7 Neural Networks (1. Hopfield network)

7.1 Introduction to Hopfield Network and Discrete Hopfield Network

Hebbian Learning having Associative memory with characteristics

- Binary Model
- Update Equations
- Convergence & Error function
- Hebbian Learning / Outer product rule

Concepts:

Attractor as a memory
Emergent behavior – order in randomness

Applications:

1) Associative memory/Content Addressable Memory (CAM)
   - Capacity results
     - Role of p/N ratio
   - Spurious memories
2) Quadratic Optimization
   -Traveling Salesman
   - Graph bipartitioning

Difficulties:

- Low capacity
- Spurious minima

Extensions:

- Bidirectional Associative Memory
- Continuous model
- Continuous model with Adaptive Hebbian learning

7.1.1 Introduction:

The network we have encountered so far are known as feedforward networks since they have no loops in them. In such networks, the relationship between the input and the output can be expressed as a static function. But loops are more common in the brain.

1) A cortical area A that projects to another cortical area B, very often receives a feedback projection from B, forming a loop. Such connections are called reentrant connections.
2) Or a cortical area can project in a series of subcortical nuclei, ultimately received feedback from a nucleus at the end of a long chain, forming a loop.
3) Neurons in cortex and deep brain nuclei typically have local, lateral connections among themselves forming loops at microscopic level.

Thus loops are more common in the brain than cascades of neural layers connected in feedforward fashion.

Hopfield network is a simple neural network model that has feedback connections. It significance lies in the fact that it was able to bring together ideas from neurobiology and psychology and present a model of human memory, known as an associative memory.

The concept of an associative memory can be best explained by contrasting it with computer memory, which is known as indexed memory.

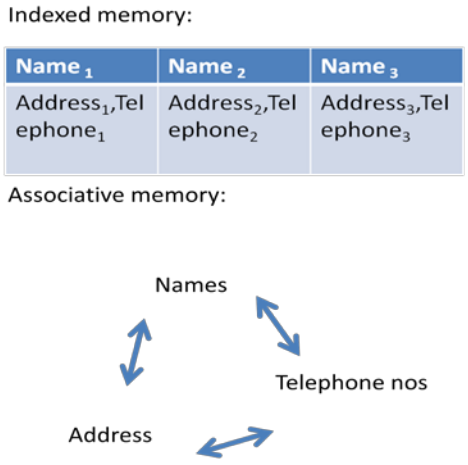


Figure 7.1.1.1: Indexed memory and Associative memory

Indexed memory consists of two columns of data. The first column refers to addresses and the second to contents. If you know the address you can fetch the content located at that address.

A simple example of an indexed memory from the real world is a traditional telephone directory. The "addresses" are the names of customers. Contents are a combination of physical address, telephone number and other details of the customers.
Such a directory has certain disadvantages. It is not possible to fetch the name with the help of a telephone number, or fetch address using telephone number.

But in an associative memory there is no separate "address" and "content." We store memories as networks of items. Each item acts as a cue to retrieve other items, which in turn can cue other items. A given item can act as an "address" or "content" depending on the context.

A telephone directory organized as an associative memory becomes immediately more usable. Instead of splitting a record into "address" (customer's name) and "content" (rest of the record), we can consider splitting it into three fields.

Customer's name ← → physical address ← → telephone number

Any of these fields can uniquely identify the remaining two. Storage and retrieval from such a memory must also be conducted on very different lines from an indexed memory. The Hopfield

network is essentially a neural network realization of an associative memory.
It is possible to have non-neural network-based based associative memories. See Kanerva's memories for example. But they are not the relevant for the present discussion.

7.1.2 Discrete Hopfield Network:

Hopfield has proposed two basic models of associative memories (Hopfield 1982, 1984).
The first of these is a 'discrete model' while the second is a 'continuous' version of the same. The terms 'discrete' or 'continuous' refer to the nature of the state variables and time, in these models.

In the discrete Hopfield network, each neuron has a binary state $V_i \in \{1, -1\}$
The state of the network with N neurons is represented by the vector,

$$V = [V_1, ..., V_i, ...V_N]^T \qquad (7.1.2.1)$$

The network is fully-connected, i.e., each neuron connected to all others.
The weight from j'th neuron to i'th neuron is given by, w_{ij} and weight matrix is given as

$$W = \{w_{ij}\} \qquad (7.1.2.2)$$

Since the network has loops, computations are dynamic and the network state evolves through time, which is a discrete variable.

At any time step,'t', each neuron receives inputs form all other neurons, and updates its own state.
The next state of i'th neuron is expressed as a function of current state of the network as follows:

Update Equations:

$$V_i(t+1) = \sigma(\sum_{j=1}^{N} w_{ij} V_j(t)) \qquad (7.1.2.3)$$

Where σ() is the sign function, sign().

Convergence and Energy Function:

To prove convergence we first introduce a certain Energy function:

$$E = -1/2(\sum_{i=1}^{N} \sum_{j=1}^{N} w_{ij} V_i V_j) \qquad (7.1.2.4)$$

- E is bounded since V is bounded.

- Assume we are updating V_i, calculate the change in E. Consider only the terms in E

  which contain Vi.

$$\Delta E = -(\Delta V_i)(\sum_{j\neq i}^{N} w_{ij}V_j) = (2V_i')(\sum_{j\neq i}^{N} w_{ij}V_j) = (2V_i')(\sum_{j=1}^{N} w_{ij}V_j) - 2w_{ii} \qquad (7.1.2.5)$$

If delta(V_i) < 0, (1 → -1): the sum is also –ve.
If delta(V_i) > 0, (-1 → 1): the sum is also +ve.
Therefore, delta(E) is always –ve or 0.
But since E is bounded the updating process in eqn. (7.1.2.3) should eventually converge.
We have used symmetry of w in the above derivation.

We now discuss the notion of attractors and the possibility of using attractors for storing memories.

Hebbian Learning Rule:

Suppose we want to store S = {Si} in a network. For that E should be minimum at S. So we chose,

$$E = -1/(2N)(\sum_i S_i V_i)^2 \qquad (7.1.2.6)$$

E is minimum at S = V.
Putting E in the standard form given above:

$$E = -1/(2N)(\sum_i S_i V_i)(\sum_j S_j V_j) = -(1/2)\sum_{ij}(\frac{1}{N} S_i S_j)V_i V_j \qquad (7.1.2.7)$$

So the weight matrix required to store a pattern S is:

$W_{ij} = 1/N\ (S_i\ S_j)$. (7.1.2.8)

To store multiple patterns, S^p the rule is:

$$w_{ij} = (1/N)\sum_p S_i^p S_j^p \qquad (7.1.2.9)$$

The last rule for pattern storage is called the 'Hebb's rule.'

