Learning and Memory

Quan Wen

25 Dec 2018

Multilayer neural network and supervised learning

As many of you may know, neural network models have been extremely successful in recent years as a tool for performing image recognition, classification, and they even beat the best go player in the world. The strategies for training such networks usually involve supervised learning. The network typically has a feedforward architecture, with one input layer, several hidden layers and one output layer. In an image classification task, the output could be a single number, for example, whether there is a cat in the picture or not. The output is compared to the ground truth, which is labelled by a human being. We can then define an error function

$$E = \frac{1}{2}\sum_k (y_k - t_k)^2, \tag{1}$$

where $\mathbf{t} = [t_1, ..., t_k, ...]^T$ is the target vector. Now if we have a batch of images, the total cost function is the sum in the whole training set:

$$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w}) \tag{2}$$

In Equation 2, N can be viewed as the number of images in a training set. For example, the ImagNet, popularized by Feifei Li, has millions of labelled images. We shall think E as a smooth function of the synaptic weights in the neural net. We can thus map the supervised learning problem to an optimization problem, and one simple way to reduce the cost function is to use gradient descent, so that

$$\mathbf{w}^{\tau+1} = \mathbf{w}^{\tau} - \eta \nabla E(\mathbf{w}^{\tau}) \tag{3}$$

Back Propagation

In the state-of-the-art deep neural network with many different layers, there are millions of synaptic weights. In this specific setting , computing the gradient

involves a special technique, called back propagation. It is not mysterious. Here let's discuss how it works.

In a general feedforward neural network, each neuron computes a weighted sum of its inputs

$$I_j = \sum_i w_{ji} r_i \tag{4}$$

The activity of the postsynaptic unit r_j, is a nonlinear function of the synaptic current

$$r_j = F(I_j) \tag{5}$$

To compute the gradient, let's now look at individual synaptic connection w_{ji}. First we note that the cost function E_n depends on w_{ji} only via the synaptic current in neuron j. By applying the chain rule of the partial derivative, we have

$$\frac{\partial E_n}{\partial w_{ji}} = \frac{\partial E_n}{\partial I_j} \frac{\partial I_j}{\partial w_{ji}}$$

From Equation 4, we have

$$\frac{\partial I_j}{\partial w_{ji}} \equiv r_i.$$

We shall denote

$$\frac{\partial E_n}{\partial I_j} \equiv \delta_j,$$

and

$$\frac{\partial E_n}{\partial w_{ji}} = \delta_j r_i.$$

The meaning of δ_j can be understood immediately for the output unit I_k,

$$\frac{\partial E_n}{\partial I_k} \equiv \delta_k = (y_k - t_k) F'(I_k),$$

which is proportional to the error signal. To compute δ_j in any hidden layer, we again use the chain rule

$$\frac{\partial E_n}{\partial I_j} = \sum_k \frac{\partial E_n}{\partial I_k} \frac{\partial I_k}{\partial I_j}$$

Here the sum is over all the neuron k in the next layer. Moreover, for each neuron in the next layer, we have

$$I_k = \sum_j w_{kj} r_j = \sum_j w_{kj} F(I_j),$$

and therefore

$$\frac{\partial I_k}{\partial I_j} = w_{kj} F'(I_j).$$

Putting everything together, we have the rules of *back propagation*

$$w_{ji} \leftarrow w_{ji} - \eta \delta_j r_i \tag{6}$$

$$\delta_j = F'(I_j) \sum_k \delta_k w_{kj} \tag{7}$$

Now, let's look at the physical meaning of this formula. The synaptic weight would update according to the product of the presynaptic neuron activity i and the error signal in neuron j. However, the error signal in neuron j is computed (or back propagated) from the error signals of all the neurons in the next layer, weighted by the output synaptic strengths from neuron j. This is *biologically implausible*.

Hebbian Rules

The learning rule in the back propagation algorithm is not biologically plausible because there is no conceivable way for the neuron in the current layer to know the error signals from the neurons in the next layer, and to transmit that signal back along the axon. Besides, this must be weighted by the strengths of the synapses between all the neurons in the next layer and the neuron of interest. What is thus the real learning rules in the brain? Hebb wrote:

> When one cell repeatedly assists in firing together, the axon of the first cell develops synaptic knobs (or enlarges them if they already exist) in contact with the soma of the second cell

