Brain-Inspired Learning Machines

Self-Organizing Maps and Competitive Learning

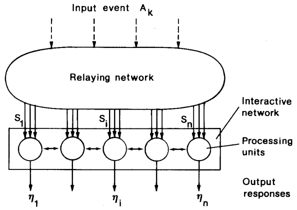
Emre Neftci

Department of Cognitive Sciences, UC Irvine,

November 4, 2016

What is Competitive Learning

A family of **unsupervised** learning techniques in neural networks to discover underlying structure in the data.



During training, the neuron with the highest activation to an input is selected and its weights are adjusted to match the input.
Two simultaneous mechanisms: Competition and plasticity

- Unsupervised because labels are not necessary for the learning task
- Nodes that are “close” to each other respond similarly
- Competition is typically implemented through "winner-take-all" connectivity

Competitive Learning Algorithm

Learning Rule:

$$\Delta \mathbf{w}_k \propto \theta(i,k)(\mathbf{s} - \mathbf{w}_k) \tag{1}$$

where:

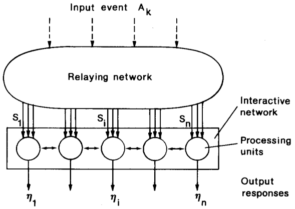
- k is the index of a neuron
- i is the index of *the winning neuron*
- $\theta(i,k)$ is a “closeness” metric ($\theta(i,i) \leq \theta(i,k) \forall k$)
- $\mathbf{s}$ is an input vector
- $\mathbf{w}_k$ is the weight vector of neuron k

Algorithm:

1. Choose data sample s
2. Find “winning neuron” index i
3. Apply $\Delta \mathbf{w}_k$ for all neurons k in the network

Historical perspective: Self-Organizing (Kohonen) Maps

- Orginally inspired by the formation of topographic projections between two laminar structures in the brain
- Useful for creating spatially organized "internal representations" of various features of input signals and their abstractions.



Kohonen 1982

Kohonen 1990

Many names: Kohonen map, Self-Organizing Feature Map, and Adaptive Resonance Theory (Grossberg et al.)

Self-organzing maps for learning topographic maps in the brain

Topographic map = Neighborhood relationship.

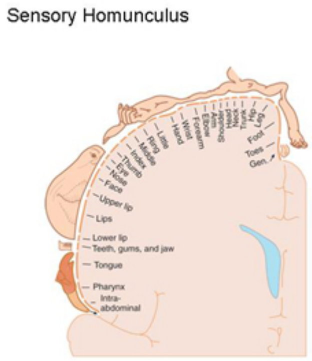


Figure 11-4. In: Barrett KE, Barman SM, Boitano S, Brooks H. *Ganong's Review of Medical Physiology*. 23rd ed. http://www.acesspharmacy.com. Accessed October 29, 2009.

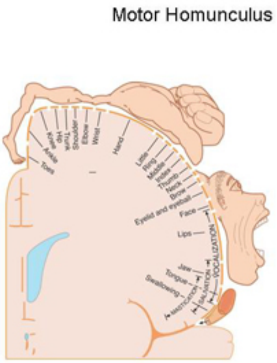
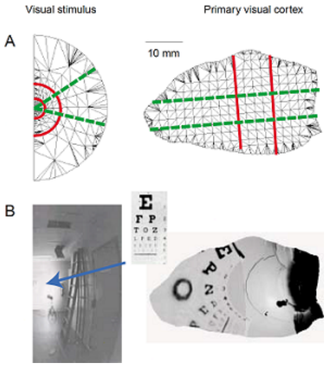


Figure 11-5. In: Barrett KE, Barman SM, Boitano S, Brooks H. *Ganong's Review of Medical Physiology*. 23rd ed. http://www.acesspharmacy.com. Accessed October 29, 2009.

