Anuran Makur
6 Dec 2017

Spiking Neural Networks: [Lynch-Musco-Parter 2017]

- focus on specific algorithmic tasks and analysis [not general computational/learning ability]
- static: synapse weights fixed [not learning]
- stochastic: neurons fire probabilistically in discrete pulses
  (note: in real systems, noise is at the synapses, not neurons)

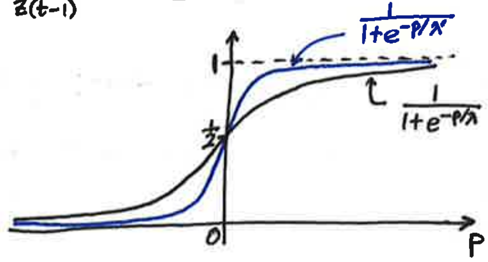
Network Model:

- n input neurons: $X = \{x_1, \ldots, x_n\}$ where each $x_i \in \{0,1\}$
- m output neurons: $Y = \{y_1, \ldots, y_m\}$ where each $y_i \in \{0,1\}$
- ℓ auxiliary neurons: $Z = \{z_1, \ldots, z_\ell\}$ where each $z_i \in \{0,1\}$
- weight function $w: (X \cup Y \cup Z)^2 \rightarrow \mathbb{R}$ that describes the directed weighted (synaptic) edges between X, Y, Z
  (note: weight $= 0 \Leftrightarrow$ no connection/edge)
- activation bias $b: Y \cup Z \rightarrow \mathbb{R}_+$ (For neuron $v \in Y \cup Z$, its potential must reach $b(v)$ for a spike to occur with 'good' probability.)
- restrictions:
  1. Indegrees of all input neurons in X are 0, i.e. $w(u,x) = 0$ for all $u \in X \cup Y \cup Z$ and $x \in X$
     ↳ feedback to inputs can be replicated using intermediate neurons
  2. Every neuron $v \in X \cup Y \cup Z$ is either inhibitory: $w(v,u) \leq 0$ for all $u \in X \cup Y \cup Z$,
     or excitatory: $w(v,u) \geq 0$ for all $u \in X \cup Y \cup Z$.
  3. All input and output neurons are excitatory.

Network Dynamics:

- An SNN (or specifically its neurons) is a discrete-time Markov chain.
- At $t = 0$, $x_i(0) = \begin{cases} 1, & i\text{th input fires} \\ 0, & \text{ow} \end{cases}$ for all $i = 1, \ldots, n$. For any $v \in Y \cup Z$, $v(0)$ is arbitrary.
- For $t > 0$, $x_i(t) = x_i(t-1)$ for all $i = 1, \ldots, n$.
- For $t > 0$, and any neuron $v \in Y \cup Z$, define the potential: $\text{pot}(v,t) \triangleq \underbrace{\sum_{u \in X \cup Y \cup Z} w(u,v)\, u(t-1)}_{\text{sum over input neurons}} - b(v)$.

  Furthermore, $\underset{\substack{\text{conditioned} \\ \text{on } X(t-1), Y(t-1), \\ Z(t-1)}}{\mathbb{P}}(\underbrace{v(t) = 1}_{\substack{\text{neuron } v \\ \text{fires at time } t}}) = \dfrac{1}{1 + \exp(-\text{pot}(v,t)/\lambda)}$, where $\lambda > 0$ is a fixed temperature parameter.



$\lambda' < \lambda$
[Smaller $\lambda \Rightarrow$ steeper sigmoid]

So, v fires at time t with 'good' probability if $\text{pot}(v,t) > \underbrace{\varepsilon(\lambda)}_{\text{constant that depends on } \lambda}$.

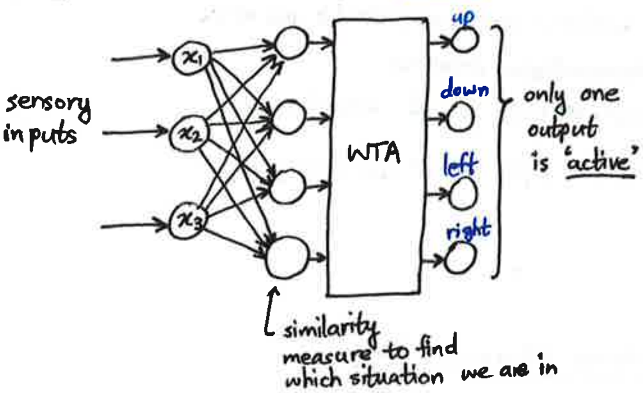
- Given the values $X(t), Y(t), Z(t)$, the neurons $X(t+1), Y(t+1), Z(t+1)$ fire independently.