Hebbian Learning:

According to Hebbian learning, when two neurons are simultaneously active, the connection between them must be strengthened; when one of them is active, while the other is inactive, the connection strength must be weakened. Interestingly, the learning rule given above (eqn. (7.1.2.8)), though derived from basic mathematical considerations, reflects the mechanism anticipated in (Hebb 1949) which was stated as follows:

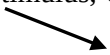
"When an axon of a cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased."
(From the 'Organization of Behavior' by Donald Hebb.)

Examples:

1) Long term Potentiation (LTP):
In the neurons of Hippocampal CA1 area, simultaneous presynaptic transmitter release and postsynaptic depolarisation led to long lasting association between them.

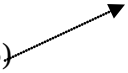
Ref: Wigström H, Gustafsson B (1986). "Postsynaptic control of hippocampal long-term potentiation". J. Physiol. (Paris) 81 (4): 228–36.

2) Classical Conditioning:
Pavlov on his research on classical conditioning in dogs noticed the following:
Presenting Food (Unconditional Stimulus, US)

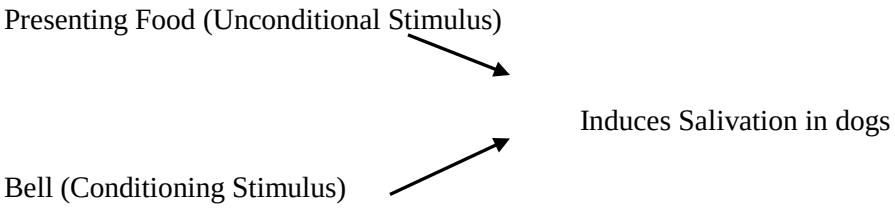


Induces Salivation in dogs (Unconditional Response)

Bell (Conditioning Stimulus, CS)



On repeated ringing the bell, just before presenting the food, it was found that the dogs salivated simply in responses to ringing the bell (A Conditional Response due to Conditional Stimulus), even before the food was presented.



The above effect can be explained using Hebbian learning.

Spurious States:

A problem arises when we try to store multiple patterns using Hebb's rule of eqn. (7.1.2.8). In addition to the stored states, certain other states known as spurious states, also end up becoming stable, interfering with the network operation. Therefore, when the network is prompted with a corrupt version of a stored pattern, the network might retrieve a pattern which is not one of the stored patterns, thereby diminishing the network's retrieval performance.

There are three types of spurious patterns:

1) Mirror states: It can be shown that if S is a stable state, -S is also a stable state.
   Proof: If S is stable, we know that
   S = g(WS)
   If g is the sign() function, we have –x = g(-x), if x = $\pm$1. Therefore, we have,
   -S = g(-WS).

2) Mixture states: S_{mix} = sign(S1 + S2 + S3)
   Proof: S_{mix} and S_p (p = 1,2,3) have the same sign on an average 3 times out of 4.
   Hence,

$$\sum_i S_i^p S_i^{mix} = N/2 \qquad (7.1.2.10)$$

   Therefore,

If the cross-terms can be ignored, the mixture state is stable.

$$h_i^{mix} = 1/N \sum_{jp} S_i^p S_j^p S_j^{mix} = S_i^1/2 + S_i^2/2 + S_i^3/2 + cross-terms \qquad (7.1.2.11)$$

3) Spin glass states: Stable states that are not correlated with the stored patterns.

Hopfield Networks and Spin Glass Systems:

Hopfield neural network model was originally inspired by the physics of magnetic materials or spin glass systems.

A magnetic material consists of a set of atomic magnets arranged on a crystal lattice.
Each atomic magnetic has a spin associated with it. Spin = ± 1
Spin of an atomic magnet is influenced by the magnetic field, h, at its location.
Field = external field (h_{ext}) + internal field (produced by other spins)

$$h_i = \sum_j w_{ij} S_j + h_{ext}$$

(7.1.2.12)

W_{ij} = (exchange interaction strength) measure of strength of interaction of influence of spin S_j on the field at S_i.
Interactions are symmetric. $W_{ij} = W_{ji}$.
Potential energy corresponding to the interaction is:

$$H = -\frac{1}{2}\sum_j w_{ij} S_i S_j - h_{ext}\sum_i S_i$$

(7.1.2.13)

Stability of a pattern & Error Analysis:

Let us calculate the probability that a particular bit in a stored pattern is stable. Consider,

$$W.S^1 = (1/N)\sum_p S^p S^{pT} S^1 = S^1 + (1/N)\sum_{p\neq 1} S^p (S^p \bullet S^1)$$

(7.1.2.14)

The i'th bit of the above expression is,

$$S_i^{\,1} + (1/N)\sum_{p\neq 1} S_i^{\,p} (S^p \bullet S^1)$$

(7.1.2.15)

For $S_i^{\,1}$ to be stable, the second term above should be less than 1 in magnitude. Or the following term, C_i, should be,

$$C_i = -(1/N) S_i^{\,1} \sum_{p\neq 1} S_i^{\,p} (S^p \bullet S^1) < 1$$

(7.1.2.16)

Or the bit becomes unstable when $C_i > 1$. We now calculate the probability of $C_i > 1$.
Note that Ci is a sum of r = (P-1)N binary numbers (with a factor of 1/N). For a single bit,

Mean = 0,
Variance = $1/(N^2)$

For the sum,
Mean = 0,
Variance = $(P-1)N/N^2 = (P-1)/N \approx P/N$.

If N is large, from Central Limit Theorem we know that C_i has a gaussian distribution with the above mean and variance.

$$P(C_i > 1) = 1/(\sqrt{2\,pi}\ sigma)\int_1^{\infty} \exp(-x^2 / 2sig\text{^}2)dx = (1/2)[1 - erf(1/\sqrt{2sig^2})]$$

$$= (1/2)[1 - erf(\sqrt{N/2P})]$$

$P(C_i>1)$	P/N
0.001	0.105
0.0036	0.138
0.01	0.185
0.05	0.37
0.1	0.61

Avalanche phenomenon occurs at P/N = 0.138. That is, for P/N > 0.138 error increases rapidly. Therefore P/N = 0.138 is considered to be the capacity of the Hopfield Network.

Hopfield got P/N = 0.15 through experiments.

Improving Capacity:

1. The self-weights must be set to zero for improved retrieval performance.
   $W_{ii} = 0$
   To understand this, consider, $w_{ii} >> 1$,
   $V_i(t+1) \approx g(w_{ii}\ V_i(t)) = V_i(t)$. That is every state is stable. Therefore, there is a greater chance of obtaining spurious states.