Retinotopic Map

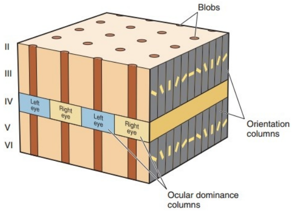


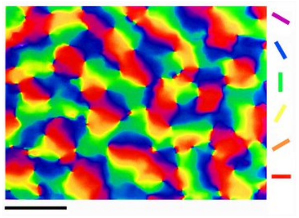
Matteo Carandini (2012), Scholarpedia, 7(7):12105

Nearby points in visual field project to nearby neurons in V1

Self-organizing maps to learn topographic maps without supervision, given some information about the underlying dimensionality of the data

Columnar Organization of V1



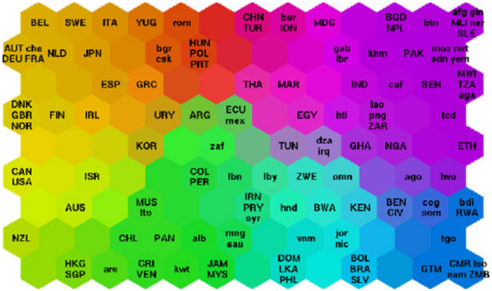


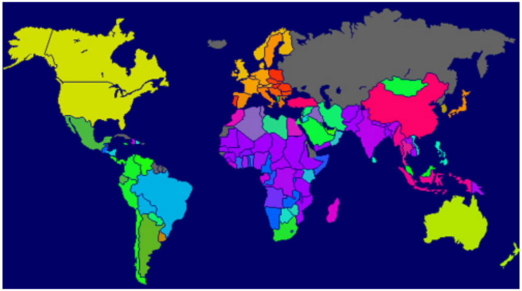
Right: Blasdel & Salama (1986)

The receptive fields are tiled to cover the entire visual field

Demonstration with pyMVPA2

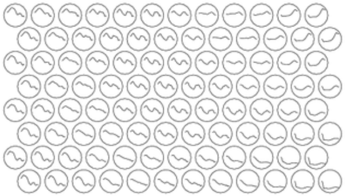
SOM examples

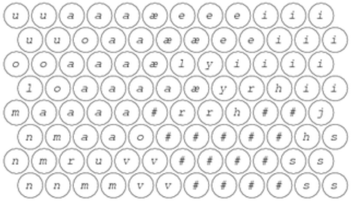




http://www.ai-junkie.com/

Models of acoustic spectra of Finnish phonemes, organized on an SOM.





Teuvo Kohonen and Timo Honkela (2007), Scholarpedia, 2(1):1568.

Competitive learning as an example of generalized Hebb rule

Competitive Learning:

$$\Delta \mathbf{w}_j \propto \theta(i,j)(\mathbf{s} - \mathbf{w}_j) \tag{2}$$

Generalized Hebb Rule:

$$\frac{\mathrm{d}}{\mathrm{d}t} w_{ij}(t) \propto a_0 + a_1^{pre} \nu_j + a_1^{post} \nu_i + a_2 \nu_i \nu_j + \ldots \tag{3}$$

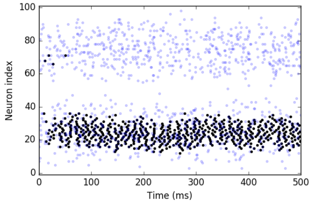
Competitive learning can be written as a generalized Hebb rule:

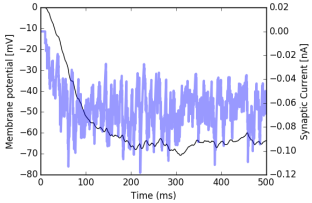
$$\begin{aligned} \nu_i &= \theta(i,j) \\ \nu_j &= s_j \\ a_0 &= a_1^{pre} = 0 \\ a_2 &= 1 \\ a_1^{post} &= -w_{ij} \end{aligned} \tag{4}$$

Example Spiking Neural Network

code/brian2_snn_wta.py

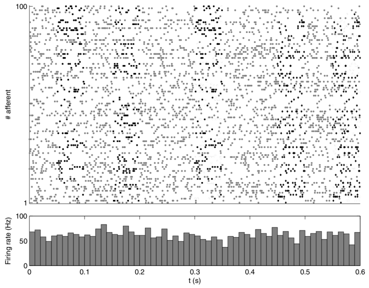
```
input_profile = (np.cos(np.linspace(0+np.pi, 4*np.pi+np.pi, 100))+
Pin = PoissonGroup(100, rates = input_profile*25*Hz)
P = NeuronGroup(100, eqs, threshold='v>Vt', reset='v = Vr',
                    refractory=5*ms, method='euler')

wff = .1 * nA
Sff = Synapses(Pin, P, on_pre='isyn += wff')
Sff.connect('i==j') #connect 1 to 1

wreci = -.01 * nA
#Inhibitory connections
Sreci = Synapses(P, P, on_pre='isyn += wreci')
Sreci.connect()

wrece = .02 * nA
```