Computational Problem:

- Given a (possibly multi-valued) target function $f: \{0,1\}^n \to \{0,1\}^m$.
- Design an SNN with a small number of auxiliary neurons so that given input $X\ (= \{x_1(0), \ldots, x_n(0)\})$, $Y(t)\ (= \{y_1(t), \ldots, y_m(t)\})$ converges quickly to $f(X)$ (or any string in $f(X)$).
- An SNN converges in t rounds if for any input X and any $t' \geq t$, $Y(t') = f(X)$ w.h.p..
- w.h.p. = with high probability, i.e. with probability $\geq 1 - \frac{1}{n^c}$ for some constant c.
- Goal: Tradeoff between ℓ and convergence time.

Winner-Take-All Networks:

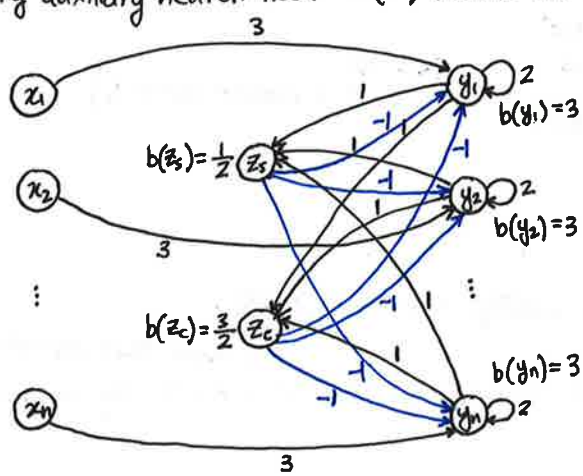
- General idea: system of neurons/cells receive inputs; cells compete so that highest input cell elicits a response. WTA network is a 'maximum decoder.'
- Biologically motivated: decision-making or action selection in the brain, eg. vision, perception, attention.
- Competitive network example: Robot action selection



- Boolean problem: $X \in \{0,1\}^n$, want to converge to $Y \in \{0,1\}^n$ (i.e. $m = n$) such that Y has a single firing neuron corresponding to any firing neuron in X. (note: X has ≥ 1 firing neurons)

- Thm: There exists an SNN with $\ell = 2$ inhibitory auxiliary neurons, that given input $X \in \{0,1\}^n$, converges in $O(\log^2(n))$ rounds to a valid WTA output Y w.h.p.. (In contrast, any SNN with $\ell = 1$ inhibitory auxiliary neuron needs $\Omega(n^c)$ rounds for convergence.)

Proof idea:



z_s = stability inhibitor – WTA state stabilises once reached

z_c = convergence inhibitor – drives convergence & competition to single winning output

[continued.]

Proof idea continued:

- Set $\lambda = \dfrac{1}{K\log(n)}$ for some constant $K > 0$. Then, for any neuron $u \in Y \cup Z$, if $pot(u,t) \geq \frac{1}{2}$,

  then $$\underline{\mathbb{P}(u(t)=1)} = \frac{1}{1+\exp(-pot(u,t)K\log(n))} = \frac{1}{1+\frac{1}{n^{K pot(u,t)}}} \geq \frac{1}{1+\frac{1}{n^{K/2}}} = \frac{n^{K/2}}{n^{K/2}+1} = 1 - \frac{1}{1+n^{K/2}} \geq \underline{1 - \frac{1}{n^{K/2}}}.$$

  So, <u>u fires w.h.p. at time t</u> if $pot(u,t) \geq \frac{1}{2}$.

- $z_s(t) = 1$ w.h.p. whenever <u>at least one</u> of the outputs fires at time $t-1$, i.e. $\exists i,\ y_i(t-1) = 1$.
  $z_c(t) = 1$ w.h.p. whenever <u>at least two</u> of the outputs fire at time $t-1$, i.e. $\exists i, \exists j \neq i,\ y_i(t-1) = y_j(t-1) = 1$.
  [<u>Reason:</u> See bias values and input edges of z_s & z_c.]