7.2 Continuous Hopfield Network

V(t+1) ← V(t) - discrete model
dV/dt = f(V) - continuous model

Equations:

$$\frac{du_i}{dt} = -u_i + \sum_{j=1}^{N} w_{ij} V_j + I_i \qquad (7.2.1)$$
$$V_i = g(\lambda u_i) = tanh(\lambda u_i)$$

Lyapunov or "Energy" Function:

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij} V_i V_j + \sum_{i=1}^{N}\int_0^{V_i} g^{-1}(V)dV - \sum_{i=1}^{N} I_i V_i \qquad (7.2.2)$$

Proof that (7.2.2) is a Lyapunov function of (7.2.1):

Consider,

$$\frac{dE}{dt} = -\sum_{i=1}^{N}(\sum_{j=1}^{N} w_{ij} V_j - u_i + I_i)\frac{dV_i}{dt}$$
$$= -\sum_{i=1}^{N}\frac{du_i}{dt}\frac{dV_i}{dt} = -\sum_{i=1}^{N} g'(u_i)(\frac{du_i}{dt})^2 \le 0$$

Therefore, E decreases monotonically with time.

Single neuron: Condition for stability:
Dynamics of a single neuron in the continuous Hopfield network may be given as:

$$\frac{du}{dt} = -u + wg(\lambda u) \qquad (7.2.3)$$

assuming that the external input I_i is zero.

At convergence we have,
du/dt = 0, or

$$u = w \tanh(\lambda u). \qquad (7.2.4)$$

Solution to the last equation may be represented as intersection of two plots,

$y = u,$ (7.2.5)

and

$y = w \tanh(\lambda u),$ (7.2.6)

which is graphically represented below. The number of intersection points obtained depends on the value of lam. For $\lambda > 1$, we have three intersection points (Fig. 7.2.1), and for $\lambda < 0$, we have only one intersection point at the origin (Fig. 7.2.2)

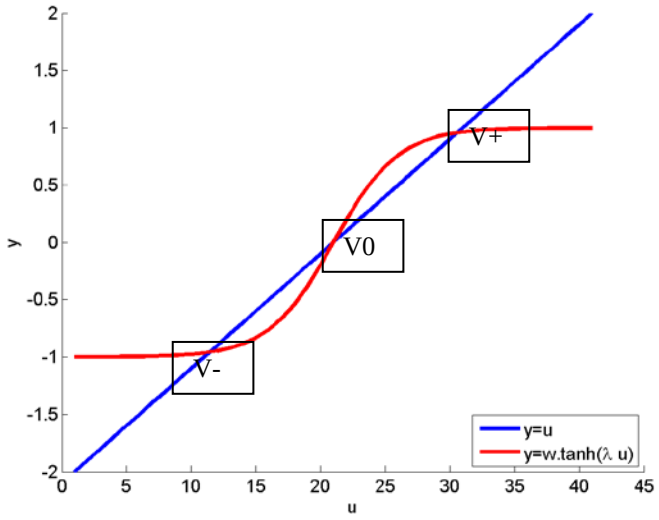


Figure 7.2.1: Stationary points for $\lambda = 2$

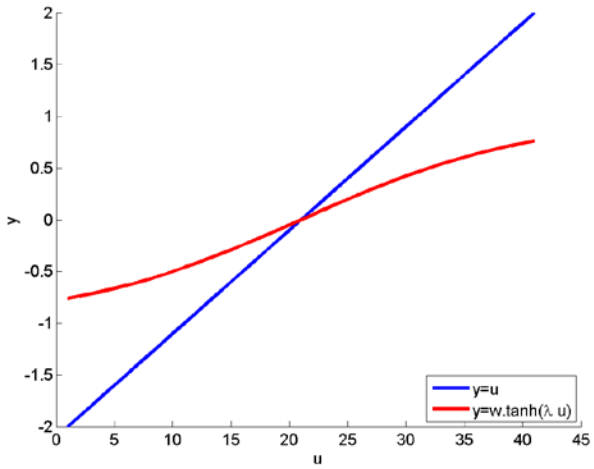


Figure 7.2.2: Stationary points for $\lambda = 0.5$

Let us examine the stability of the intersection points in the two cases.

For $\lambda > 1$:

Let us call the three intersection points V+, V-, V0 (origin), which are also the stationary points of Eqn. (7.2.4). A stationary point is stable, if

$$\frac{d^2u}{dt^2} < 0 \; ; \; \frac{d^2u}{dt^2} = -1 + w\lambda g'(\lambda u)$$

The two terms on the right side are the slopes of the eqns. (7.2.5, 7.2.6) respectively. Therefore, $\frac{d^2u}{dt^2} < 0$ when the slope of the sigmoid function in Fig. 7.2.1 is less than 1, which is true for both V+, V_ but not true at the origin. Therefore, V+ and V_ are stable but the origin is unstable.

For $\lambda < 1$:

In this case, there is only one intersection point at the origin (Fig.7.2.2). Since the slope of the sigmoid is less than 1 at the origin, the origin is stable.

A two-neuron system:

Let us consider a simple two-neuron system with weight matrix
W = [1 -1
-1 1].
Eqn. (7.2.1) can be expanded in this case (assuming that the external input I_i is zero) as,

$$\frac{du_1}{dt} = -u_1 + w_{11}V_1 + w_{12}V_2$$

$$\frac{du_2}{dt} = -u_2 + w_{21}V_1 + w_{22}V_2$$

$V_1 = \tanh(\lambda u_1)$ and $V_2 = \tanh(\lambda u_2)$, $\lambda = 2$.

Fig. 7.2.3 below depicts the vector field defined by the above system.

There are stable nodes at (-2,2) and (2,-2) and the origin is a saddle node.

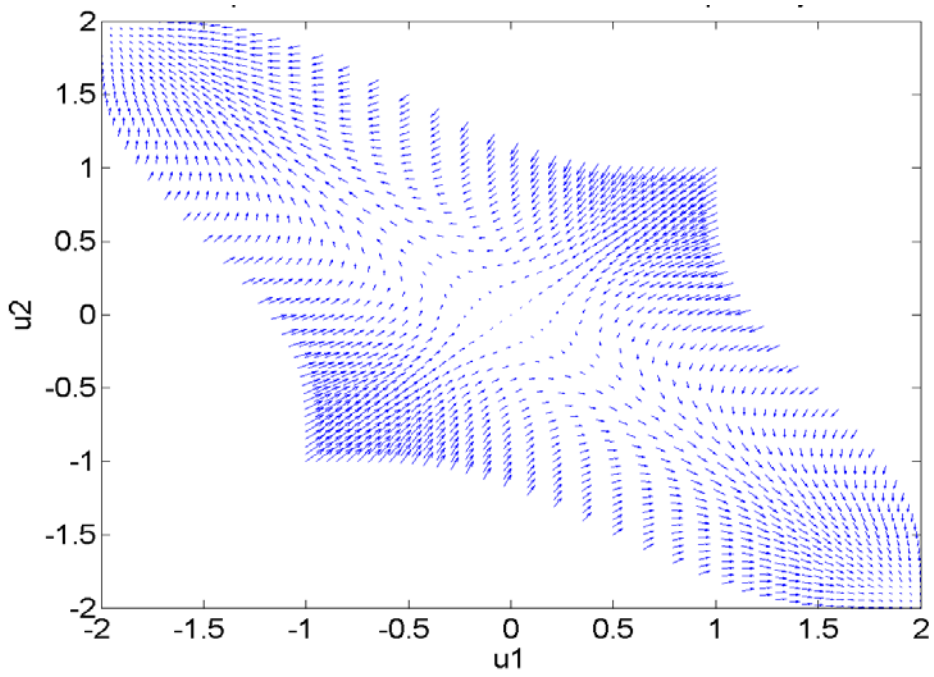


Figure 7.2.3: Arrow plot of the two neuron continuous hopfield system

7.3 The Adaptive Hebbian Learning in Hopfield Network:

In both discrete and continuous Hopfield network weights trained in a one-shot fashion and not trained incrementally as was done in case of Perceptron and MLP. There is a variation of Hopfield network in which weights are adjusted incrementally. Dynamics of the Hopfield network with adaptive weights is described below.

