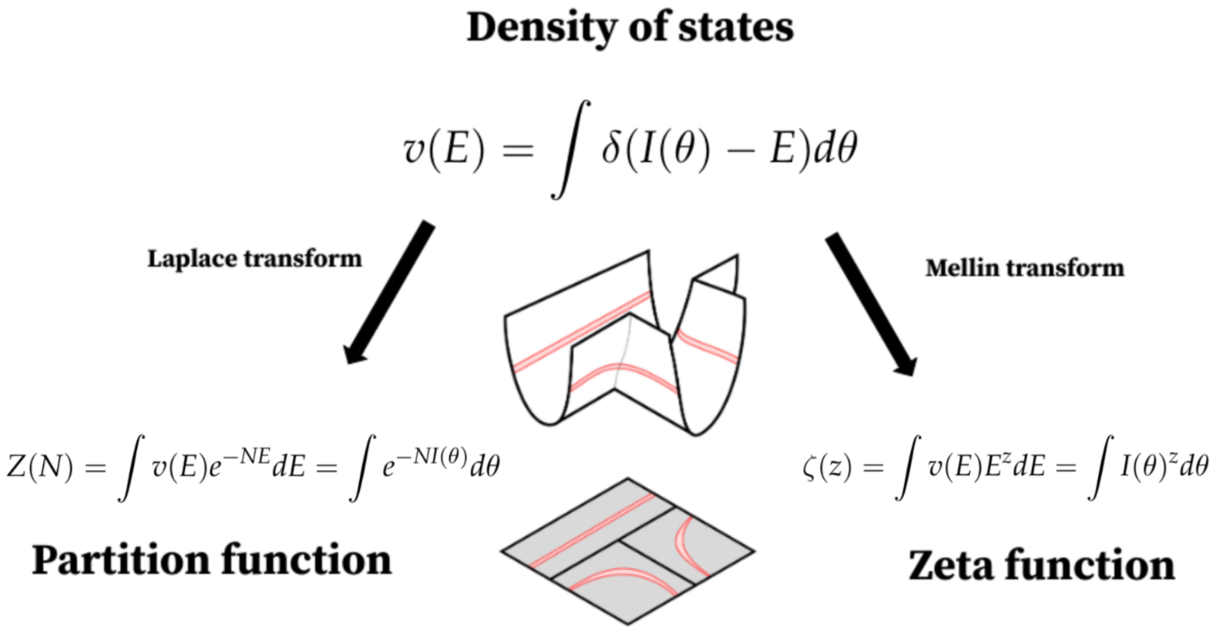
- Learning not accomplished by *pre-before-post* and *post-before-pre* rules.
- Learning accomplished by *gradual weight decay* and *post-spike increment*.



Conjectures and Extensions

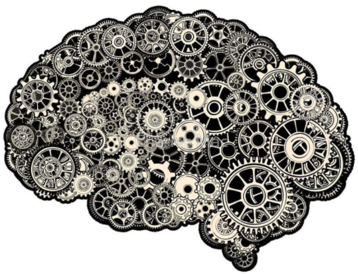
- **Conjecture.** Our learning algorithm also explains the triplet rule of STDP.
- **Conjecture.** Discriminator surprise explains dopamine-based neuromodulation.
- Think of the counts c_{ijt} as *spike credits*. Build a model where the credits decay with time, and derive the corresponding learning algorithm.
    - **Conjecture.** Credit decay is achieved with *adaptation* potentials.
    - **Conjecture.** Credit decay explains refractoriness after a spike.
    - **Conjecture.** Credit decay gives rise to discounted momentum.

Singular Learning



[12] Jesse Hoogland, "Physics I: The Thermodynamics of Learning",Singular Learning Theory and Alignment Summit 2023.