- If input $x_i(0) = 0$, then $y_i(t) = 0$ w.h.p. for all $t \geq 1$ as the bias of y_i is 3.
  So, only output neurons corresponding to <u>firing</u> inputs ever fire w.h.p..

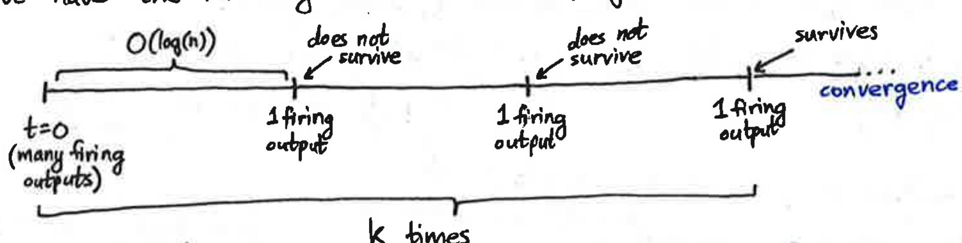
- Suppose $z_s(t) = z_c(t) = 1$. Then, any y_i such that $y_i(t) = 1$ will have $pot(y_i, t+1) = 0$ w.h.p. (as $x_i(t) = 1$ w.h.p.). Hence, such y_i will fire at round $t+1$ with probability $\mathbb{P}(y_i(t+1) = 1) = \frac{1}{2}$.
  If we start with n firing outputs, at each round, we reduce the number of firing outputs by about $\frac{1}{2}$. Thus, after <u>$O(\log(n))$ rounds,</u> we are left with <u>one firing output</u> w.h.p..

- If t is the first round when only one output fires, then $z_s(t) = z_c(t) = 1$ and the firing output only has probability $\frac{1}{2}$ of surviving in round $t+1$.

  <u>Case 1:</u> (single firing output survives) $\exists i,\ y_i(t) = y_i(t+1) = 1$ is the only firing output.
  Then, $z_s(t+1) = 1$ and $z_c(t+1) = 0$ w.h.p.. As a result, for all $t' \geq t+1$, $y_i(t') = 1$ is the only firing output w.h.p.. [convergence!]
  ↳ [See weight & bias values.]

  <u>Case 2:</u> (single firing output does not survive)
  We consider this as a <u>reset</u> to the initial setting.

- We have the following events as rounds progress.



<u>Want:</u> $(1-\underbrace{\mathbb{P}(\text{survive})}_{\text{constant in } n})^k \approx \frac{1}{n^c} \Leftrightarrow k = O(\log(n))$.
↑ small prob. of failing to survive k times

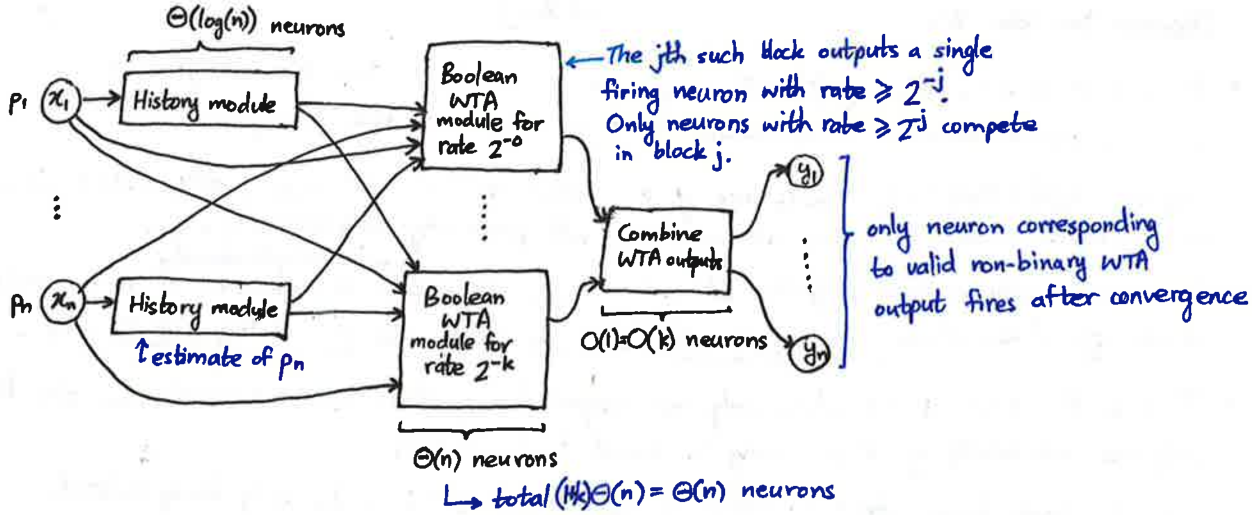
$\therefore$ After $kO(\log(n)) = \underline{O(\log^2(n))\ \text{rounds}}$, we have convergence w.h.p..