$$\tau_a \frac{du_i}{dt} = -u_i + \sum_{j=1}^{N} w_{ij} V_j + I_i \qquad \text{(activation dynamics)} \qquad (7.3.1)$$

$$\tau_w \frac{dw_{ij}}{dt} = -w_{ij} + (V_i V_j) \qquad \text{(Weight dynamics)} \qquad (7.3.2)$$

where ($\tau_a << \tau_w$) and

$V_i = \tanh(\lambda u_i)$, $\lambda >> 1$.

The combined system has an expanded "energy" function given as,

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij} V_i V_j + \sum_{i=1}^{N}\int_0^{V_i} g^{-1}(V)dV - \sum_{i=1}^{N} I_i V_i + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}^2 \qquad (7.3.3)$$

Note that when the weights stabilize,

$\frac{dw_{ij}}{dt} = 0$, or

$$w_{ij} = (V_i\ V_j) \qquad (7.3.4)$$

Which is similar to the Hebb's rule of the earlier Hopfield models.

But the rule,

$w_{ij} = (V_i\ V_j)$

would amount to storing the network state in the weights. But that is not necessarily desirable because if the network is in a random state, then eqns. (7.3.1,7.3.2) simply tend to render the present state of the network persist. What is needed is a mechanism that stores the external input I_i in the weights by Hebb's rule. That can be assured indirectly if the external input imposes itself on the network state V, which in turn can be written onto the network weights w.

I → V → w

To ensure such a steady transfer of information from the external input to the network weights, we need certain additional mechanisms. To this end, we add a few coefficients (A, B and C) to the eqns. (7.3.1,7.3.2) as follows:

$$\tau_a \frac{du_i}{dt} = -u_i + B(\sum_{j=1}^{N} w_{ij} V_j) + AI_i \qquad \text{(activation dynamics)} \qquad (7.3.5)$$

$$\tau_w \frac{dw_{ij}}{dt} = C(-w_{ij} + (V_i V_j)) \qquad \text{(Weight dynamics)} \qquad (7.3.6)$$

We divide the operation of the network into separate 1) storage and 2) retrieval stages.

1) **Storage dynamics:**
   During storage the pattern to be stored is presented as external input I. The network state, V, must copy I. The network state V is then written onto the weights by eqn. (7.3.6).

   For this to be possible, in eqn. (7.3.5), 'A' must be a large positive number. Similarly 'B' must be a small positive number so that it does not allow the preexisting weights to influence the current state of the network.
   During storage, obviously the weights have to be adapted. Therefore, 'C' is must be a finite positive number.

2. Retrieval dynamics:

Retrieval dynamics must be obviously dominated by the preexisting weights; therefore B should be large positive number. The external input should only be suggestive of the pattern that needs to be retrieved, and therefore need not be very large. Therefore 'A' must be a small number. During retrieval the weights are in principle not adjusted; therefore 'C' is zero.

Summary:
The Continuous Hopfield network with adaptive weights must have its coefficients A, B, C to be set as follows:

1. Storage stage: high A, low B, and high C.
2. Retrieval stage: low A, high B and zero C.

We will now see that mechanism of the type described above can be seen in Hippocampus, a part of the brain involved in memory.

7.4 Hippocampus – memory storage and recall:

We have seen in Chapter 1, which deals with the nervous system anatomy and organisation, that hippocampus (HC) is a "scratch pad" of memories. Damage to HC is known to impair our ability to store and retrieve memories. Earliest knowledge about the memory-related functions of HC came out of studies of a patient known as Henry Molaison, usually referred to as HM. HM had his medial temporal lobe removed as a treatment to his epileptic attacks, with which he was suffering since his childhood. Loss of HC which is located in the temporal lobe, cause memory impairment or amnesia. He suffered from two kinds of amnesias:

Retrograde amnesia: inability to recall old memories. Had poor memory of events that happened immediately before his surgery, but older memories were intact.

Anterograde amnesia: inability to store new memories.

Hebbian Learning and Long-term Potentiation (LTP) in hippocampus

We have seen that synaptic modification is the key substrate to learning and memory phenomena in the brain. Particularly we have seen how Hebbian learning can enable storage of information as attractors in network dynamics. Although when Hebbian learning was proposed in late '40s, it was only a hypothesis, experimental evidence for the same was first discovered in the synapses of HC in the '60s.

These studies have essentially shown that when both pre- and post-synaptic ends are simultaneously stimulated, the synapse gets strengthened or potentiated. This phenomenon is known as Long term Potentiation (LTP). When only one side is stimulated, the synapse gets weakened or depressed, a phenomenon known as Long-term Depression (LTD).
In studies of this kind, typically, the presynaptic terminal is stimulated by a high-frequency volley of impulses, which amounts activation of the presynaptic side. On the post-synaptic side the membrane is depolarized by injecting a positive current. Initial strength of the synapse is assessed by stimulating the presyanptic side with a low frequency volley of impulses, and amplitude of the resulting PSP is noted. After stimulating the pre- and post-synaptic sides simultaneously, as described above, for about 10-15 mins, the synaptic strength is again assessed by stimulating the pre-synaptic side with a low frequency volley. It is seen that the PSP now is greater than the PSP observed before stimulation (Fig. 7.4.1).

