Learning via Synaptic Tuning

June 26, 2014

1 Hebbian Origins

Neural networks are often viewed as functions or mappings from activation states of input neurons to those of output neurons, with the intelligence behind this transformation distributed over the myriad connection weights. Comprehensive logical descriptions of these functions are often extremely difficult to formulate. However, the learning process through which these mappings emerge can follow very simple, intuitive, local rules for synaptic modification – many inspired by that classic principle of Hebb (1949): fire together, wire together.

These local rules appear to underlie a good deal of biological learning and are sufficient to produce sophisticated input-output relationships in both natural and artificial neural networks. From their simplicity arises complex knowledge representations. This chapter investigates some of these local learning schemes and how they support various search processes that lead to the emergence of complex global network patterns and, in general, intelligent behavior.

First, however, we consider the most classic and widespread form of ANN adaptivity: supervised learning via the backpropagation algorithm. Though it embodies little emergence, and even less biological realism, it employs a local search method that enables ANNs to learn a wide range of complex mappings, thus making thousands of practical contributions across numerous disciplines.

2 Supervised Learning in Artificial Neural Networks

Supervised Learning scenarios arise reasonably seldom in real life. They are situations in which an agent (e.g. human) performs many similar tasks, or many similar steps of a longer task, and receives frequent and detailed feedback: after each step, the agent learns a) whether or not her action was correct, and, in addition, b) the action that should have been performed. In real life, this type of feedback is not only rare, but annoying. Few people wish to have their every move corrected.

In Machine Learning (ML), however, this learning paradigm has generated considerable research interest for decades. The classic tasks involve classification, wherein the system is given a large data set, with each case consisting of many features plus its characterization or class. The system uses these examples of feature-to-class associations to construct a general-purpose mapping from feature vectors to hypothesized classes.
For example, the features might be meteorological factors such as temperature, humidity, wind velocity, etc., while the class could be the predicted precipitation level (high, medium, low, or zero) for the next day. The supervised-learning classifier system must then learn to predict future precipitation when given a vector of meteorological features. The system receives the input features for a case and produces a precipitation prediction, whose accuracy can be assessed by a comparison to the known precipitation attached to that case. Any discrepancy between the two can then serve as an error term for modifying the system (so as to make better predictions in the future). Classification tasks are a staple of Machine Learning, and feed-forward ANNs trained by backpropagation are one of several standard tools for tackling them. Since its conception in the mid 1970’s (Werbos, 1974) and formalization and popularization a decade later (Rumelhart et al., 1986), backpropagation has accounted for a large percentage of all successful ANN applications.

The basic idea behind backpropagation learning is to gradually adjust the ANN’s weights so as to reduce the error between the actual and desired outputs on a series of training cases. The presence of these desired outputs demarcates the task as supervised, while the strategy of searching in the direction of decreasing error falls under the category of gradient descent search methods.

Each case is typically a pair, \((d_i, r_i)\), indicating an element of the mapping between a domain and a range for some (yet unknown) function. By training the ANN to reduce this error, one effectively discovers the function.

Figure 1 summarizes the basic process, wherein training cases are presented to the ANN, one at a time. The domain value, \(d_i\), of each case is encoded into activation values for the neurons of the input layer (left). These values are then propagated through the network to the output layer (right), whose values are then decoded into a value of the range, \(r^*\), which is compared to the desired range value, \(r_i\). The difference between these two values constitutes the error term.

The challenge in gradient descent is to figure out which combination of weight changes (to the many hundreds or thousands of synapses) will actually reduce the output error. Intuitively, we know that if a change in a weight will increase (decrease) the error, then we want to decrease (increase) that weight. Mathematically, this means that we look at the derivative of the error with respect to each weight: \(\frac{\partial E}{\partial w_{ij}}\), which represents the change in the error given a unit change in the connection weight from neuron \(j\) to neuron \(i\).

Once we find this derivative, we will update \(w_{ij}\) via the following:

\[
\Delta w_{ij} = -\eta \frac{\partial E_i}{\partial w_{ij}}
\]  

This essentially represents the distance times the direction of change. The distance, \(\eta\), is a standard parameter in neural networks and often called the learning rate. In more advanced algorithms, this rate may gradually decrease as training progresses. The above equation is easy, and it captures our basic intuitions: decrease (increase) a weight that positively (negatively) contributes to the error. Although the calculations for \(\frac{\partial E}{\partial w_{ij}}\) are beyond the scope of this book, they are available in most neural network textbooks (Haykin, 1999; Arbib, 2003).

If we update all the weights using the same formula, then this amounts to moving in the direction of steepest descent along the error surface, as shown in Figure 2. This constitutes a form of local search, since the entire weight vector is employed to compute output values, the basis for evaluation. However, unlike many of the local-search methods discussed earlier, backpropagation involves a very intelligent modification of each weight, based on the detailed calculation of \(\frac{\partial E_i}{\partial w_{ij}}\), a process involving information from all over the network - and thus not local. Furthermore, since the desired outputs and the errors that they spawn are easily interpreted as global directives as to the behavior of the system, the training of ANNs via backpropagation...
Figure 1: The essence of the backpropagation algorithm. (Top) During training, cases are encoded and sent through the network. The resulting outputs are compared to desired outputs to compute an error term, $E$, which is then used to compute changes to the weights of the network so as to reduce $E$. (Bottom) Data is initially partitioned into training and test sets, with the former passing through the network many times – and inducing weight change each time – while the test data passes through just once, without learning, to assess the networks general ability to handle data on which it has not been trained.
hardly qualifies as emergent.

Thus, one of the biggest success stories in Bio-AI, the backpropagation algorithm, is only loosely inspired by neuroscience and the emergent nature of intelligence. Still, it performs a sophisticated search in a space of complex mappings, often yielding solutions when other techniques fail. This earns respect not only from engineers and other technical problem solvers, but also from brain researchers, who may see no scientific value in the learning algorithm itself but do appreciate the final result: a proof-of-principle neural network that can perform a particular mapping from sensory inputs to motor outputs. The existence of such a network can help justify a neuroscientist’s claim that, indeed, a collection of neurons (with particular connection strengths between them) can achieve a desired input-output behavior.

Figure 2: Abstract depiction of gradient descent as search in an error landscape, where the choice is among alternative changes to the ANN’s complete weight vector. That change which gives the maximum decrease (arrow) from the current location (black dot) is preferred. Note that in this case, the maximum decrease, a greedy choice to move to the right, does not lead in the direction of the global error minimum, which resides on the far left.

Backpropagation is not without its shortcomings, which include a tendency to get stuck in local error minima and a common need for extremely many repeat presentations of an entire training set, both of which can be argued to have some precedence in the brain but rarely with the same negative consequences as in ANNs. Hence, an ongoing challenge in ANN research is to find other sources of neuro-inspiration that can translate more directly into problem-solving success. Many such attempts begin at the level of the synapse, and its modification by predominantly local, Hebbian means.

3 Hebbian Learning Models

The key insight of Hebb (1949), that stronger synaptic bonds form between neurons that fire together, underlies numerous learning rules for artificial neural networks, a few of which are discussed below. These rules are typically local in that synaptic change hinges solely on the behaviors of the presynaptic and postsynaptic neurons, and not on any global error signals (as in classic supervised ANNs). This locality is appealing from both a biological angle, since the brain appears to employ Hebbian learning, and from an ALife perspective, wherein complex system functionality emerges from the myriad local interactions of
simple components, in the complete absence of a global controller.

Unfortunately, the naive application of local rules across repeated ANN learning rounds can quickly lead to explosive weight increases and decreases, thus driving networks to states of extreme instability or deeply entrenched stagnation. Only by invoking additional mechanisms, which typically lack biological plausibility and/or strict locality, can the modeller reign in these networks, forcing them into behavioral regimes exhibiting an ample mix of stability and adaptivity.

In the discussion that follows, \( u_i \) and \( v \) will typically represent either the last activation level or the difference between a) the firing count of the neuron during the past \( m \) time steps, and b) the average firing count (per \( m \) time steps) computed over \( M \) such \( m \)-step periods. The term neural output will be used as a general reference to \( u_i \) or \( v \), without any assumption about the exact nature of the physical variable.

Also, the term long-term potentiation (LTP) refers to a strengthening of a weight, while long-term depression (LTD) denotes a weakening. The long-term aspect of each change stems from the biological uses of LTP and LTD, where this type of synaptic change persists for hours, days or longer. In ANNs, there is no such guarantee of the duration of the change.

The classic Hebbian learning rule is simply:

\[
\Delta w_i = \lambda u_i v 
\]  

(2)

where \( \lambda \) is a positive real number (often less than 1) representing the learning rate. Equation 2 captures the basic proportionality between weight change and the correlation between the two neural outputs.

![Diagram](Image)

Figure 3: The essence of learning in neural networks: the comparison of pre-synaptic and post-synaptic firing histories determines changes to the connection weight between the two neurons. Here, weights have a single subscript, denoting the pre-synaptic neuron. Graphs are of the neuron’s membrane potential as a function of time.