To formulate Hebb's idea mathematically, we may write

$$\frac{dw_{ji}}{dt} = r_j u_i, \tag{8}$$

where r_j and u_i are the firing rates of postsynaptic and presynaptic neuron respectively. Now let's consider all the presynaptic neurons to postsynaptic neuron j, and denote the weight vector as $\mathbf{w}$. Let's also average over different input and output activity patterns using $\langle ... \rangle$. As a result, we have

$$\tau_w \frac{d\mathbf{w}}{dt} = \langle r\mathbf{u} \rangle \tag{9}$$

In a linear feedforward network model, $r = \mathbf{w^T u} = \mathbf{u^T w}$. Plugging this equation into Equation 9, we found that

$$\tau_w \frac{d\mathbf{w}}{dt} = \mathbf{Q w} \tag{10}$$

$$\mathbf{Q} \equiv \langle \mathbf{u u^T} \rangle \tag{11}$$

Here $\mathbf{Q}$ is an $N \times N$ correlation matrix of the input vector $\mathbf{u}$. The simple Hebb's rule will cause the weight to increase without bound. To see this, we note that

$$\tau_w \frac{d\|\mathbf{w}\|^2}{dt} = 2\tau_w \mathbf{w} \cdot \frac{d\mathbf{w}}{dt} = 2\tau_w \langle r\mathbf{w} \cdot \mathbf{u} \rangle = 2\langle r^2 \rangle > 0$$

LTP and LTD

Synapses can be potentiated and depressed. The Hebb's rule in Equation 9 only leads to potentiation. To describe LTD, for example, we can modify Equation 9 to

$$\tau_w \frac{d\mathbf{w}}{dt} = \langle (r - \theta_r)\mathbf{u} \rangle \tag{12}$$

θ_r is a threshold. If the neural activity is above the threshold, the weight would be potentiated; if the activity is below the threshold, the weight would be depressed. In practice, the threshold is not a constant, which may depend on the activity of the postsynaptic neuron. Let's assume that $\theta_r = \langle r \rangle$, which is the average postsynaptic neural activity. In this case,

$$\tau_w \frac{d\mathbf{w}}{dt} = \langle \mathbf{u}\mathbf{u}^\mathbf{T} - \mathbf{u}\langle \mathbf{u}^\mathbf{T} \rangle \rangle \mathbf{w} \tag{13}$$

By using the equality of the covariance matrix

$$\mathbf{C} \equiv \langle (\mathbf{u} - \langle \mathbf{u} \rangle)(\mathbf{u}^\mathbf{T} - \langle \mathbf{u}^\mathbf{T} \rangle) \rangle \equiv \langle \mathbf{u}(\mathbf{u}^\mathbf{T} - \langle \mathbf{u}^\mathbf{T} \rangle) \rangle \tag{14}$$

we arrive at

$$\tau_w \frac{d\mathbf{w}}{dt} = \mathbf{C}\mathbf{w} \tag{15}$$

In this formulation, the total synaptic weights would still increase without bound. To see this, we note that

$$\tau_w \frac{d\|\mathbf{w}\|^\mathbf{2}}{dt} = 2\tau_w \mathbf{w} \cdot \frac{d\mathbf{w}}{dt} = 2\tau_w \langle (r - \langle r \rangle)\mathbf{w} \cdot \mathbf{u} \rangle = \langle r^2 \rangle - \langle r \rangle^2 > 0$$

Oja rule

A constraint on the sum of the squares of the synaptic weights can be imposed by using a modification of the basic Hebb rule known as the Oja rule,

$$\tau_w \frac{d\mathbf{w}}{dt} = \langle r\mathbf{u} - \alpha r^2 \mathbf{w} \rangle, \tag{16}$$

where α is a positive constant. The stability of the growth of synapses is guaranteed since

$$\tau_w \frac{d\|\mathbf{w}\|^\mathbf{2}}{dt} = 2\tau_w \mathbf{w} \cdot \frac{d\mathbf{w}}{dt} = 2r^2(1 - \alpha\|\mathbf{w}\|^\mathbf{2}) \tag{17}$$

Principle Component Analysis

We now ask what kind of computations the simple Hebb's rule might be able to perform. We shall use Equation 15. As before, $\mathbf{C}\mathbf{e}_\mu = \lambda_\mu \mathbf{e}_\mu$, and because the covariance matrix is symmetric, all the eigenvectors are orthogonal to each other,

and $\mathbf{w}(t) = \sum_\mu c_\mu(t)\mathbf{e}_\mu$, and $c_\mu(t) = c_\mu(0)exp(\lambda_\mu t)$, where $c_\mu(0) = \mathbf{w}(\mathbf{0}) \cdot \mathbf{e}_\mu$. Over a long time, the weight vector would be dominated by the eigenvector of the covariance matrix of the inputs that have the maximum eigenvalue. The firing rate of the output neuron would be proportional to the projection of the input vector to that vector. $r \propto \mathbf{e_1} \cdot \mathbf{u}$.

Long term memory

In a modern computer, memories (in bits) are installed in different physical addresses. However, in our daily life, a recall of the memory item may require a partial or an approximate representation of a stored item. This is called associative memory, which is based on content rather than on an address, or content-addressable memory.