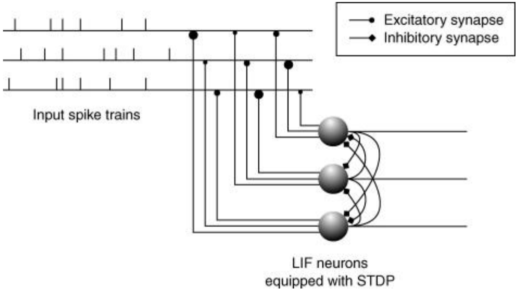




Competitive Learning in Spiking Neurons

Unsupervised Learning of Visual Features through Spike Timing Dependent Plasticity

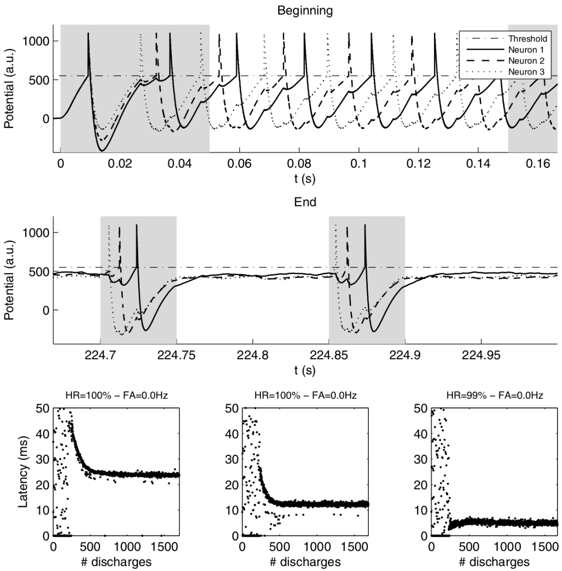




Thorpe et al. 2008

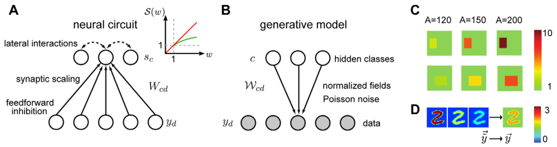
Competitive Learning in Spiking Neurons

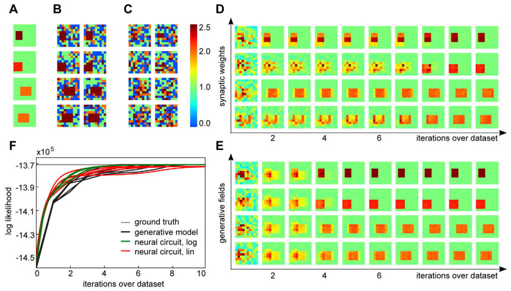
Unsupervised Learning of Visual Features through Spike Timing Dependent Plasticity



Thorpe et al. 2008

Competitive Learning in biological Neural Networks





Generalized Hebb Rule:

$$\Delta W_{cd} = \epsilon s_c (y_d - W_{cd})$$

Assuming input is normalized ($\sum_d y_d^{(n)} = A$)

Keck, Savin, and Lücke, *PLoS computational biology*, 2012

Machine learning foundations of competitive learning

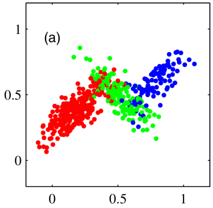
Inference in mixture models

A mixture model assumes that each input stimulus belongs to one out of C classes, where each class is described by a representative input and its typical variations

Example: *Mixture of Gaussians*

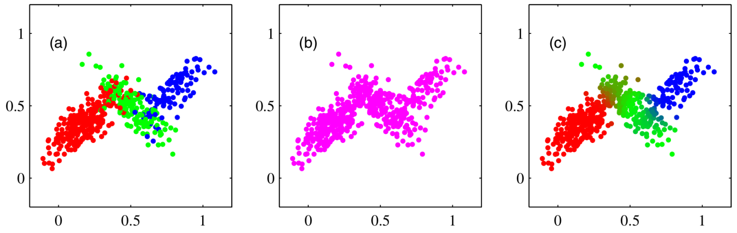
$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k N(\mathbf{x}|\mu_k, \Sigma_k) \tag{5}$$

where:
π_k is the "mixture coefficient", and
$N(\mathbf{x}|\mu_k, \Sigma_k)$ is the multivariate Normal density function evaluated at $\mathbf{x}$