When \( u_i \) and \( v \) represent recent or time-averaged firing-rates, as they often do, they are never negative. Thus, the right-hand side of equation 2 is always positive, and weights increase without bound. In fact, a
positive feedback occurs, since positive firing rates produce weight increases, which insure that the influence
of $u_i$ upon $v$ gets stronger, which tends to raise $v$'s firing rate, which then elevates the weight, etc.

One can simply put an upper bound on weight values, but then all weights eventually reach this limit and
the ANN has no diversity and hence no interesting information content.

Another attempt to relieve the problem is to view $u_i$ and $v$ as either a) membrane potentials, or b) the
differences between current and average firing rates. Both of these interpretations allow $u_i$ and $v$ to be
positive or negative. Thus, $u_i v$ will often be negative, and weights should decrease as well as increase.

The latter interpretation has spawned several useful learning rules in which presynaptic, postsynaptic or
both firing rates are relativized to a normal level. For example, the weight-updating expression of equation
3, known as a homosynaptic rule, normalizes only the postsynaptic output to a threshold, $\theta_v$:

$$\Delta w_i = \lambda(v - \theta_v) u_i$$  \hspace{1cm} (3)

Notice that this rule can exhibit long-term potentiation LTP or LTD depending upon whether $v$ is above
or below $\theta_v$, which typically represents an average firing rate (computed and updated over the course of
a simulation). The term homosynaptic refers to the fact that if $v$ is above (below) threshold, then all
presynaptic neurons that have a non-zero firing rate will experience LTP (LTD) along their connection to $v$.
That is, all presynaptic neurons that fire will see the same type of change.

Conversely, equation 4 constitutes a heterosynaptic rule, since, when $v > 0$, only those presynaptic neurons
that fire above threshold $\theta_i$ will experience LTP, while the others - even those that do not fire at all - will
experience LTD:

$$\Delta w_i = \lambda v (u_i - \theta_i)$$  \hspace{1cm} (4)

A popular learning scheme that only normalizes the postsynaptic neuron but which requires both pre- and
postsynaptic firing is the BCM (Bienenstock, Cooper and Munro) rule (Bienenstock et al., 1982):

$$\Delta w_i = \lambda u_i v(v - \theta_v)$$  \hspace{1cm} (5)

Rules that normalize both pre- and postsynaptic neurons include:

$$\Delta w_i = \lambda(v - \theta_v)(u_i - \theta_i)$$  \hspace{1cm} (6)

and

$$\Delta w_i = \lambda v(v - \theta_v)u_i(u_i - \theta_i)$$  \hspace{1cm} (7)

3.1 Weight Normalization

Returning to the pure Hebbian learning rule of equation 2, as described above, when $u_i$ and $v$ are non-
negative numbers, the weights will increase without bound. One seemingly obvious fix is to use activation
functions whose outputs can be negative, for example, the hyperbolic tangent function described earlier. Unfortunately, this is not enough to avoid positive feedback and the ensuing unstable increase in weights (See Appendix ?? for details).

To avoid problems of weight-vector instability that occur with so many of the popular learning rules, more direct methods of weight restriction are available. One of the most direct mechanisms is to simply normalize the weights associated with a particular neuron. Here, the options include:

1. **which** weights to normalize, for instance weights on all incoming (or outgoing) links to (from) each neuron.

2. **how** to normalize, whether by:
   (a) dividing each weight by the sum of the weights or the sum of the absolute values of the weights (in cases where weights can be positive or negative), or
   (b) subtracting the same quantity from each weight.

Another common approach is to randomly initialize the weights and then, for each node, store one of two sums: those of its incoming or outgoing connection weights. Assume that you choose the incoming weights, whose initial sum for node $i$ is $\sigma_i$. Then, after each round of learning-based weight change, use equation 8 to renormalize each weight such that the input sums for each node equal the originals.

$$w_{ij} \leftarrow \frac{\sigma_i w_{ij}}{\sum_{j=1}^n |w_{ij}|}$$  \hspace{1cm} (8)

Although computationally expensive, these methods do have the desired effect of preventing runaway weights.

A simpler and computationally cheaper method is the Oja rule (Oja, 1982), which includes a forgetting or leakage term that involves the weight itself:

$$\Delta w_i = \lambda v (u_i - v | w_i |) = \lambda u_i v - \lambda v^2 | w_i |$$  \hspace{1cm} (9)

Note that the first term in this rule is purely Hebbian, while the second term involves leakage that is proportional to both the magnitude of the current weight and the post-synaptic firing level. This second term combats the standard positive-feedback problem with the earlier rules: when neurons fire at high rates, synapses tend to strengthen, which helps neurons to fire even harder in the future. This tendency for post-synaptic firing to increase weight leakage reappears in the upcoming discussion of spike-timing dependent plasticity (STDP).

As detailed in (Oja, 1982), the Oja rule stems from a simplification of the Taylor-series expansion of the following (standard) weight-normalization scheme, wherein each new weight value is divided by the Euclidean sum of all new weights:

$$w_i \leftarrow \frac{w_i + \lambda vu_i}{\left(\sum_{i=1}^n [w_i + \lambda vu_i]^2\right)^{0.5}}$$  \hspace{1cm} (10)
Thus, the beauty of the Oja rule is its ability to achieve the same result as weight normalization, but without actually performing that computationally-expensive process after each round of learning. In short, weight normalization emerges from repeated applications of the local update rule.

The Oja rule has nice theoretical properties, including the ability to perform principle component analysis (PCA), though it lacks complete biological plausibility. However, its stability is easily proven (Dayan and Abbott, 2001), and thus it is a popular learning mechanism for ANNs designed to solve complex engineering problems by predominantly local learning mechanisms.

4 Unsupervised ANNs for AI

All of the above mathematical formulations of Hebb’s Rule provide the basis for unsupervised learning: the tuning of synapses despite the lack of external feedback from an instructor, or from the environment in general. The network itself must extract meaningful relationships (and build useful higher-level structure) from the sensory/input data. These may be invariant patterns in the data or key variants that allow the system to differentiate important classes of inputs. Either way, the system learns these concepts by itself.

In many cases, the result of unsupervised learning is a set of classes or clusters, where each previous input scenario falls into one of them. These facilitate generalization, since new input cases can be ushered into the proper cluster and then handled with the action associated with that group. This ability to generalize behavior is critical for the survival of living organisms and essential for the success of AI systems in complex environments (where all possible input scenarios cannot possibly be enumerated and planned for ahead of time).

Many ANNs used in AI research abstract away a good deal of the biological detail while retaining the essence of Hebbian learning. These models have legitimate utility for many of AI’s unsupervised learning problems while providing further instances of search and emergence in a neural context. In many of these cases, the emergent global patterns stem from local competitive and/or cooperative interactions among artificial neurons. In general the basic Hebbian notion of firing together and wiring together has a very cooperative connotation: neurons working in concert will tend to promote one another’s activity. Conversely, neurons can often be viewed as competitors in trying to a) excite a common postsynaptic neuron, or b) become the main postsynaptic detector of a presynaptic pattern. These mechanisms take center stage in the ANNs below.

4.1 Hopfield Networks

Hopfield networks (Hopfield, 1982) are a popular species of ANN that, despite their simplicity, capture two of the brain’s key functions: storage and retrieval of distributed patterns. They do this in a completely unsupervised manner by recognizing correlations among neural firing histories and modifying weights to record those relationships.

In the brain, many forms of information appear to be distributed across large populations of neurons. This population coding has several advantages:

1. Storage efficiency – in theory, k neurons with m differentiable states can store $m^k$ patterns.
2. Robustness – if a few neurons die, each pattern may be slightly corrupted, but none is lost completely.
3. Pattern completion – given part of a pattern, the network can often fill in the rest.

4. Content-addressable memory – patterns are retrieved using portions of the pattern (not memory addresses) as keys.

Hopfield nets take advantage of population coding to store many patterns across a shared collection of nodes, with each node representing the same aspect of all stored patterns and each connection weight denoting the average correlation between two aspects across all stored patterns.

Figure 4 displays an auto-associative Hopfield network, where auto implies that the correlation is between aspects/components of the same pattern. In this case, the components are simply small regions of the image, with components a and b representing small regions centered at distinct locations of the image plane. An auto-associative network encodes the average correlations (across all stored patterns) between all pairs of components.

In Figure 4, the correlation between components a and b is computed for each of the two images. It is positive in the upper left image, since a and b both contain some black color, but it is negative in the upper right image, since a contains black but b does not. Thus, the left image contributes +1 to the average correlation between a and b, while the right image contributes -1. If there are P patterns to store, then P such correlations will be averaged for every pair of components. This correlation analysis of all P patterns constitutes the training phase of the Hopfield algorithm. It is where the learning occurs.

![Figure 4](image)

Figure 4: Overview of basic procedures for an auto-associative Hopfield network. (Top) the training (learning) phase. (Bottom) the testing (pattern retrieval) phase.

Note that this training reflects Hebbian learning at a coarse level: when two components are highly correlated in a pattern, then the link between their nodes in the ANN will be strengthened. Alternatively, a negative correlation (i.e. one component is on/black while the other is off/white) leads to a weight reduction.
The Hopfield network (a small portion of which appears in the middle of Figure 4) has a clique topology, meaning that every node is connected to every other node. Each node represents a component, and each arc weight denotes the average correlation between the components of the arc. In Hopfield networks, the arcs are bidirectional, since a correlation is a symmetric property.