Thank you!



shaoweilin.github.io

Part IV

Appendix

Relative Information Rate

The relative information rate is given by

$$\frac{d}{dt} I_{\bar{q}\|p}(X_{0\ldots t})\Big|_{t=0} = \lim_{s\to 0} \frac{1}{s} \left[I_{Q\|P}(X_{0\ldots s}) - I_{Q\|P}(X_0)\right]$$

Using the chain rule $I(X,Y) = I(Y|X) + I(X)$,

$$\begin{aligned}\frac{d}{dt} I_{\bar{q}\|p}(X_{0\ldots t})\Big|_{t=0} &= \lim_{s\to 0} \frac{1}{s} I_{\bar{q}\|p}(X_{0\ldots s}|X_0) \\ &= \lim_{s\to 0} \frac{1}{s} \sum_{c_0} \bar{\pi}(c_0) \sum_{x_{0\ldots T}} q(x_{0\ldots s}|c_0) \log \frac{q(x_{0\ldots s}|c_0)}{p(x_{0\ldots s}|c_0)}\end{aligned}$$

Let Γ^*, Γ be the transition rate matrices of q, p respectively. Let $\delta = 1/\gamma$ and

$$P_{cc'} = \begin{cases} \delta\Gamma_{cc'} & \text{if } c' \neq c, \\ 1 - \delta\Gamma_{cc} & \text{otherwise} \end{cases}, \quad P^*_{cc'} = \begin{cases} \delta\Gamma^*_{cc'} & \text{if } c' \neq c, \\ 1 - \delta\Gamma^*_{cc} & \text{otherwise.} \end{cases}$$

Relative Information Rate

We expand the relative information

$$\sum_{x_{0\ldots T}} q(x_{0\ldots s}|c_0) \log \frac{q(x_{0\ldots s}|c_0)}{p(x_{0\ldots s}|c_0)}$$
$$= \sum_{n=0}^{\infty} \sum_{y_1,\ldots,y_n} \frac{(s/\delta)^n}{n!} e^{-s/\delta} \; \prod_{i=0}^{n-1} P^*_{y_i y_{i+1}} \log \frac{\prod_{i=0}^{n-1} P^*_{y_i y_{i+1}}}{\prod_{i=0}^{n-1} P_{y_i y_{i+1}}}.$$

When $n = 0$, the summand vanishes because the right-most factor is $\log 1 = 0$. The higher order terms for $n \geq 2$ vanish in the limit as $s \to 0$. Hence,

$$\lim_{s\to 0} \frac{1}{s} I_{Q\|P}(X_{0\ldots s}|X_0 = x_0) = \sum_{y_1} \frac{1}{\delta} P^*_{y_0 y_1} \log \frac{P^*_{y_0 y_1}}{P_{y_0 y_1}}$$

Relative Information Rate

When $y_0 = y_1$, we have $P^*_{y_0y_0} = 1 - \delta\Gamma^*_{y_0y_0} \approx e^{\delta\Gamma^*_{y_0y_0}}$, so for very small δ,

$$\frac{1}{\delta} P^*_{y_0y_1} \log \frac{P^*_{y_0y_1}}{P_{y_0y_1}} \approx \frac{1}{\delta} e^{\delta\Gamma^*_{y_0y_0}} \log \frac{e^{\delta\Gamma^*_{y_0y_0}}}{e^{\delta\Gamma_{y_0y_0}}}$$

$$= e^{\delta\Gamma^*_{y_0y_0}} \left(\Gamma^*_{y_0y_0} - \Gamma_{y_0y_0}\right) \approx \Gamma^*_{y_0y_0} - \Gamma_{y_0y_0}$$

When $y_0 \neq y_1$, we have $P^*_{y_0y_1} \approx \delta\Gamma^*_{y_0y_1}$, so

$$\frac{1}{\delta} P^*_{y_0y_1} \log \frac{P^*_{y_0y_1}}{P_{y_0y_1}} = \frac{1}{\delta}\, \delta\Gamma^*_{y_0y_1} \log \frac{\delta\Gamma^*_{y_0y_1}}{\delta\Gamma_{y_0y_1}} = \Gamma^*_{y_0y_1} \log \frac{\Gamma^*_{y_0y_1}}{\Gamma_{y_0y_1}}$$

Putting it all together,

$$\frac{d}{dt} I_{\bar{q}\|p}(X_{0\ldots t})\bigg|_{t=0} = \sum_c \bar{\pi}(c) \left[\Gamma^*_{cc} - \Gamma_{cc} + \sum_{c' \neq c} \Gamma^*_{cc'} \log \frac{\Gamma^*_{cc'}}{\Gamma_{cc'}}\right]$$