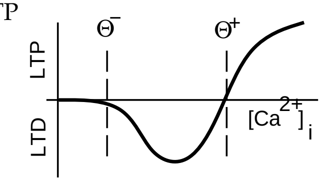
1 Combined Model & Task Learning

1. Pros and Cons: Use Both.
2. Inhibition is also an Important Bias.
3. Generalization & Deep Networks.

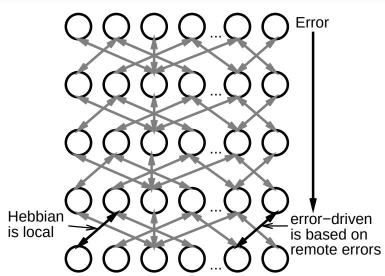
2 Biology Says: Both

	Plus Phase					
	$x_i^+, y_j^+ \approx 0$			$x_i^+, y_j^+ \approx 1$		
Minus Phase	Err	Hebb	Combo	Err	Hebb	Combo
$x_i^-, y_j^- \approx 0$	0	0	**0**	+	+	**+**
$x_i^-, y_j^- \approx 1$	–	0	**–**	0	+	**+**

No $Ca^{2+} \rightarrow$ no learning
Mod $Ca^{2+} \rightarrow$ LTD
High $Ca^{2+} \rightarrow$ LTP



3 Functional: Pros and Cons



	Pro	Con
Hebbian (local)	autonomous, reliable	myopic, greedy
Error-driven (remote)	task-driven, cooperative	co-dependent, lazy

Error-driven = Left-wing, Hebbian = Right-wing

4 Combining Error-driven + Hebbian

Get benefits of both:

$$\Delta w_{ij} \approx \Delta_{hebb} + \Delta_{err} \qquad (1)$$

$$\Delta_{hebb} = \epsilon a_j(a_i - w_{ij})$$

$$\Delta_{err} = \epsilon[(a_i^+ a_j^+) - (a_i^- a_j^-)]$$

$$\Delta w_{ij} = (k_{hebb})\Delta_{hebb} + (1 - k_{hebb})(\Delta_{err}) \qquad (2)$$

5

Inhibitory Competition as a Bias

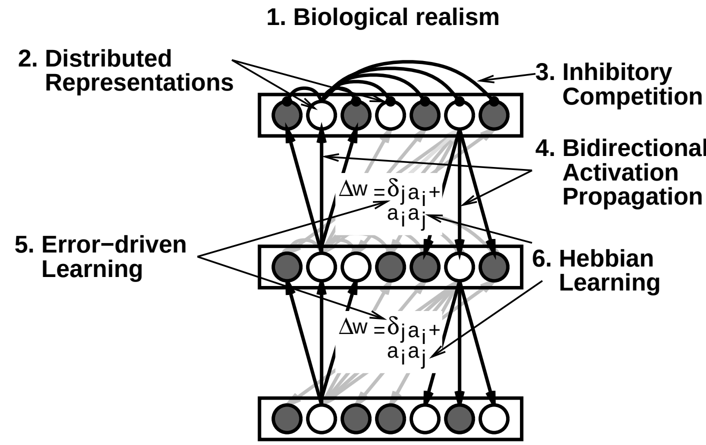
Inhibition:

- Causes sparse, distributed representations (many alternatives, only a few relevant at any time).
- Competition and specialization: survival of fittest.
- Self-organizing learning.

(Often more important than Hebbian bias)

6

The Whole Enchilada

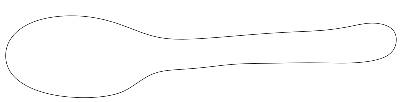


7

Generalization

How well do we deal with things we've never seen before?

nust



each time you walk into class, each social interaction, each sentence you hear, etc.

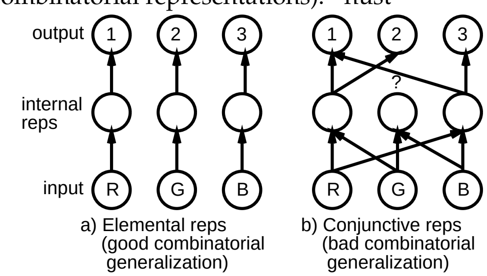
We're constantly faced with new situations, and generalize reasonably well to them.

How do we do it?

8

Generalization

Distributed reps: novel items are novel combinations of existing features (combinatorial representations): "nust"



Hebbian & inhibition: produce elemental, combinatorial reps.