Once trained, the auto-associative Hopfield net can be employed to retrieve one of the P original patterns when given only a portion (or corrupted version) of it, as shown at the bottom of Figure 4.

The retrieval process begins by loading the partial pattern into the network: components that are on/black in the partial pattern will have their corresponding ANN nodes set to a high activation level, while those that are off will engender low activation levels. The network is then run, with nodes summing their inputs and using their activation functions to compute new output levels. This continues until either the network reaches a quiescent state (i.e., no nodes are changing activation levels) or a fixed number of update steps have been performed. The final activation levels of the nodes are then mapped back to the image plane to create the output pattern. For example, if node f has a high activation level, then the f component of the image will be painted black.

Similar to an auto-associative net, a hetero-associative net records average correlations between pairs of patterns. These are useful for associating one pattern with its typical successor pattern in a sequence. Hence, when given the predecessor pattern, the network can predict the successor.

### 4.2 Basic Computations for Hopfield Networks

In Hopfield and many other associative networks, the learning phase is a one-shot, batch process in which all patterns are analyzed and all correlations averaged. The weights of the network are then set to those averages and never modified.

A typical learning (i.e. weight-assignment) scheme for Hopfield networks is:

\[
    w_{jk} \leftarrow \frac{1}{P} \sum_{p=1}^{P} c_{pk}c_{pj}
\]  

(11)

where P is the number of patterns, \(c_{pk}\) is the value of component k in pattern p, and \(c_{pk}c_{pj}\) is the local correlation in pattern p between components k and j.

For a hetero-associative network, the corresponding scheme is:

\[
    w_{jk} \leftarrow \frac{1}{P} \sum_{p=1}^{P} i_{pk}o_{pj}
\]  

(12)

where P is now the number of pattern pairs, \(i_{pk}\) is the kth component of the predecessor (input) pattern of pair p, and \(o_{pj}\) is the jth component of the successor (output) pattern of pair p. The product of the two components is the local (k,j) correlation for pair p.

Once all weights have been computed, the net can be run by loading input patterns and updating activation levels. For discrete Hopfield networks, where firing levels are either +1 or -1, the following activation function is common:
\[
c_k(t + 1) \leftarrow \text{sign}\left( \sum_{j=1}^{C} w_{kj} c_j(t) + I_k \right)
\]

where \( C \) is the number of components (for example, 64 in an 8-by-8 image plane), \( c_k(t + 1) \) is the activation level of the \( k \)th component’s neuron at time \( t + 1 \), \( w_{kj} \) is the weight on the arc from node \( j \) to node \( k \), and \( I_k \) is the original input value for component \( k \). The \( I_k \) term insures that the original bias imposed by the input pattern has an effect throughout the entire run of the network. \(^1\)

Figure 5 illustrates the basic training and test procedure for Hopfield networks.

---

\(^1\)The summation in equation 13 is the standard integration function, as described in the introductory chapter on ANNs.
4.3 Search in Hopfield Networks

The process by which a running associative network gradually transitions to quiescence has a search-like quality. In this case, a state is a vector consisting of the activation levels of each neuron in the ANN, and the biases that push it toward equilibrium are the connection weights, whose values reflect the network’s history of pattern learning.

Unfortunately, stable states do not necessarily correspond to any of the original P input patterns. They may be spurious, i.e., they map to patterns that were not part of the training set.

Figure 6 illustrates this problem. The original input pattern (top) is loaded into the auto-associative network, but nothing guarantees that the quiescent state will map to the correct pattern (bottom middle) instead of one of several spurious ones (bottom left and bottom right).

Hopfield (1982) quantified the search for quiescence as a search for minima on an energy landscape by defining a function for mapping ANN states to energy levels. The general Hopfield learning procedure (equation 11) and activation function (equation 13) then insure that low-energy states are those corresponding to patterns on which the net was trained. However, nothing prohibits some spurious patterns from achieving locally-minimal energy as well. Here, a local minimum is defined as a state, M, such that any state M* that is created by updating all activation levels of M once, has higher energy than M.
Hopfield’s energy function is:

\[ E = -a \sum_{k=1}^{C} \sum_{j=1}^{C} w_{jk} c_j c_k - b \sum_{k=1}^{C} I_k c_k \]  \hspace{1cm} (14)

where \( c_k \) is the activation level of component k’s neuron, \( I_k \) is the original input value loaded into the kth component’s neuron, and \( a \) and \( b \) are positive constants.

Albeit complex in appearance, equation 14 is actually quite intuitive. Consider the term \( w_{jk} c_j c_k \) and assume that \( w_{jk} > 0 \), meaning that the jth and kth components had, on average, a positive correlation in the training patterns. Next, consider two cases:

1. \( \text{sign}(c_j) = \text{sign}(c_k) \), which means that the jth and kth components are both on, or both off. Hence, they are positively correlated in the current state of the ANN. This agrees with \( w_{jk} \), which also indicates a positive correlation between components j and k, since \( w_{jk} > 0 \). Hence, there is no conflict, only agreement, between \( c_j \), \( c_k \), and \( w_{jk} \). This is reflected in the fact that \( w_{jk} c_j c_k > 0 \). Since the summations in equation 14 are preceded by negative factors, each pair of activation values that agrees with the corresponding weight will contribute negatively to the total energy, where lower total energy means more local agreement.

2. \( \text{sign}(c_j) \neq \text{sign}(c_k) \), which means that components j and k are negatively correlated in the current state. This disagrees with the positive weight between them and is reflected in the fact that \( w_{jk} c_j c_k < 0 \). Hence, this pair of components will contribute positively to the total energy.

There are a similar set of cases when \( w_{jk} < 0 \); in those, agreement is signaled when \( \text{sign}(c_j) \neq \text{sign}(c_k) \).

Given Hopfield’s energy function, we can now sketch an energy landscape for an associative network, where network activation states map to energy levels. Figure 7 illustrates a possible landscape for the hypothetical situation of Figure 6. Note that the two spurious patterns occupy local minima and are thus deceptive quiescent states for Hopfield search.

Just as Simon’s ant runs along trajectories determined by the beach’s terrain, and Simon’s expert draws conclusions based on the history of her experience, Hopfield’s ANN runs downhill along a landscape sculpted by the networks history of learned patterns.

The emergent nature of Hopfield networks is certainly debatable. Though only neuron-neuron interactions govern the emergence of global patterns, the clique topology of the network seems to stretch the definition of local. Still, no single group of neurons run the show; the combination of all low-level activation levels and weights determines the final outcome.

### 4.4 Hopfield Search in the Brain

The behavior of real brains is believed to exhibit many of the same properties as Hopfield networks in that:

1. The strength of synaptic connections between two neurons (or populations of neurons) often reflects the degree to which the receptive fields of those neurons are correlated, where the receptive field of a
neuron is that part of the sensory space (for example, a small region in the upper left quadrant of the visual field) for which the neuron appears to be a detector.

2. Certain activity patterns appear to be stable attractors, i.e., quiescent states, to which real neural networks eventually transition. These attractors may represent salient concepts in the brain.

Consider the duck-rabbit flip-flop picture of Figure 8. Most people cannot view this as both a rabbit and duck simultaneously; rather, the interpretations seem to alternate, particularly if you stare at the picture for a long time. This is evidence that both interpretations represent stable attractors in memory but that there exists some overlap between the neural states corresponding to each, as shown by the abstract neural networks on the left of the figure.

The components of each attractor stimulate one another, as shown by the thick links in Figure 8, in much the same way that highly correlated nodes in a Hopfield network excite one another due to their high connection weights. However, due to overlap, when one attractor is active, some of its components (such as the eyes and neck nodes) also stimulate portions of competing attractors.

In addition, neurons are known to habituate to stimuli, meaning that they tend to reduce their AP production in the presence of a continuous stimulus. To see this, try staring at something for a few minutes and feel how hard it is to keep your thoughts focused on that particular item; the mind tends to wander.

Hence, a stable attractor such as the duck pattern will habituate while simultaneously lending some stimulation to the mouth and ears nodes. Eventually, the balance of firing power shifts and rabbit becomes the main interpretation, until it habituates and the duck returns.
4.5 Competitive Networks

In certain types of ANNs, the nodes compete for activity such that the most active nodes can both a) inhibit other nodes from firing, and b) localize learning to only their own connections, typically the incoming (a.k.a. afferent) links. In the brain, topologies in which neurons inhibit many of their neighbors (whether immediate or slightly more distant) are commonplace. This often serves the important function of filtering noise, such that the final stable pattern consists of only the neurons that detect meaningful signals. Other competitions lead to useful structural isomorphisms between aspects of the physical world and regions of the brain specialized to handle those properties. These topological maps are beautiful examples of how the brain encodes much of the inherent structure of the physical world.

Figure 9 illustrates the essence of competitive learning in the brain. In many brain regions, particularly the cortex, the main (often excitatory) neurons are known as principal cells. In close proximity are one or several types of interneurons, which are typically inhibitory. Active principal cells tend to stimulate nearby interneurons, which then inhibit all of their post-synaptic targets. This manifests feedback inhibition, wherein the activation of one (or a few) neurons in a layer quickly leads to the inhibition of the rest of the layer. Thus, the principal cells can be viewed as competing; and that neuron which is most active simultaneous with a subset of input neurons can become a detector for that input pattern.