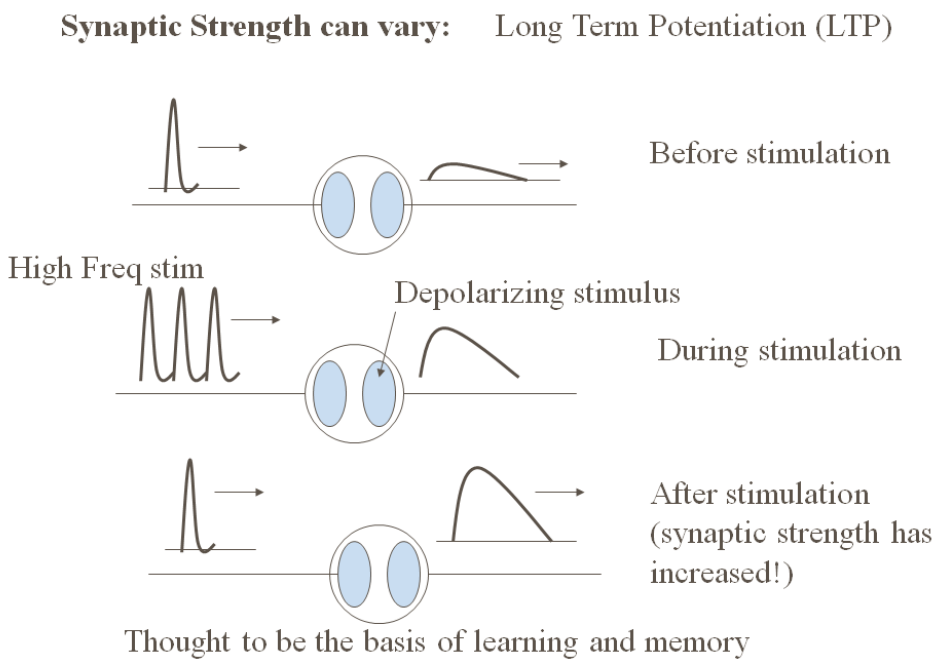


Figure 7.4.1: A schematic of Long-term Potentiation (LTP). Top figure shows a weak EPSP in response to the single Action Potential. Middle figure shows the synapse stimulated by a high frequency volley of APs on the pre-synaptic side, and a depolarizing stimulation on the post-synaptic side. The bottom figure shows that the response of the synapse to a single AP is stronger than before.

LTP, which is a biological substrate of Hebbian learning, has been found to occur in many hippocampal synapses. But for pattern storage we require Hebbian learning occurring in a network with strong recurrent connections. Does HC have strongly recurrent connectivity?

To answer this question, let us consider the anatomical connectivity patterns within the HC and also its connections to other parts of the brain, particularly the cortex.

Cortex and the Hippocampus connections:

HC is basically a system for association. It captures relationships among representations of the same object present in different sensory modalities. For example, when you hold a coffee cup in your hand, your visual system processes the appearance of the cup, - its size, color, distance from your eyes etc. Your somatosensory system processes the feel of the cup, its hardness or softness, its temperature etc. Both these streams of sensory information contain information about the cup. This high level information about the cup is extracted at higher stages of sensory cortices and forwarded to HC. Thus HC receives compact high-level representations of the object. HC also receives reward related information from prefrontal cortex and subcortical structures like amygdala. HC combines the compact representation of the object, and the relevant reward information and decides if the information is worth storing for longer-term back in the cortex, or must be discarded.

HC receives inputs from large parts of the neocortex which project to parahippocampal gyrus and perirhinal cortex (Fig. 7.4.2). The last two areas project to Entorhinal cortex which is considered

the gateway to HC. Information from EC propagates in that sequence via Dentate Gyrus, CA3, CA1 and subiculum before returning to EC. EC projects back to the neocortical targets via parahippocampal gyrus and perirhinal cortex. Thus HC receives compressed representation of information arising out of many sensory modalities, and applying the criteria of saliency which depends on the associated reward, sends the information back to the neocortex for long-term storage.

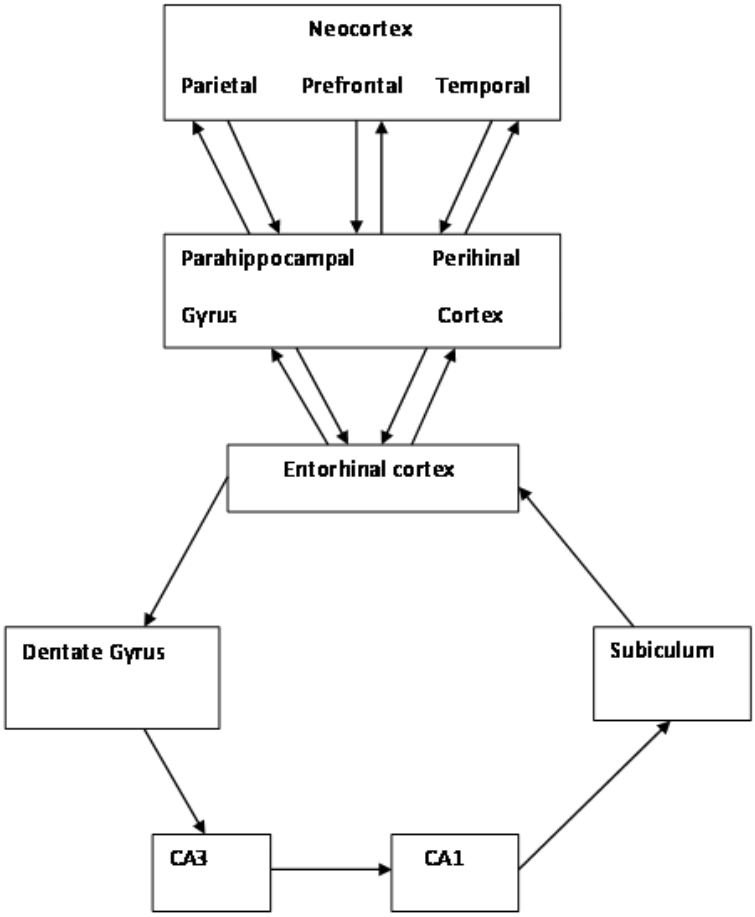


Fig 7.4.2: Inputs received by the hipocampus

The flow of information within the HC described above is rather simplistic. While it is true that information arising out of EC circulates through the HC and returns to EC, the path it follows is not a simple, singular loop as described above. Information from EC flows through multiple parallel pathways before returning to EC (Fig. 7.4.3).