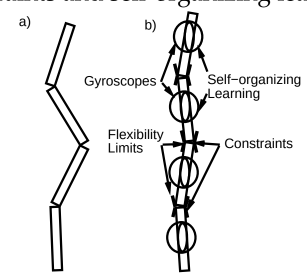
9 Sims: Generalization

10 Deep Networks

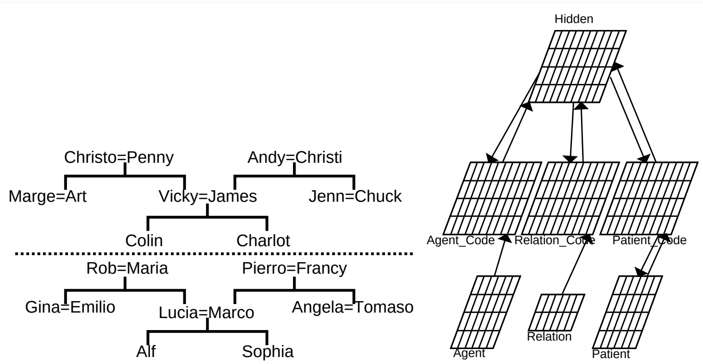
Need many hidden layers to achieve many stages of transformations (dramatically re-representing the problem).

But then the error signals are very remote & weak.

Need to add constraints and self-organizing learning:



11 Example: Family Trees



12 Sequential & Temporally-Delayed Learning

1. The Problem.
2. Sequential Learning & Context.
3. Temporally-delayed Learning & Reinforcement.

13 The Problem

Currently: networks learn *immediate* consequence of a given input.

What if current input only makes sense as part of a temporally-extended *sequence* of inputs?

What if the consequence of this input comes *later* in time?

14 Sequence Learning

How do we do it?
For example:

My favorite color is purple.
Purple my color favorite is.
Is my purple color favorite.
Is purple my color favorite.

The girl picked up the pen.
The pig raced around the pen.

We represent the *context*, not just the current input.

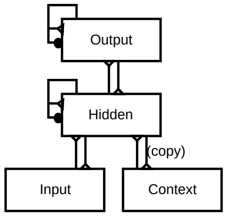
in language, social interactions, driving (who goes at a 4-way stop?)

15 Representing Context for Sequence Learning

How does the brain do it?
How would we get our models to do it?

Add layers to keep track of context (prefrontal cortex).

16 Representing Context for Sequence Learning



Simple Recurrent Network (SRN; Elman, Jordan).

$c_j(t) \approx h_j(t-1) + c_j(t-1)$

$c_j(t) = fm_{hid} h_j(t-1) + fm_{prv} c_j(t-1)$

17

An Example Task

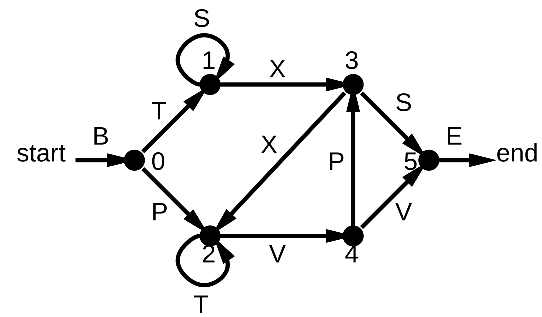
BTXSE
BPVPSE
BTSXXTVVE
BPTVPSE

Which of the following sequences are allowed?:
BTXXTTVVE
TSXSE
VVSXE
BSSXSE

18

An Example Task

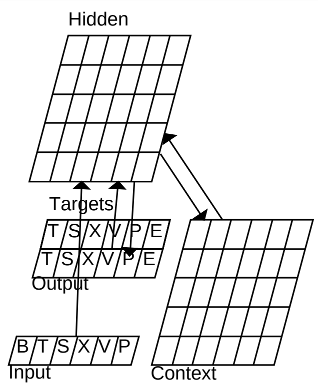
BTXSE, BPVPSE, BTSXXTVVE, BPTVPSE, BTXXTTVVE
TSXSE, VVSXE, BSSXSE



We implicitly learn such grammars (e.g., pressing buttons faster to letters that follow grammar).

19

The Network



Randomly chooses one of two possible next states.

Hidden/context units learn to encode states, not labels.

20

Time & Sequences

Currently: networks learn *immediate* consequence of a given input.

What if current input only makes sense as part of a temporally-extended *sequence* of inputs? (*context*)

What if the consequence of this input comes *later* in time?