In practical applications of competitive ANNs, the focus is on the weights of the input arcs to each output node, $n_i$. The vector of input weights to $n_i$, $\langle w_i \rangle = \langle w_{i1}, w_{i2}, \ldots, w_{in} \rangle$, typically represents a prototype of the patterns that $n_i$ is (or has learned to become) specialized to detect. In this sense, each output node represents a class or category, and input patterns can be clustered according to the output node that they maximally stimulate.

Figure 10 illustrates a standard topology for artificial competitive networks, with one input and one output layer. The output nodes represent categories/classes that essentially compete to capture the different input vectors, with each such vector falling into the category whose prototype it most closely matches.
Figure 9: Competitive learning in a layer of neurons. (Top left) Principal cells fire randomly, and neuron P2 fires approximately coincident to the first and fourth input neurons (top row). (Top right) P2 inhibits its neighbor neurons, thus insuring that very few neurons in its layer fire together. The synapses between the active inputs (shown as filled circles) and P2 strengthen by Hebbian learning (denoted by thickened line segments). P2 thus becomes a detector for input pattern 1001. (Bottom left and right) A similar process occurs with P3, such that it becomes a detector for 0101.

Figure 10: The basic topology of a competitive network, with an input layer and an output layer. Each output node represents a class, whose prototype is given by the input weights to that node. Through learning, input cases become associated with different output nodes and thereby become clustered into instance sets (clouds) of the classes represented by those nodes.
The generic competitive ANN algorithm is quite simple. Patterns are repeatedly presented to the input layer and activations recorded on the output layer. The output node with the highest activation on pattern P wins and has its input weights adjusted so that its prototype more closely resembles P. The update formula for the weights into winning node \( n \) on input pattern P is then:

\[
    w_{ij} \leftarrow w_{ij} + \eta(P_j - w_{ij})
\]  

(15)

where \( P_j \) is the jth value of pattern P, i.e., the value loaded onto input neuron j. After many epochs (i.e., presentations of the entire training set), the cases often become clearly segregated into classes, with the prototype vector of each class located close to the middle of the subspace delineated by its cases. In this way, competitive networks function as clustering mechanisms for complex data sets.

In practice, many competitive networks are not really neural networks at all. They are simply lists of prototype vectors that are matched to case vectors. The vector with the closest match, using a standard Euclidian distance metric for n-dimensional space, is updated in the direction of the case.

However, one can capture these same dynamics in an actual neural network that involves a few key components:

1. Many inhibitory links between all output units.
2. Excitatory links between output units and themselves.
3. Normalized case and prototype vectors

The first two features constitute a Maxnet, which insures that when all output nodes are activated, they will compete to shut each other off, while stimulating themselves. After a transitory period, the network settles into a state where the only neuron with a non-zero activity level is the one that originally had the highest activation level.

Maxnets are easy to implement, but care must be taken to properly set the (fixed) weights on the inhibitory and self-stimulating arcs, \( \epsilon \) and \( \theta \), respectively. For example, if the network consists of m output neurons, then the following settings work well:

\[
    \theta = 1, \quad \epsilon \leq \frac{1}{m}
\]  

(16)

4.5.1 Weight Normalization in Competitive Networks

Normalization of base and prototype vectors (though slightly complicated) is the key to understanding how equation 15 achieves its purpose: how adjusting the prototype weight vector of output node \( n \) to more closely match an input case, \( C \), actually insures that \( n \) fires harder on the next presentation of \( C \). This serves as one more example of the importance of weight normalization in neural networks.

First consider the basic Euclidean distance between an input pattern vector, \( P \), and the weight vector for output neuron \( i, \langle w_i \rangle \):
\[
\sqrt{\sum_{j=1}^{n}(P_j - w_{ij})^2} = \sqrt{\sum_{j=1}^{n} P_j^2 - 2P_j w_{ij} + w_{ij}^2}
\]  

(17)

Now, if input vectors and prototype weight vectors are in normalized form, then we know that their unit length is 1. Hence:

\[
\sum_{j=1}^{n} P_j^2 = 1 = \sum_{j=1}^{n} w_{ij}^2
\]  

(18)

Combining equations 17 and 18, we find that:

\[
\sqrt{\sum_{j=1}^{n}(P_j - w_{ij})^2} = \sqrt{\sum_{j=1}^{n} P_j^2 - 2P_j w_{ij} + w_{ij}^2} = \sqrt{2 - 2\sum_{j=1}^{n} P_j w_{ij}}
\]  

(19)

Thus, to minimize the distance between \( P \) and \( \langle w_i \rangle \), we should maximize \( \sum_{j=1}^{n} P_j w_{ij} \), which is just the sum of weighted inputs to neuron \( i \). Hence, the prototype vector with the best match to the input case (i.e., that which is closest to it in Euclidean space) will have the highest sum of weighted inputs and thus will have the highest activation level.\(^2\)

Another way of looking at this is that each normalized vector represents a unit vector in n-dimensional space. The term \( \sum_{j=1}^{n} P_j w_{ij} \) is the dot product of these vectors, and with unit vectors, the dot product is equal to the cosine of the angle, \( \phi \), between the vectors. Then, \( \cos \phi \) is large, i.e., approaches 1, if and only if \( \phi \) approaches 0, i.e. the vectors are similar. Hence, a good match between the normalized input case and prototype is indicated by a large dot product: \( \sum_{j=1}^{n} P_j w_{ij} \).

In summary, if a) the application allows a normalization of input patterns and weight vectors, and b) computational resources permit output neurons to participate in Maxnet competitions to determine the largest activation level, then competitive networks can be implemented as true ANNs that a) learn via equation 15 and b) allow a population of dedicated pattern detectors to arise from local interactions.

### 4.6 Self-Organizing Maps

The neural networks that are perhaps most synonymous with emergence are self-organizing maps (SOMs), a.k.a. Kohonen nets (Kohonen, 2001). These employ a dynamic mixture of competition and cooperation to produce topological mappings between two spaces, e.g., a perceptual field and a neural layer, two neural layers, or a neural layer and an actuator field.

In many competitive maps, the spatial relationships between the output neurons have no significance, and thus it makes no sense to discuss the neighbors of a neuron. However, these relationships have meaning inside the brain, since the firing of a neuron in a region of the brain will often have consequences for nearby neurons.

---

\(^2\)Competitive learning networks often use a linear activation function, so a higher weighted sum of inputs yields a correspondingly higher activation.
When neurons are viewed as detectors of various phenomena, whether visual, olfactory, auditory or tactile, it is very often the case that nearby neurons in the brain serve as detectors for similar stimuli. These local similarities of preferred stimuli are abundant in the brain. Figure 11 shows one impressive example from the auditory cortex, where a long chain of processing layers maintain a correlation between neural neighborhoods and sound frequencies.

Figure 11: Tonotopic Maps: Topological mapping between the space of sound frequencies and 7 successive layers of the auditory processing system in the mammalian brain. The correlation between frequencies and neuron locations is preserved through all 7 layers. Based on images and descriptions in (Bear et al., 2001; Kandel et al., 2000).

The essence of a topological map is the isomorphism between two spaces, at least one of which is a population of neurons. The two spaces are isomorphic when components that are close (distant) in one space map to components that are close (distant) in the other space. This basic idea is shown in Figure 12, where the top mapping is not isomorphic, but the bottom one is.

SOMs operate similarly to standard competitive networks in that a case vector is presented, and the best-matching prototype is modified in the direction of the case. However, Kohonen added an interesting twist: neurons have a meaningful spatial relationship to other neurons, and when an output neuron wins a case, it shares the prize with its neighbors in that both winner and neighbors adjust their prototypes toward the case. This neighborhood of sharing changes during the training phase, typically beginning with a large radius that decreases as training progresses and prototypes specialize toward particular subsets of the case patterns. Over time, this sharing results in an isomorphic mapping in which the prototypes of nearby (in neuron space) neurons are more similar to one another than to those of distant neurons. In addition, SOMs typically have a dynamic learning rate (\(\eta\) in equation 15) that begins large and decreases during the run.

The net result is a convergence of the Euclidean and topological neighborhoods of the neurons in the SOM, as depicted at the bottom of Figure 12. The significance of this convergence is multi-faceted, including physical factors such as reduced total wiring between brain layers, measured in terms of the lengths of axons and dendrites.

In terms of brain function, a key advantage of topological maps is that if situation \(A_1\) maps to neuron (or
more likely neuron population) $N_1$, presumably via a learning process, then situations similar to $A_1$, such as $A_2$ and $A_3$ would map to neighbors of $N_1$, which would elicit similar behavior. Thus, the topological map allows the brain to reuse and generalize its behaviors to situations similar to those that it has explicitly learned. The neighbors of $N_1$ may not share all of its functionality, which would seem appropriate, since $A_2$ and $A_3$ may require similar, but not exactly the same, actions as $N_1$ supports.