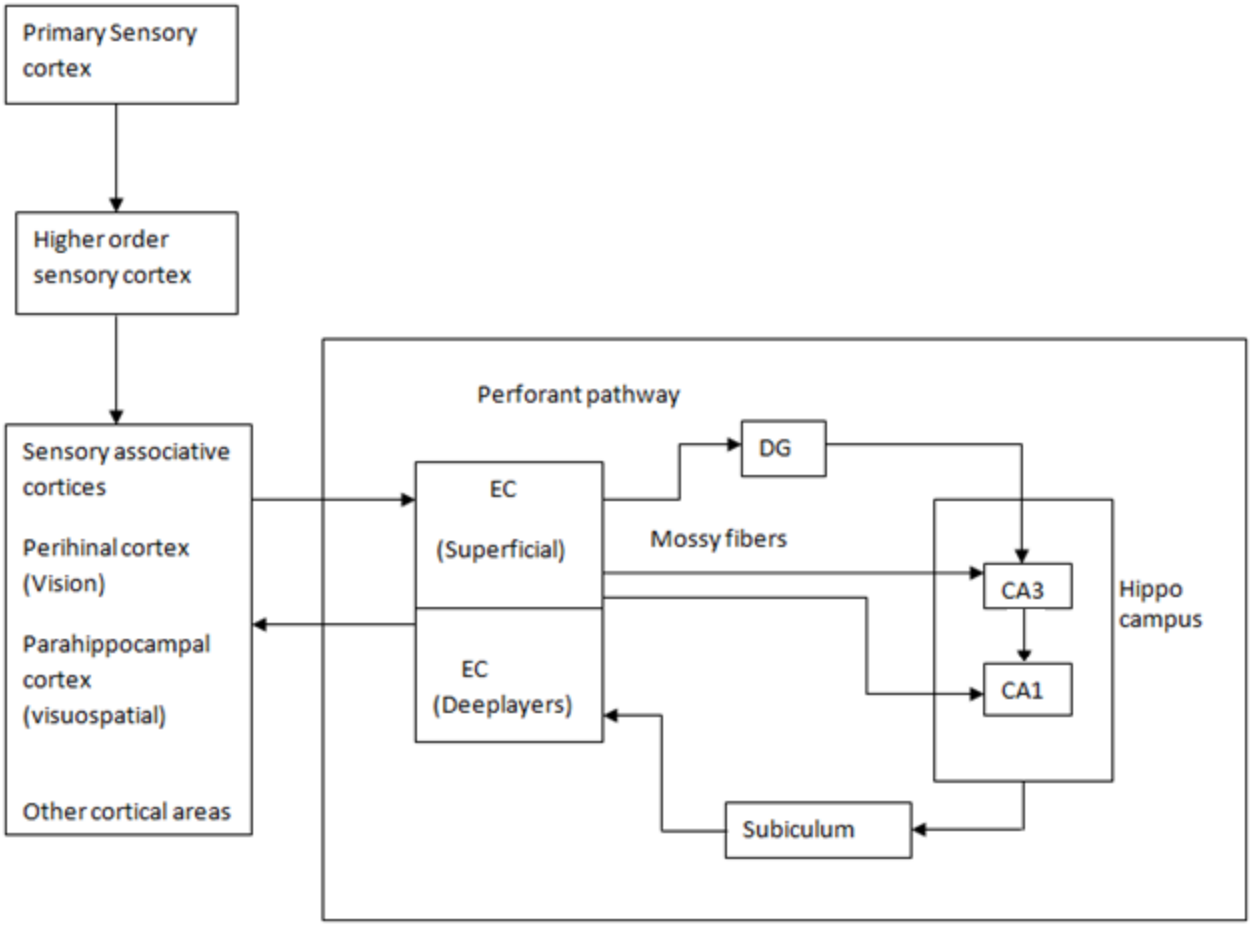


Figure 7.4.3. Multiple pathways of information of EC

One branch of EC output into the HC goes to Dentate Gyrus, which in turn projects to CA3 via fibers known as mossy fibers. EC also projects directly to Dentate gyrus via the perforant pathway. The mossy fibers are known to have strong synapses, while the perforant pathway has a weaker influence on CA3. This difference in the relative strengths of the two afferents to CA3 seems to be significant, as we shall see shortly, to memory storage and recall operations of HC. CA3 neurons have extensive recurrent connections among themselves (Figs. 7.4.4, 7.4.5).

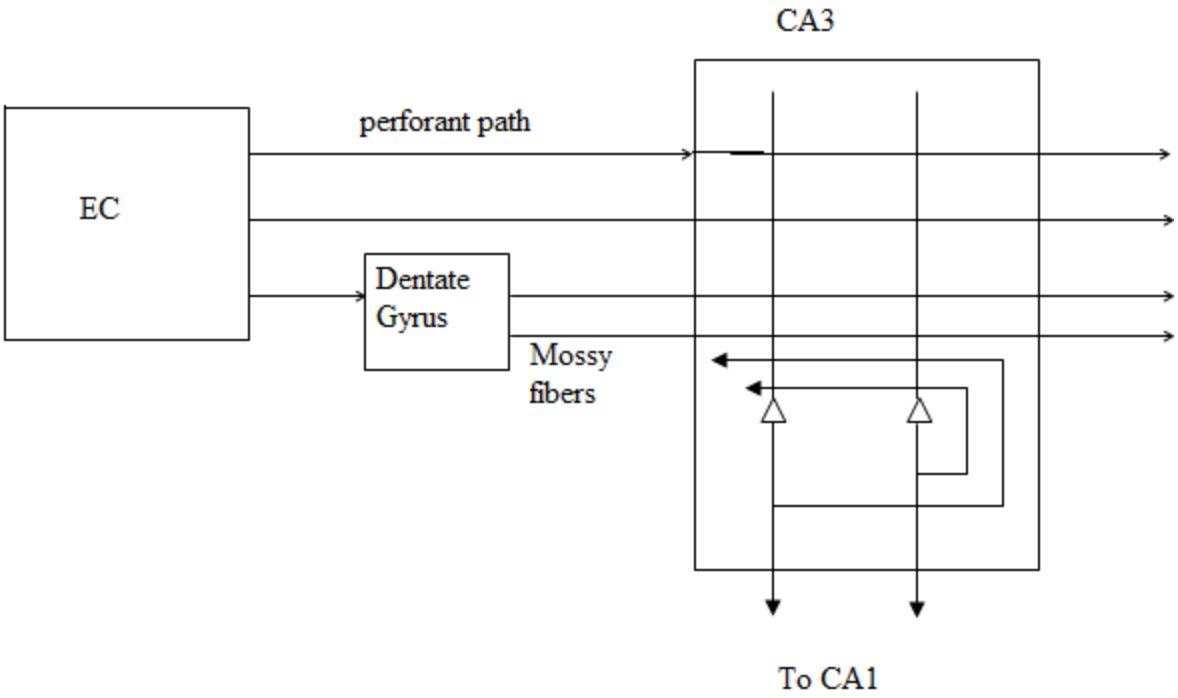


Figure 7.4.4: Figure shows the inputs and outputs of CA3 and also the recurrent connections in CA3.