Bishop,, 2006

Inference in mixture models



Bishop,, 2006

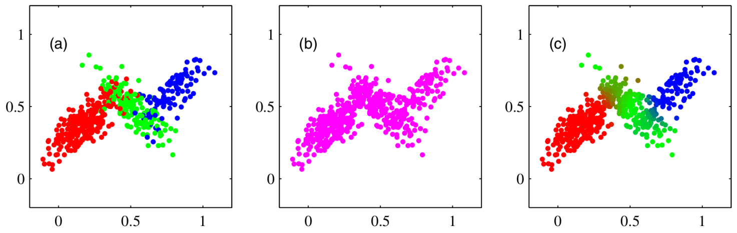
With Bayes rule, we find:

$$p(z_k = 1|\mathbf{x}) = \frac{\pi_k N(\mathbf{x}|\mu_k, \Sigma_k)}{p(\mathbf{x})} \tag{6}$$

where $\mathbf{z}$ is a 1-hot representation of the class k.

Once μ_k and Σ_k are known, we can infer the class k of a data sample $\mathbf{x}_n \in \mathbf{X}$

Maximum Likelihood of Mixture Models



Bishop,, 2006

Goal of learning: maximize the likelihood of data X by finding optimal parameters $\pi_k, \mu_k, \Sigma_k \quad \forall k$:

$$\log p(\mathbf{X}) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k N(\mathbf{x}|\mu_k, \Sigma_k) \right) \tag{7}$$

Expectation Maximization Algorithm

Expectation-Maximization: A general technique for finding maximum likelihood estimators in latent variable models (*e.g.* mixture models)

Dempster, Laird, and Rubin, *Journal of the Royal Statistical Society. Series B (Methodological)*, 1977

1. E-Step: evaluate

$$p(z_{nk} = 1|\mathbf{x}_n) = \frac{\pi_k N(\mathbf{x}_n|\mu_k, \Sigma_k)}{p(\mathbf{x}_n)}$$

2. M-Step: estimate new parameters

$$\mu_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} p(z_{nk} = 1|\mathbf{x}_n)\mathbf{x}_n$$

$$\Sigma_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} p(z_{nk} = 1|\mathbf{x}_n)(\mathbf{x}_n - \mu_k^{new})(\mathbf{x}_n - \mu_k^{new})^\top$$

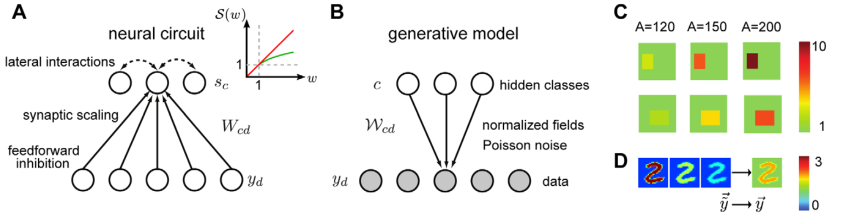
$$\pi_k^{new} = \frac{N_k}{N}$$

   with $N_k = \sum_{n=1}^{N} p(z_{nk} = 1|\mathbf{x}_n)$

3. Evaluate $\log p(\mathbf{X}|\Sigma, \mu, \pi)$ and repeat until convergence

Further reading (proof) in Bishop *et al.* 2006, Chap. 9

Competitive Learning Performs Expectation Maximization



Network computed weights:

$$W_{cd} \approx A \frac{\sum_n f_c(\vec{y}^{(n)}, W) y_d^{(n)}}{\sum_{d'} \sum_n f_c(\vec{y}^{(n)}, W) y_{d'}^{(n)}} \approx A \frac{\langle f_c(\vec{y}, W) y_d \rangle_{p(\vec{y})}}{\sum_{d'} \langle f_c(\vec{y}, W) y_{d'} \rangle_{p(\vec{y})}}, \qquad (5)$$

EM update (written more generally):

$$\mathrm{E-step:} \quad p(c|\vec{y}^{(n)}, \mathcal{W}) = \frac{\exp(I_c)}{\sum_{c'} \exp(I_{c'})}, \quad \text{where } I_c = \sum_d \log(\mathcal{W}_{cd}) y_d^{(n)} \qquad (8)$$

$$\mathrm{M-step:} \quad \mathcal{W}_{cd}^{\mathrm{new}} = A \frac{\sum_n p(c|\vec{y}^{(n)}, \mathcal{W}) y_d^{(n)}}{\sum_{d'} \sum_n p(c|\vec{y}^{(n)}, \mathcal{W}) y_{d'}^{(n)}}, \qquad (9)$$