SOMs have been applied to a wide range of clustering and unsupervised-learning problems. One of the more creative of these (Durbin and Willshaw, 1987) involves the Traveling Salesman Problem (TSP), in which the goal is to find the shortest circuit that includes every point (e.g., city) in a particular collection (Garey and Johnson, 1979). Framing TSP for solution by an SOM requires the formalization of two spaces: a) the Euclidean space that houses a set of $N$ 2-dimensional points (e.g., city locations), and b) the topological space of $M$ neurons, which is a 1-dimensional ring: each neuron has exactly two immediate-neighbor neurons. The connection between the two spaces is via the afferent links of the neurons; each neuron has two links whose weights represent $x$ and $y$ values of the neuron’s prototype city location.

The N city locations then serve as input vectors to the SOM, and each neuron competes to be the winning detector of a certain subset of those locations. As shown at the top of Figure 13, each neuron of the ring houses a prototype (weight) vector that maps to a point on the Cartesian plane. The bottom of Figure 13 projects the ring onto the city space, with each neuron’s location given by its prototype vector. The projected neuron ring then serves as scaffolding for a TSP tour (as detailed by Durbin and Willshaw (1987)). When these prototype vectors are randomly initialized, the ring begins as a twisted mess(bottom left), but after many rounds of training, with a gradually shrinking ring neighborhood and declining learning rate, the ring unfolds into a circuit (bottom right) providing the backbone of an optimal TSP tour. From competition and cooperation emerges a global solution.
Figure 13: Illustration of the use of a self-organizing map to solve the Traveling Salesman Problem (TSP) using the method of Durbin and Willshaw (1987). (Above) The neuron topology is a ring, with each neuron’s weight vector denoting a location in 2-d space. (Bottom) City locations (stars) and neurons (octagons) shown on the Cartesian plane, with coordinates corresponding to a) the locations of the cities on a scaled map, and b) the weight vectors of each neuron.
5 Spikes and Plasticity

The earlier mathematical models of Hebbian plasticity assume that the presynaptic and postsynaptic outputs represent firing rates, which are sufficient for many ANNs (such as Hopfield and Kohonen nets) and their application scenarios, wherein neural outputs typically represent general activity levels, not actual action-potentials, i.e., spikes. However, in the brain, the basic Hebbian model also holds at the level of spike trains. At the firing-rate level, the model indicates that neurons with simultaneously high activity tend to develop strong synaptic interconnections - i.e., the model explains correlation-based learning. But dropping one level to that of the spike times of individual neurons indicates how the brain may learn causal relationships among stimuli, and thus become adept at predicting future from present stimuli.

5.1 Prediction

Several prominent neuroscientists have recently championed the ability to predict as a fundamental function of all brains, from insects to humans (Llinas, 2001; Hawkins, 2004). The basic argument for prediction’s unique importance begins with sensorimotor behavior, where the processing speed of sensory inputs simply cannot keep pace with the rate at which the coupled agent-environment system changes in many (often intense and/or life-threatening) situations. In calculating the next action, the nervous system does not have time to interpret all sensory data. Instead, it uses predictions of future states to guide action, and only when predictions conflict with reality must the system realign its expectations with the sensory present. Neuroscientists (Wolpert et al., 1998; Llinas, 2001) generally agree that the brain needs predictive abilities similar to those found in control theory (such as Kalman filters) to handle the different timescales involved in sensing, moving, subconscious action-choice and conscious decision-making.

As described in my earlier writing (Downing, 2007) (based on a similar distinction made by Squire and Zola (1996)), a prediction involving an explicit expression of knowledge is termed declarative, whereas one in which no such expression occurs but the agent’s behavior clearly indicates anticipation of a particular situation is procedural. The former tend to involve information stored in the cortex (Hawkins, 2004) and hippocampus (Gluck and Myers, 1989), whereas the latter involve the cerebellum (Wolpert et al., 1998) and basal ganglia (Houk et al., 1995).

Assume the dictionary definition (Prediction, 2009) of prediction: to declare or indicate in advance. Then, a typical predictive-learning scenario involves an organism that witnesses event A followed by event B. It learns this correlation, along with the all-important temporal precedence of A over B such that, in the future, when the organism observes A, it can declare (e.g. via language) or indicate (via various anticipatory actions) that B will soon occur. Thus, if a neuron (or neural cluster) represents A, its activation should lead to the stimulation of a neuron or cluster representing B; but the linkage need not (and in many cases should not) be bi-directional: B’s cluster need not stimulate A’s. The trick is to learn this temporally-dependent relationship despite the fact that the neurons for A and B will both be highly active within overlapping time windows.

5.2 Spike-Timing Dependent Plasticity (STDP)

Several neurological studies (summarized by Sen Song and Abbott, 2000) reveal the presence of two complementary tuning curves for synaptic strength, both based directly on $\Delta t$, the time of the pre-synaptic spike minus that of the post-synaptic spike. Figure 14 shows these relationships of $\Delta t$ to synaptic potentiation (when $\Delta t \leq 0$) and depression (when $\Delta t > 0$).
Figure 14: The essence of spike-timing dependent plasticity (STDP) involves the dependence of the change in synaptic strength ($\Delta s$) upon the timing difference between pre- and post-synaptic spikes ($\Delta t$). The time window for these dynamics is roughly 40 ms, and the maximum magnitude of change is roughly 0.4% of the maximum possible synaptic strength/conductance (Sen Song and Abbott, 2000).

On the surface, basic STDP would seem to explain how the brain can learn causal relationships: if neuron A fires just before neuron B, then the connection from A to B is potentiated; and in the future, firing of A should stimulate B, thus predicting the event that B represents. Also, since B fires after A, the $B \rightarrow A$ link would actually weaken, thus reducing the chance that the future occurrence of B’s event would predict A’s event. However, neurons rarely fire single action potentials, but many per second, and often so many that the 40-ms windows overlap to such a degree that there is no obvious pairing of spikes with which to assess before-after relationships.

Figure 15: Two hypothetical neurons firing at 160 Hz (or 8 spikes per 50-ms time window).

For example, in Figure 15, each neuron fires 8 times within a 50-ms time frame, with neuron A’s 3 pulses (labeled U) leading off the frame. Since these spikes all come prior to neuron C’s first triple (labeled X), neuron A would seem to be the predecessor (at least within this one time frame). However, by the end of sequence Y, neuron C has fired more times (6) than neuron A (5). To further confuse the issue, the pairing of pre- and post-synaptic pulses quickly becomes non-trivial. For example, is the second spike of X a successor to the second (or third) spike of U, or is it a predecessor to the first spike of sequence V? Maybe it should be treated as both? Maybe all relationships between all pre- and post-synaptic spikes (within a 40 ms window of one another) should factor into the calculation of the synaptic change, $\Delta s$? In fact, many methods for comparing spike trains (see (van Rossum, 2001) for several examples) consider all such pairings.
But how might STDP arise from a pair of neighboring neurons with complex spike trains? Surely the brain does not explicitly use each pairing of pre- to post-synaptic spikes to govern synaptic modification, as do many computational neuroscience algorithms.

One intriguing possible explanation arises from Song and Abbott’s STDP model (Sen Song and Abbott, 2000). As depicted in Figure 16, the pre- and post-synaptic spikes in this model each lead to the modification of various factors ($P_a$, $P_b$, and $M$), and these affect synaptic conductances ($w_a$ and $w_b$), but at no point are spike times compared (to produce a $\Delta t$). Still, the algorithm yields STDP as in Figure 14; furthermore, it achieves weight stabilization without explicit normalization of conductance values. In short, STDP and weight stability emerge from a small set of local interactions.

Figure 16: The essential interactions in the algorithm of Sen Song and Abbott (2000) depicted for a circuit of 2 presynaptic neurons (a and b) and one postsynaptic neuron (c). The key presynaptic memory parameters, $P_a$ and $P_b$, are incremented (by $m_+$) when neurons a and b fire, respectively, while the memory for postsynaptic firing (M), increases (by $m_-$) when c fires; typically, $m_- > m_+$. All three memories experience constant decay. The parameter k represents the maximum attainable synaptic weight/conductance. Presynaptic firing initiates decreases in the corresponding weights ($w_a$ and $w_b$), while postsynaptic firing raises the weights. The differentials in replenishment and decay between the presynaptic and postsynaptic memories result in the ultimate increase or decrease of each synaptic weight in a manner that mirrors the relationships in Figure 14.

In this procedure, note that presynaptic firing leads to increases in the corresponding memory terms ($P_a$ or $P_b$). If neuron A fires several times in quick succession, as on the far left of Figure 15, then $P_a$ increases without suffering a lot of decay. If postsynaptic neuron C fires shortly thereafter, the update of $w_a$ is significant, whereas a longer delay (i.e. a more negative $\Delta t$ in Figure 14) entails greater decay of $P_a$ and thus a less sizeable increment to $w_a$ when C eventually does fire.
Conversely, if C fires many times just prior to a spike from neuron A, then C’s activity raises M. If A fires right after this buildup of M, \( w_a \) experiences a large decrease, whereas a longer delay entails greater decay of M and a smaller decrease in \( w_a \) (similar to the far right end of the depression (right) half of Figure 14).