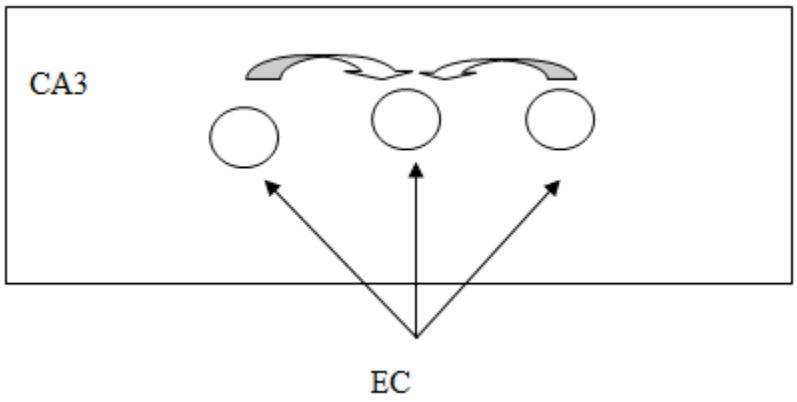


Figure 7.4.5: Recurrent connection in CA3

As mentioned above, CA3 neurons have extensive recurrent connections. In rat there are about 300,000 pyramidal neurons in CA3, with each of them receiving inputs from about 12,000 other CA3 pyramidal neurons. In contrast, each CA3 pyramidal neuron receives only 4000 inputs from EC. Effectively, each CA3 neuron receives inputs from about 4% of all CA3 pyramidals, a level of recurrency that is high compared any other brain region. Furthermore, these recurrent connections are modifiable by LTP. Therefore, though the level of recurrency in CA3 is much lesser than that in the Hopfield network, it suggests the strong possibility of CA3 acting as some sort of a Hopfield network in which patterns are stored as attractors.

We have seen that the memory capacity, P, of a fully-connected Hopfield network, in which the binary state (1/0) of each neuron is 1 or 0 with equal probability, equals 0.14 N, where N is the number of neurons in the network. But it has been found that capacity is greater when sparseness conditions are imposed on the stored patterns. A pattern is considered to be sparse when instead of half the neurons being active (+1), only a small fraction, a, of neurons are active. There is evidence that only a small fraction of CA3 neurons are active at any given instant (evidence regarding this matter is reviewed in (Treves and Rolls, 1994)).

Storage capacity of CA3 system in rat, which depends on sparseness, a, is estimated by Treves and Rolls (1994). Sparseness is defined as:

$$a = \frac{(E[r])^2}{E[r^2]} \qquad (7.4.1)$$

where 'r' refers to activity level of neurons in CA3.

Storage capacity, P, is estimated as,

$$P = \frac{C^{RC} k}{a \ln(1/a)} \qquad (7.4.2)$$

Where,

C^{RC} - is the number of recurrent connections per CA3 neuron. (C^{RC} =12,000 in rat)
k – is a parameter that depends on the neural activity distribution, connectivity pattern etc (Treves and Rolls 1994)
a – sparseness, which is taken to be 0.02 based on experimental data

Accordingly, P is calculated to be 36,000 patterns.

Storage and Retrieval in CA3:

We have considered evidence that CA3 has features of an autoassociative network – high recurrence, sparseness and plastic synapses. But these features in themselves are not sufficient, since for the network to be useful as a memory, it must be operable in two modes – storage and retrieval. Very different conditions must prevail in the network when it operates in these two modes.

Recall the two modes of operation of the Adaptive Hopfield model during storage and retrieval.
During storage:

- External input is strong
- Recurrent connections are weak
- Recurrent connections are plastic

During retrieval:

- External input is weak
- Recurrent connections are strong
- Recurrent connections are not plastic

Let us consider if the above features are available in and around CA3.

1. CA3 has two external inputs: one from EC by perforant pathway, and the other from EC via Dentate Gyrus by mossy fibers. There is evidence that perforant pathway inputs are weaker than mossy fibers.
2. Strength and plasticity of CA3 recurrent connections: Acetylcholine (Ach) is a neuromodulator that is capable of controlling the strength and plasticity of recurrent connections in CA3.

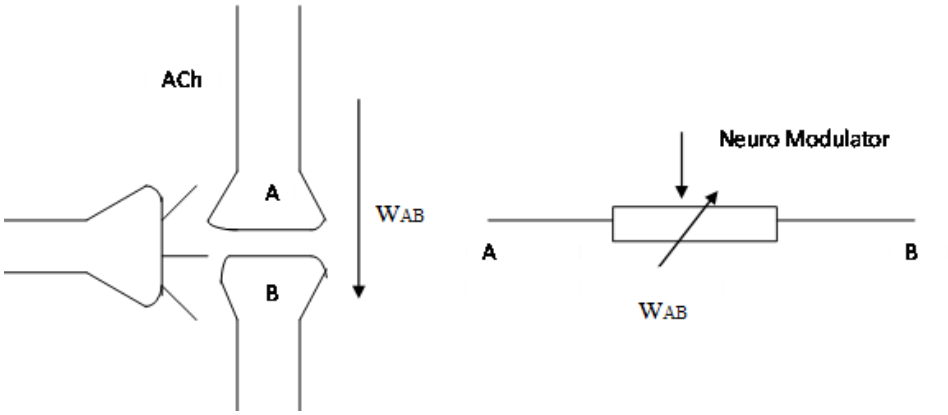


Figure 7.4.6: Ach is a neuromodulator that controls strength and plasticity of synapses

A neuromodulator alters the strength of a synapse. Some common neuromodulators in the brain are:
Acetylcholine, dopamine, norepinephrine and serotonin.

Blockage of ACh transmission in HC is known to interfere with memory storage.
Scopolamine (Ach antagonist) → temporary amnesia

Volunteers were given a dose of scopolamine and were asked to memorize some information.
They had little or no memory of the information presented during the test
Some volunteers even forgot about the trial.

Scopolamine before experiment:
Controls = 45 out of 128 words
Scopolamine subjects = 6 out of 128 words
When subjects were given scopolamine after the words were presented, there no impairment in performance.

Specifically, there is also evidence that Ach controls LTP in the recurrent connections of CA3.

Michael Hasselmo and coworkers have suggested that Ach satisfies the criteria necessary to switch between storage and retrieval as follows:
At high levels of Ach,

- Input is strengthened
- Recurrent connections of CA3 are weakened
- Increased plasticity in recurrent connections of CA3

At low levels of Ach,

- Input is weakened
- Recurrent connections of CA3 are strengthened
- Decreased plasticity in recurrent connections of CA3

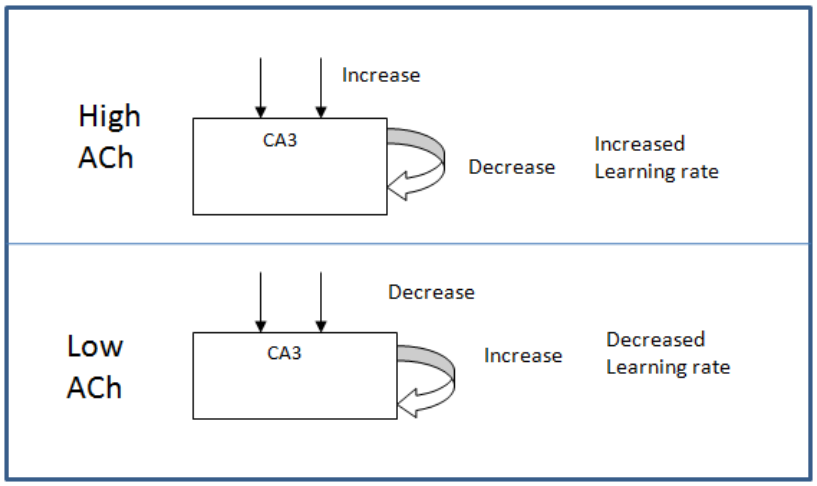


Figure 7.4.7: ACh level changes affect the learning rate

Such complex differential action of Ach is possible because, Ach has different actions on different parts of the neuron. Furthermore, inputs to CA3 pyramidal neurons from different sources are anatomically segregated in CA3.
Inputs from EC to CA3 pyramidal neurons are located in a CA3 layer known as stratum lacunosum-moleculare.
Recurrent inputs from other CA3 pyramidal neurons are located in a CA3 layer known as stratum radiatum.