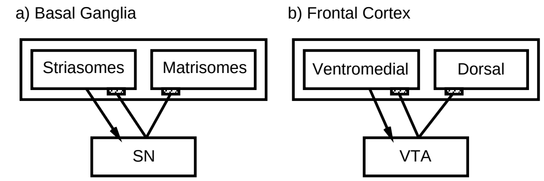
21 Temporally-delayed Learning & Reinforcement

Reinforcement often delayed from the action(s) that lead to it: need to "span the gap".

Key ideas: We want to predict rewards consistently over time. This process leads us to learn what events are associated with rewards, earlier and earlier back in time.

We use the Temporal Differences (TD) algorithm (Sutton).

22 Reinforcement Biology

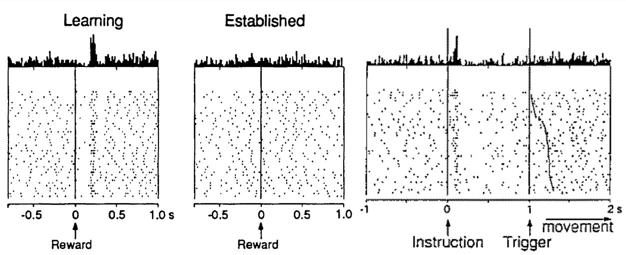


Midbrain dopimanergic (DA) systems modulate cortex & basal ganglia: Substantia Nigra (SN) & Ventral-Tegmental Area (VTA).

SN & VTA are controlled by other cortical/BG areas.

These other areas are like an "Adaptive Critic" (AC), which evaluates stimuli & actions for their rewarding value..

23 Actual Recordings from Actual Neurons



VTA firing moves from responding to reward to *anticipating* it at the instruction.

24 The Equations

Value function, sum of discounted future rewards:

$$V(t) = \langle \gamma^0 r(t) + \gamma^1 r(t+1) + \gamma^2 r(t+2) \ldots \rangle \qquad (3)$$

Recursive definition:

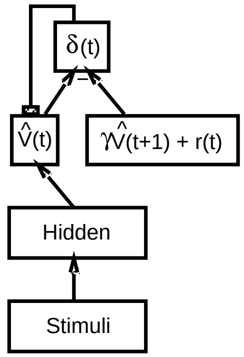
$$V(t) = \langle r(t) + \gamma V(t+1) \rangle \qquad (4)$$

Error in predicted reward:

$$\delta(t) = \Big(r(t) + \gamma \hat{V}(t+1) \Big) - \hat{V}(t) \qquad (5)$$

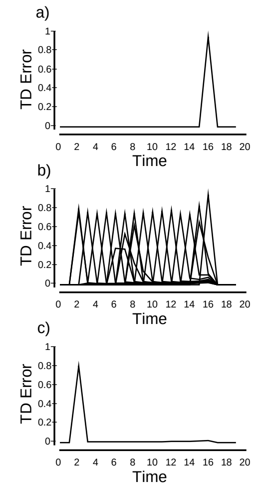
25

Network Implementation



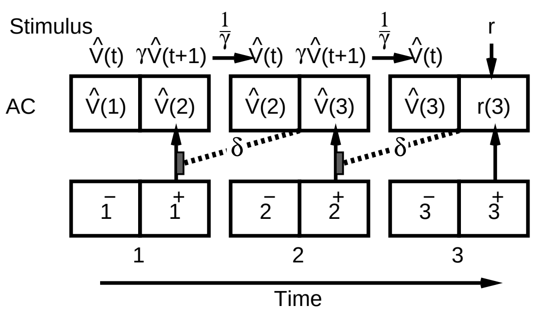
26

Model: CS at t=2, US at t=16



27

Phase-based Implementation



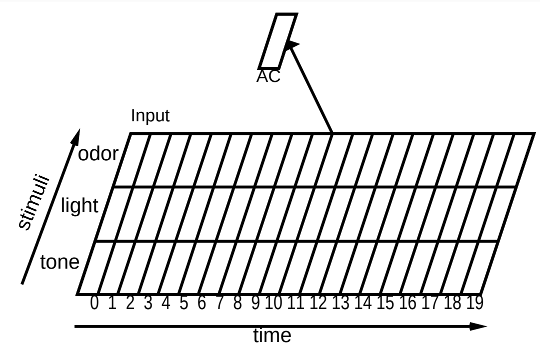
Plus phase: AC settles via weights = expected reward at t+1 (or r).

Minus phase: AC clamped to previous plus phase value (0 at start).

Learning goes "backwards in time" to affect previous time step..

28

Exploration



CSC input (Complete Serial Compound): a unique unit for each stimulus at each time point.

Not realistic, but good for demonstration.