In both of these scenarios, the memory variables and their effects upon the weights produce the fundamental STDP relationships of Figure 14, but without the explicit comparison of numerous pre- and postsynaptic spike times. This exhibits an interesting form of emergence in that LTP and LTD constitute \textit{temporally} global states (i.e., those of long duration) that arise from interactions that are spatially and \textit{temporally} local. In Figure 16, note that none of the update equations involve time stamps: variables change based on the current values of other variables, not on the details of their histories. Although \( P_a, P_b \) and M give lumped indications of previous spiking activity, they are not detailed memories; their interactions with other variables do not require comparisons of a temporally non-local nature (as when comparing spikes across large time windows). All weight modifications stem from the current values of a few leaky integrative memories – mechanisms that reappear in several guises throughout this book.

In addition to the classic STDP produced on individual synapses, this model also leads to an emergent competition among synapses for conductance strength. Those presynaptic neurons whose action potentials consistently precede postsynaptic spikes are rewarded with potentiation, while those whose spikes tend to come after the downstream AP become depressed. Furthermore, randomly-firing afferents also yield synaptic depression, due to a critical element of the model: \( m_- > m_+ \). Since a randomly-firing neuron’s spikes will precede and succeed a postsynaptic spike with equal probability, its net synaptic change would be negligible if \( m_- \) were to equal \( m_+ \). Thus, the model insures that only temporal precedence of presynaptic spikes can produce potentiation, while all else leads to depression; the weight vector of the postsynaptic neuron becomes tuned to the predictive presynaptic neurons and ignores all others. Thus, the tuning of synapses to capture salient afferents is another emergent aspect of neural learning.

Presynaptic neurons may cooperate as well as compete in their implicit attempts to control the firing pattern of a postsynaptic neighbor. Consider a situation (Figure 17) in which several presynaptic neurons in a group (G) fire at the same time, t, while a lone presynaptic competitor (C) fires at time \( t+2d \), where \( d \) is just a few milliseconds. Furthermore, assume that these combined G+C inputs stimulate the postsynaptic neuron (P) to fire at time \( t+3d \). STDP entails that the C-P synapse will strengthen to a greater degree than the G-P synapses. But all weights should increase, as will the net effect of G and C upon P. Hence, in the future, P may fire earlier than \( t+3d \). In fact, although the G-P weights may not be as strong as the C-P weight, the cumulative effect of the G neurons could be sufficient to excite P to threshold before time \( t+2d \), at, for example, time \( t+d \). This modification to the relative firing order now means that the G-P weights should continue to increase, but the C-P weight will decline, since C now fires after P. In short, by firing together, the G neurons seize control of P, while the solo neuron C, though initially spiking in a perfect position for LTP, loses the competition with G for influence upon P.

As emphasized by Sen Song and Abbott (2000), their STDP model manifests emergent weight normalization via two distinct firing modes of the postsynaptic neuron, P. In one mode, P receives considerable input – its activation function is saturated – and thus fires in response to the average excitation from the entire population of presynaptic neurons, \( G^* \). This normally entails a regular firing pattern that is insensitive to small changes in the firing patterns of individual presynaptic neurons. In this situation, assume that many of these neurons were to correlate with P and thus have rising weights; the risk of explosive weight gain would seem high. However, at some point, a small subset, \( G' \) of \( G^* \) would be sufficient to evoke a spike from P. All members of \( G^* - G' \) would then lose the pre-before-post relationship to P and would thus experience weight decreases on their synapses to P. So the total synaptic strength has an emergent upper bound.

In the second firing mode, P’s total input is much less (and closer to the firing threshold) such that P’s spike train can be driven by the train of one or a few active members of \( G^* \). These members would experience pre-then-post firing behavior to strengthen their synapses by STDP, while less active \( G^* \) neurons would have both a) few spikes, and b) few spikes in the temporal window of P’s spikes. Thus the inactive members
Figure 17: The emergence of cooperative postsynaptic control. (Above left) A group (G) of synchronously-firing (i.e., cooperating) neurons helps stimulate postsynaptic neuron P, as does the single presynaptic neuron C. By spiking just before P, neuron C achieves the greatest LTP (thick line), while connections from G receive some, but less. (Above right) However, the slight increase in G’s efferent connections enables it to stimulate P to fire earlier than C (at time t+d). (Lower middle) This produces strong LTP on G’s efferents (thick lines) while invoking LTD on C’s efferent (dashed line). Thus, C’s influence upon P diminishes, while G controls it.
would not invoke much LTD, so the net result would be an increase in total synaptic weight. Hence, a lower bound also emerges.

Note that in both Song et. al.’s STDP model and in the Oja rule (Equation 9), post-synaptic activity directly affects weight leakage, forming a negative feedback loop that stabilizes each neuron’s vector of afferent weights. Furthermore, in both models, the leakage associated with postsynaptic firing is larger than the weight gain stemming from presynaptic activity. This is achieved by $m_- > m_+$ in the STDP model. In the Oja rule, presynaptic activity ($u_i$) increases synaptic weight by the factor $v$, while weight leakage involves a $v^2$ factor. Since the activation functions in (Oja, 1982) are linear (not, for example, sigmoidal), output values often exceed 1, and thus, $v^2 > v$ for strongly-stimulated postsynaptic neurons. Consequently, a highly active postsynaptic neuron will produce weight increases on only the synapses from its most active afferents; all others will decrease.

In both cases, a local learning algorithm drives the neural circuit to a poised state in the sense that most presynaptic firing patterns elicit no significant response from the postsynaptic neuron; but those patterns to which it has become tuned, can consistently cause it to fire. The poised state is thus an informational state for the brain, indicating the presence of a salient activity pattern, but not responding (with a potentially false positive) otherwise.

5.3 Emergent Predictive Circuits

The above STDP model has tight ties to the work of Artola et al. (1990), which characterizes important preconditions for LTP versus LTD, the combination of which plays an important role in a trial-and-error synaptic-tuning mechanism that appears to underlie many predictive networks in the brain.

From the viewpoint of synaptic electrophysiology, the acquisition of declarative predictive models within the brain’s hierarchical neural network has a very plausible explanation based on bi-modal thresholding. As illustrated in Figure 18, Artola et al. (1990) have shown that weak stimulation of neurons (in the visual cortex) leads to long-term depression (LTD) of the synapses that were active during this stimulation, while stronger stimulation incurs long-term potentiation (LTP) of the active synapses.

As shown in Figure 18, a common connectivity pattern in the brain, particularly in the cortex, involves a combination of bottom-up and top-down pathways, wherein lower-level sensory signals enter a neuron via proximal dendrites, while higher-level signals (carrying predictive information) synapse distally (Hawkins, 2004; Mountcastle, 1998). All other factors being equal, this gives the bottom-up signal greater influence than the top-down signal upon the post-synaptic activity of neuron N. In short, sensory reality dominates speculation. However, experience and learning can change this relationship.

Three learning cases (summarized in Figure 19) are worth considering with respect to neuron N, its low-level sensory inputs, S, and its high-level predictive inputs, P. First, if S is active but P is not, then the effects of S on N will produce a high enough firing rate in N to incite LTP of the S-to-N proximal synapse. Hence, N will learn to recognize certain low-level sensory patterns.

Second, if both P and S provide active inputs to N, then an even higher firing rate of N can be expected, so LTP of both the S-to-N and P-to-N synapses should ensue. In essence, the predictive and sensory patterns create a meeting point at N by tuning the synapses there to respond to the P-and-S conjunction. In fact, after repeated co-occurrences of S and P, the synapses in N may strengthen to the point of responding to the P-or-S disjunction as well, in effect saying that it trusts the prediction P even in the absence of immediate sensory confirmation.
In the third case, when only P is active, the distal contacts of the P axons may only suffice to weakly stimulate N, thus leading to LTD: a weakening of the P-to-N synapses. Hence, future signals from P will not suffice to fire N, and thus P’s predictions will not propagate through N in the absence of verification from S. In short, the system learns that P is not a good predictor of S.

These examples of LTP and LTD seem to agree with Song and Abbott’s STDP model. In case 1, if S and N are both firing at a high frequency, with S spikes beginning slightly earlier and thus helping to initiate the N spikes, then LTP of the S-N synapse seems likely. In case 2, with S and P spiking frequently enough to excite N at a comparable frequency, there should be ample pre-before-post activity to produce LTP on both the S-N and P-N synapses. In case 3, assuming that pre-synaptic P produces spikes at a much higher frequency than post-synaptic N, then more decrements than increments of the synaptic weight (e.g. $w_a$ in Figure 16) can be expected as many pre-synaptic spikes go unanswered by N: the weight decreases that follow pre-synaptic spikes are not balanced (or exceeded) by weight increases the follow post-synaptic spikes.

Given this basic wiring pattern for the integration of sensory inputs and top-down expectations, along with an established LTP/LTD mechanism, the brain’s ability to predict has an interesting explanation in terms of search and emergence. This combination of bottom-up and top-down processing as a route to emergent intelligence in a neural substrate was popularized in the 1970’s by *Adaptive Resonance Theory*...
(ART) (Grossberg, 1976b,a), which formalizes the interplay between reality and expectations in an ANN. More recent results in computational neuroscience provide further inspiration for ART-like approaches to cognitive modeling.