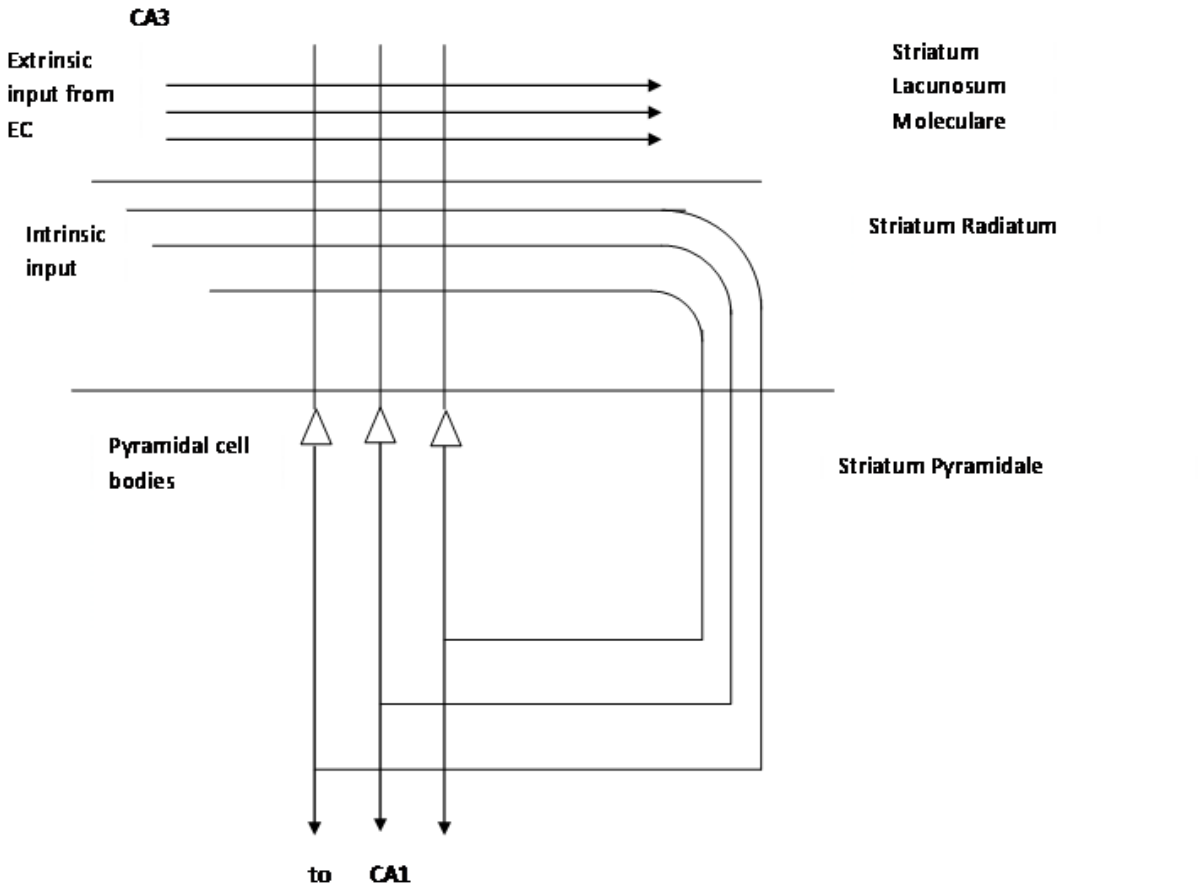


Fig 7.4.8: CA3 pyramidal neurons from different sources are anatomically segregated in CA3.

When ACh or an agonist of ACh is applied to CA3, it suppresses the synapses in stratum radiatum much more than it does to synapses in stratum lacunosum-moleculare.
Thus at higher concentrations ACh suppresses recurrent connections more than inputs coming from EC. At the same time increased ACh increases LTP in CA3 recurrent connections.

What controls ACh release to HC?

ACh is released by two neuronal clusters located in the basal forebrain (Fig. 7.4.9). One cluster called the nucleus basalis, projects ACh to widespread cortical targets. The other cluster called medial septum projects to HC. There is also a branch from HC to medial septum, known as the hippocampal-septal pathway. When this branch is stimulated it decreases the neural activity in medial septum. Thus, the loop from medial septum to HC and backwards acts as a self-regulating system for HC. Particularly it is known that the feedback exists from CA1 field of HC.

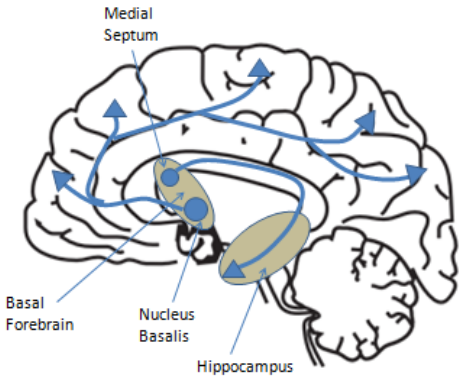


Figure 7.4.9: Neuronal clusters located in the basal forebrain

Hasselmo and colleagues proposed that the above loop is capable of making sure that ACh is released just when it is required. Basically ACh must be released when the input to HC, arriving at EC, needs to be stored. If the input is already stored, ACh release must be blocked.

Note from Figs. 7.4.2-7.4.4 above that CA1 receives direct inputs from EC and indirectly via CA3. Hasselmo and coworkers proposed that CA1 acts as a comparator, that compares the original pattern in EC, and its reconstructed version from CA3. If the two copies are the same, the pattern must have already been stored in CA3. Then CA1 sends a signal to medial septum inhibiting it, and blocking further release of ACh.
If the two copies are not the same, CA1 inhibition of medial septum ceases, thereby causing of release of ACh to CA3, which in turn induces pattern storage.

Reference:

M. Gluck and C. Meyers, Gateway to Memory.