∎

Anuran Makur
6 Dec 2017

- **Non-Binary WTA:**
  - Suppose each $x_i \in X$ fires independently with probability $p_i \in [0,1]$ in each round.
  - Want: Design an SNN that converges to the WTA output: a single neuron fires corresponding to an input neuron with maximum p_i.
  - Suppose each $p_i = 2^{-j}$ for some $j \in \{0, \dots, k\}$ with $k = O(1)$.
  - Thm: There exists an SNN with $\ell = O(n\log(n))$ auxiliary neurons such that the output converges in $O(\log^2(n))$ rounds to a valid WTA output Y w.h.p..

Idea:



Neuro-RAM:

- **Motivation: (Similarity Testing)**
  - Given $X_1, X_2 \in \{0,1\}^n$, test if $X_1 = X_2$ or if $d_H(X_1, X_2) \geq n\varepsilon$, for some $\varepsilon > 0$. (Hamming distance)
  - Randomized algorithm: Select $\Theta\left(\frac{\log(n)}{\varepsilon}\right)$ indices at random.
    If X_1, X_2 match at all indices, declare $X_1 = X_2$.
    If there is ≥ 1 index where X_1, X_2 do not match, declare $d_H(X_1, X_2) \geq n\varepsilon$.

    Reason: $\mathbb{P}(X_1, X_2 \text{ match at all } k \text{ indices} \mid d_H(X_1,X_2) \geq n\varepsilon) \leq \frac{(1-\varepsilon)n}{n} \times \frac{(1-\varepsilon)n - 1}{n} \times \cdots \times \frac{(1-\varepsilon)n - k + 1}{n}$, where $k = \Theta\left(\frac{\log(n)}{\varepsilon}\right)$

$$
\begin{aligned}
&= (1-\varepsilon)^k \left(1 \times \left(1 - \frac{1}{n(1-\varepsilon)}\right) \times \cdots \times \left(1 - \frac{k-1}{n(1-\varepsilon)}\right)\right) \\
&\leq (1-\varepsilon)^k \\
&= (1-\varepsilon)^{C\log(n)/\varepsilon} \qquad [k = C\log(n)/\varepsilon \text{ for some } C > 0] \\
&= \exp\Big(C\log(n)\underbrace{\frac{\log(1-\varepsilon)}{\varepsilon}}_{\leq -1}\Big) \\
&\leq \frac{1}{n^C}
\end{aligned}
$$

$\Rightarrow \mathbb{P}(\text{declare } d_H(X_1,X_2) \geq n\varepsilon \mid d_H(X_1,X_2) \geq n\varepsilon) \geq 1 - \frac{1}{n^C}$

So, given $X_1 = X_2$, the algorithm succeeds a.s., and given $d_H(X_1,X_2) \geq n\varepsilon$, the algorithm succeeds w.h.p..

  - To implement this algorithm via an SNN, we need to:
    1. generate random indices: use $\log(n)$ neurons with potential 0 (as prob. of firing is $\frac{1}{2}$).
    2. construct neuro-RAM: Given $X \in \{0,1\}^n$ and index $Y \in \{0,1\}^{\log(n)}$, output bit of X at index Y.
       ↳ This is also a multiplexer!

- Thm: There is an SNN, that has $O(\sqrt{n})$ auxiliary neurons and converges in $O(\sqrt{n})$ rounds w.h.p., which implements a neuro-RAM.

  Cor: There is an $O\left(\frac{\sqrt{n}\log(n)}{\varepsilon}\right)$ neuron SNN for similarity testing.

- Thm: Any SNN neuro-RAM converging in $\sqrt{n}$ rounds w.h.p. must use $\Omega\left(\frac{\sqrt{n}}{\log^2(n)}\right)$ auxiliary neurons.