Contemporary models of several, diverse, brain regions – including the thalamocortical loop (Rodriguez et al., 2004; Granger, 2006), hippocampus (Wallenstein et al., 1998) and neocortex (Hawkins, 2004) – reveal a common topology, which I call the Generic Declarative Prediction Network (GDPN) (Downing, 2009). Declarative prediction requires machinery that can associate two patterns, both of which have strong correlations with external stimuli. The perception of the first stimulus in the environment should then stimulate thoughts of the second.

Depicted in Figure 20, the GDPN consists of several neurons connected as in Figure 18, with a layer of detector neurons (A, B and C) sandwiched between a layer of sensory inputs and one of top-down predictor neurons (W,X,Y and Z). Once again, note the proximal influence of the sensory signals and the distal connection of the predictors. In contrast to procedural networks, declarative topologies tend to have considerable recurrence: combinations of bottom-up and top-down links.

Consider a situation in which stimulus A precedes stimulus B. The following series of events explains how the network learns to predict B when A occurs in future situations.

First, at time t1, stimulus A has a strong effect upon neuron A, via its proximal synapse. Neuron A then fires and sends bottom-up signals to W,X,Y and Z at time t2. At this level, as in many areas of the brain, neurons fire randomly, with probabilities depending upon their electrochemical properties and those of their surroundings. Assume that neuron X happens to fire during, or just after neuron A, and that synapse S1 is modifiable, then the A-X firing coincidence will lead to a strengthening of S1, via standard Hebbian means. In reality, several such high-level neurons may coincidentally coactivate with A and have their proximal

Figure 20: The Generic Declarative Prediction Network (GDPN). Neurons A, B and C serve as low-level detectors for stimuli A, B and C, while W-Z represent neurons at a higher level that serve as predictors of activity among the detectors. Only the axonal projections from X are shown, though W, Y and Z have similar links to the lower level. The T1 - T4 diamonds represent time steps, while S1 and S2 denote important synapses, as further discussed in the text.
When X fires, it sends signals horizontally and to both higher and lower levels. These latter *top-down* signals have a high fanout, impinging upon the distal dendrites of neurons A, B and C. Since entering distally, along unrefined synapses, these signals have only weak effects upon their respective soma, so at time t3, neurons B and C are receiving only mild stimulation. At this stage, we can metaphorically say that a) X is *waiting* for B and C (and thousands of other low-level neurons) to fire, and b) X *hedges its bets* by investing equally and weakly in each potential outcome. In short, X performs a parallel search for a post-synaptic target that is a *reliable* successor to stimulus A: one that tends to activate directly after neuron A does.

At time t4, when event B occurs, neuron B will spike at a high frequency due to the proximal stimulation from below. This will cause further bottom-up signalling, as when A fired, but the critical event for our current purposes involves the LTP that occurs at synapse S2. Previously, stimulation from X alone was not sufficient to fire neuron B at a high frequency. Now, however, with help from stimulus B, both neurons, X and B, are very active, causing S2 to strengthen. Thus, in the future, the firing of X will send stronger signals across S2, possibly powerful enough to fire neuron B *without help* from stimulus B.

Through one or several A-then-B stimulation sequences, S1 and S2 can be modified to the point that an occurrence of stimulus A will fire neuron A, as before, but this will then directly cause X to fire, which in turn will fire neuron B. Thus, stimulus A will cause the brain to *predict* stimulus B.

Over time, neuron X ceases to hedge its bets and achieves a significant bias toward neuron B. This stems from both the strengthening of S2 and the weakening, via LTD, of X’s synapses upon other detector-level neurons (that are not simultaneously activated by bottom-up signals); X’s pre-synaptic spikes are not answered by post-synaptic spikes among these other detectors, thus causing LTD. So X becomes a dedicated prediction pathway between A and B. After the repeated presentation of many sequential patterns, LTP and LTD gradually convert a blanket of bet-hedging anticipatory links into a smaller population of dedicated connections between associated pattern-detecting neurons.

In the GDPN, terms such as *detector* and *predictor* serve only as scaffolding to explain the slow emergence of a hierarchical network in which the neurons at higher levels represent more complex concepts, which are often detectors (and promoters) of lower-level activation patterns. These support prediction in the sense that the bottom-up formation of some lower-level patterns induces the higher level to activate further primitive patterns, thus *completing the picture* via top-down expectations.

The presence of a temporal ordering in this learning may seem arbitrary. For example, if a child learns the complex concept of a *red ball*, then does *red* originally predict *ball* or vice versa? Rodriguez and Granger’s models of the thalamocortical loop (Rodriguez et al., 2004; Granger, 2006) provide one answer: rather than necessarily linking sequences of real-world events, GDPN networks can associate sequential neural states of perceptual processing, wherein the initial states tend to involve the most salient features (such as *red object*), with other aspects (such as roundness) registering in later steps. This work blurs the borders between prediction and conventional association, since any perception can now be interpreted as a time series of partial interpretations of sensory input, each predicting the next.

The GDPN is most clearly evident in the neocortex, where the individual neurons of Figure 20 are replaced by cortical columns, each functioning as a processing module (Hawkins, 2004; Fuster, 2003). Bottom-up sensory interpretation involves cascades of neural firing from the back (sensory) areas of the brain to the front (executive) areas, while top-down, parallel, search for reliable successors moves front to back.

This search is the GDPN’s key contribution to emergent intelligence. Just as neurons migrate by filopodia searching for footholds, and axons find targets by successive growth and retraction, hierarchical neural layers tune themselves by flooding lower layers with exploratory signals and then allowing LTP and LTD to gradu-
ally convert mats of undifferentiated connections into finely specialized synaptic networks. This exploratory process is parallel and unmistakably unintelligent, but it produces networks capable of remembering, recognizing and producing the intricate patterns of mental life.

6 Place Cells and Prediction in the Hippocampus

The GDPN is also based on the CA3 region of the hippocampus, a brain area exhibiting one of the highest densities of recurrent connections (Rolls and Treves, 1998; Kandel et al., 2000). Neuroscientists generally agree that recurrence is essential for the pattern storage and completion/retrieval that underlies associative learning (Rolls and Treves, 1998). When these associations include a temporal component, they become predictive; and indeed, the hippocampus is also touted as a premier predictive area of the brain, particularly with respect to navigation (Burgess and O’Keefe, 2003; Gluck and Myers, 1989).

![Diagram](image)

**Figure 21:** An illustration of the emergence of phase precession in hippocampal place cells based on the explanation by Mehta (2001). (Top) Assume that a rodent moves through contexts A to F (detected in CA3) prior to and after sensing a key landmark, whose presence is signaled by the firing of place cell X in CA1. Then, firing times for the context detectors A-C should precede that of X, while D-F should fire after. (Bottom) These spike-timing differences produce graded LTP on the leftmost 3 connections and LTD on the rightmost. Hence, in the future, the presence of the leftmost contexts could fire neuron X prior to arrival at the landmark. Thickness of arrows indicates relative strength, with thin dashed lines being the weakest.

This is probably best exemplified by the well-documented phenomenon of *phase precession* in hippocampal place cells (Burgess and O’Keefe, 2003) — which I will also refer to as *context detectors*, most of which lie in neighboring regions CA3 and CA1 — whereby a neuron that codes for location X begins to fire (predictively) at locations prior to X along a familiar path. As detailed in Figure 21, the formation of these predictive
links between detector neurons coding for successive locations along an often-travelled route is convincingly explained by Mehta (2001), who shows that standard spike-timing dependent plasticity (STDP) on the synapses between place cells can form asymmetric place fields, such that a neuron is highly active prior to arrival at its place field, but inactive immediately afterwards.

Note that place cells are a particularly tacit example of declarative representation in that specific neurons fire when the animal resides in a particular location (X). Phase precession is an equally compelling example of declarative prediction, since the place cell fires on the approach to X. Furthermore, phase precession (and thus the predictions underlying it) may play a much deeper role in cognition, due, in part, to the dual roles of the hippocampus in both navigation and general memory formation.

As shown in Figure 22, as a rodent moves along the corridor from locations A to G, STDP could easily lead to the formation of predictive connections between the detectors for each location, particularly when the intervals between arrival at successive locations are in the range of 0 - 50 msec (the time window for STDP).

![Figure 22](image)

Figure 22: (Top) A corridor along which a mouse runs, with contexts (A - G) encountered approximately every 50 msec. (Bottom) A hypothetical connection pattern, formed via STP, among CA3 detectors for contexts A-G.

Once formed, these synapses can be activated in sequence, with gaps much smaller than 50 msec. Lisman and Redish (2009) have shown that gamma waves (40-100 Hz) elicit this fast replay, with these high-frequency oscillations riding atop the slower (6-10 Hz) theta waves that characterize hippocampal activity. As shown at the top of Figure 23, during one theta cycle, a good many successive context detectors can be activated, each by a gamma peak. At the peak of each theta cycle, the current location determines the active detector, but throughout the remainder of the cycle, succeeding detectors activate in a predictive manner.

This rapid sequencing brings several detector firings within the 0-50 msec window of STDP. So, for example, after cell A fires due to the current sensory input (as shown on the far left of the wave in Figure 23), cells B, C, D and E will fire in rapid succession via gamma stimulation. STDP then dictates that synapses from A to B, C, D and E will all experience LTP, thus forming cell assemblies that manifest information chunking.

The dual role of the hippocampus in both navigation and general memory formation raises the obvious question of whether this place-cell chunking could manifest general information binding and integration. So the aggregated sequences could represent locations, steps in a procedure, or words in a memorable phrase or song melody. In all such cases, the same basic predictive machinery (grounded in the dynamics of STDP) combines with gamma-induced replay (and further STDP) to produce tightly-linked neurons and neural firing patterns which may represent episodes or concepts.
Gamma waves (40-100 Hz) ride atop the theta waves, with each gamma peak stimulating the next place cell in the sequence.

Figure 23: (Left) Gamma waves riding atop theta waves stimulate place cells in rapid succession. (Right) Rapid stimulation facilitates predictive chunking of temporally-related contexts. Stronger connections are drawn with thicker lines.

The leap from place cells to concepts is elaborated by Buzsaki (2006), who begins by describing an interesting property of place-cell learning: when mice move back and forth along a corridor, the same location (depicted as "C" in Figure 25) binds to two different place cells ($C_1$ and $C_2$) depending upon the direction of travel. Hence, to the mouse, these are two different locations. Conversely, in an open arena, without the constraints to movement imposed by corridor walls, the place cells tend to be omnidirectional: the same cell fires, regardless of the angle of approach. Initially, they are unidirectional, as in the corridor, but with continued exploration of the arena, a unique place cell begins to represent the same location, without directional bias.

Dragol et al. (2003) explain this situation with evidence of the continuous mapping and re-mapping of place cells to spatial fields (via LTP): repeated trials in an arena environment could easily stimulate this re-mapping. Thus, many approach episodes become bound to the same place cell, a process fitting of the term generalization.

Consider the neural network in Figure 24, where two different contexts (B and D) originally predict context C, but again, via independent place cells $C_1$ and $C_2$. In time interval T1, context B arises as a prelude to C, thus forming the 4 leftmost connections. During T2, context D arises and hooks up to $C_2$ via the 5 rightmost links. Now, before interval T3, assume that B-to-$C_1$ strengthen due to the repeated occurrence of context B, while context D does not appear at all, thus weakening several D-to-$C_2$ links. Finally, at T3, assume that context D fires. Due to the weakened D-to-$C_2$ links, $C_2$ may not fire, but the shared context (pentagons) between B and D now have enough strength to invoke $C_1$. The co-firing of D and $C_1$ can then strengthen several (previously weak) connections between them. Thus, in the future, D will easily trigger $C_1$. Furthermore, since B and D will both excite $C_1$, their shared context should be co-active with $C_1$ more than will the unshared context of either. Links between the shared context and $C_1$ should then becomeparticularly strong, thus representing the essence of the overlapping contexts. Buzsaki (2006) likens this generalization over experienced episodes to concept formation, wherein the invariants of many specific scenarios are eventually distilled into a general-purpose representation: a concept, which can be metaphorically described as a thought arrived at from many different paths of reasoning.

Since the hippocampus plays a key role in both navigation and memory consolidation (Andersen et al., 2007; McClelland et al., 1994), place cells and their interconnections (modifiable for both look-ahead and chunking) could provide the substrate for both spatial recognition and general concept formation. And in both cases, the predictive links formed by STDP provide a fundamental starting point. So if Buzsaki’s analogy is correct, the hippocampus could be a critical junction between an advanced form of sensorimotor behavior (i.e. navigation), and some of the highest cognitive faculties: abstraction and concept formation.
Figure 24: Emergence of a single detector for a general concept (right) from an earlier group of detectors (left). Pentagons represent neurons encoding shared aspects of contexts B and D, while circles in B and D detect unshared, idiosyncratic details. Arrow thickness denotes relative synaptic strength.

Figure 25: (Left) Unidirectional place-cell formation in a corridor environment. (Right) Transition of unidirectional to omnidirectional place cells in an arena environment. $T_1$ and $T_2$ are distinct temporal intervals, and B, C and D are locations.
As hinted earlier, the connection between physical and mental is believed to go much deeper than correlations between body movements and neural patterns. Cognitive incrementalism (Clark, 2001) - as discussed in the introduction - posits sensorimotor activity as the foundation of higher-level cognitive activity. This view is implicit in AI research (a lot of which involves simple sensorimotor agents) that aspires to AI relevance, since the study and automation of advanced intelligence has little use for wall-following and floor-sweeping robots unless their underlying control mechanisms have implications for cognition.

Many good books have covered this topic in great depth (Clark, 1997, 2011; Pfeifer and Scheier, 1999), and a few include reasonable proposals for the neural basis (Deacon, 1998; Lakoff and Nunez, 2000; Llinas, 2001). Of particular note are Pfeifer and Bongard (2007), who emphasize the role of an efference copy of motor commands in forming a sensorimotor self image. In a nutshell, the premotor cortex is believed to not only send commands to the motor cortex, but also to higher-level cognitive areas, which include this information as part of the current body-world context.

Figure 26: Sketch of three different neural groups (ovals) representing three episodes, all of which contribute to the formation of a concept detector, C. Pentagons denote neurons with a sensory basis, while circles reflect a motor grounding. Solid shapes constitute the essence of the concept, with the majority being sensory, while most unshared neurons have a motor basis. Arrow thickness denotes relative synaptic strength.

Returning to Buzsaki's theory of concept formation via overlapping episodic memories (Buzsaki, 2006), the neural diagram of Figure 26 illustrates a simple role of sensorimotor data in cognition. Imagine three episodes with overlapping sensory and motor components. The concept arising from them stems from the shared neurons, which, in this example, tend to be sensory, not motor. Though many of our procedural memories - such as the movements needed to slam a ping-pong ball - would seem to require very precise recordings of the motor actions, a declarative memory for a concept would seem less tied to motor specifics. For example, my memory of the beautiful view of a river valley from atop a steep hill has only vague connections to the actions that I used to get there, whether walking, running, cycling or driving. My first visits to that spot might have had an episodic motor coupling, but after many visits using many forms of transport, the essential invariant (i.e. the concept) would probably involve the basic impression of being elevated above a wooded area with a river winding through it. A good many declarative concepts would seem to have a similar basis in sensory input. The sensory impression would prevail in later recall of the general location, while each motor efference copy was simply a different mental vehicle for helping to activate neuron (or neural population) C. In organisms with many muscles controlled by the cortex, these efference copies involve many neurons and thus have significant cumulative power to activate context-detectors. Motor neuron activity thus has a bulk effect during the episodic memory formation, but most of this scaffolding has no presence in the final concept.
In general, the task of navigation could be a critical link between sensorimotor activity and cognition, due largely to its demands for recognizing and anticipating spatial locations. These predictions may constitute a key gateway between action and high-level reasoning. As discussed earlier, declarative concept formation may exapt mechanisms for place-cell mediated navigation, namely a) STDP-based predictive linkage, b) gamma-cycle-driven replay of predictive sequences, and c) the consequent chunking, a key element of memory formation. So the predictive sequencing and aggregation underlying navigation may also manifest one of the most advanced cognitive faculties: abstraction.

8 Hebb Still Rules

Unsupervised learning is the most primitive form of adaptivity in ANNs, yet the most prevalent in the brain. Whether cooperative or competitive in nature, the process relies heavily on a fundamental feature of neural systems: Hebbian synaptic modification. This grounding in a simple, local principle makes unsupervised learning very easy to implement in ANNs, but steering emergence toward meaningful global patterns requires careful modeling and parameter selection. Nature has had millions of years to evolve the proper components, which neuroscientists now labor to decipher, and Bio-AI researchers hope to exapt for computational purposes.

The emergent nature of unsupervised neural processing begins at the low level of STDP, continues through the dynamics of recurrent networks, and even appears in the interpretation of ambiguous figures such as the duck/rabbit and vase/kissing-couple staples of psychology textbooks. Behaviors at the level of networks of neurons have a clear local search nature. In the learning phase, different combinations of synaptic weights are tried during the gradual tuning of a complex set of salient-pattern detectors. This matrix of weights (W) is the focal, dynamic representation of learning. During recall, the important dynamic representation is the vector (V) of neural activation levels, which self-organizes to a form that is most compatible with W: a concise, imperfect record of the network’s history.

Whether mice in boxes, ants on beaches, or teenagers in a new mall, the same basic principle holds: trial-and-error exploration combined with Hebbian synaptic tuning leads to representational refinement. Though the ant itself may remember nothing, lay down no mental trace, its pheromone trail serves the greater good of the colony: it constitutes an evolving structure, an extended memory, that enhances group performance. The mouse, on the other hand, internalizes much more of its environment by modifying the synapses onto place cells. Repeated wanderings, driven by little more than random inclinations, support information gathering. Over time, the emerging structures, whether networks of pheromones or interlinked CA3 cells, influence behavior, adding a purposeful or goal-directed appearance to movement. As for teenagers, whatever cannot be offloaded to their extended memory of choice, the cell-phone camera, probably ends up in CA3 as well.

You really can go a long way with the simple Hebbian rule of firing and wiring together. It appears to be the fundamental principle behind associational pattern formation, which, in turn, is touted as one of the premier properties of animal intelligence (Hawkins, 2004). It is also one of the most obvious and impressive characters in the story of intelligence emerging.

References