In the last lecture, we have discussed short-term memory models in which information can be stored by means of persistent activity: the eye position can be represented by a stereotyped population activity profile after stimuli onset (in this case, some bursting motor neuron activity from the brain stem). More interestingly, the dynamics in this short-term memory model can be viewed as one-dimensional. This persistent activity idea can be extended to a broader set of different population profiles, which are called memory patterns. Each of these is a fixed point of the dynamics of the network. The memory patterns are determined by and stored within the synaptic weight of the recurrent synaptic weights of the network. To remember something (memory retention) does not require persistent activity patterns; However, to recall the memory requires the persistent activity of a specific population of neurons.

During recall, an associative memory performs the computational operation of pattern matching, finding the memory pattern that most closely matches a distorted or partial activity pattern. This is achieved by initializing the network with an activity profile close to one of the memory patterns, letting it relax to a fixed point, and treating the network activity at the fixed point as the best matching pattern. Next, we shall discuss a famous and elegant mathematical model that instantiate the above ideas, and this is the Hopfield network.

In the Hopfield network, we make even a more drastic assumption that the activity of a neuron is a binary variable, $s_i \in [+1, -1]$. In the deterministic version of this model, the state of a cell is set according to the following equation

$$s_i = \text{sgn}(\sum_j w_{ij} s_j + h_i), \tag{18}$$

where h_i is the external input to a neuron. We consider an asynchronous update rule for each neuron (choosing either a random or fixed update order and update r_i according to that order at each clock cycle). The basic problem setup is that we want to store a number of patterns $\xi^\mu(\mu = 1, ..., P)$ in the

connection matrix of the network. Therefore, we require that ξ^μ should be a stable fixed point of the network. Let us denote that pattern by $\xi^\mu = [\xi_1, ..., \xi_N]$. The condition for ξ^μ to be a fixed point of the network dynamics is:

$$\xi_i = \text{sgn}(\sum_j w_{ij}\xi_j) \tag{19}$$

So that we need a matrix W to satisfy the above equation. Let's choose $w_{ij} = w\xi_i\xi_j$. Clearly the above equation is satisfied. Let's choose the constant of proportionality as $1/N$, and therefore $w_{ij} = \frac{1}{N}\xi_i\xi_j$.

What if we want to store multiple patterns $\xi^\mu(\mu = 1, ..P)$ in the connection matrix of the network. A straight generalization of the synaptic weight matrix suggests the following expression

$$w_{ij} = \frac{1}{N}\sum_{\mu=1}^{P}\xi_i^\mu\xi_j^\mu \tag{20}$$

Why this might work? Let us plug this equation into the updating rule for the Hopfield network, and the fixed point condition for storing memory pattern ξ^ν requires that

$$\xi_i^\nu = \text{sign}\left(\xi_i^\nu + \frac{1}{N}\sum_{j=1}^{N}\sum_{\mu=1\nu\neq\mu}^{P}\xi_i^\mu\xi_j^\mu\xi_j^\nu\right) \tag{21}$$

Note that ξ_i are uncorrelated random variables with mean zero and variance 1. Thus, the last term on the right had side has mean 0 and variance $NP/N^2 = P/N$. In the limit $P \ll N$, this is a term smaller than 1, and ξ^μ is a stable fixed point.

Given the synaptic weight matrix defined above, we have shown that the desired pattern ξ^μ are stable fixed points of the network dynamics in the limit $P \ll N$. However, they are not the only stable fixed points, a lot others exist. A non-trivial one is any mixture of an odd number of memory patterns, such as

$$\xi^{mix} = \text{sign}(\xi^{\mu_1} + \xi^{\mu_2} + \xi^{\mu_3}) \tag{22}$$

To see this, let's plug this equation into the updating rule and we have

$$\sum_j w_{ij}\xi_j^{mix} = \frac{1}{N}\sum_j\sum_{\mu=1}^{P}\xi_i^\mu\xi_j^\mu\xi_j^{mix}$$

$$= \frac{1}{N}\left(\sum_j \xi_i^{\mu_1}\xi_j^{\mu_1}\xi_j^{mix} + \sum_j \xi_i^{\mu_2}\xi_j^{\mu_2}\xi_j^{mix} + \sum_j \xi_i^{\mu_3}\xi_j^{\mu_3}\xi_j^{mix} + \text{crosstalk terms}\right)$$

Looking into this equation, the mean value $\langle \xi_j^{\mu_1} \xi_j^{mix} \rangle$ is $\frac{3}{4}(+1) + \frac{1}{4}(-1) = \frac{1}{2}$. Thus,

$$\sum_j w_{ij} \xi_j^{mix} = \frac{1}{2}(\xi_i^{\mu_1} + \xi_i^{\mu_2} + \xi_i^{\mu_3}) + \text{crosstalk term}$$

Ignoring the crosstalk term (it is small when $P \ll N$), we confirmed that the right hand side has the same sign as ξ_i^{mix}, and this spurious memory pattern is all a stable fixed point.

The error-free recall of a stored pattern of N neurons is $(1 - P_{error})^N \approx 1 - NP_{error}$. the P_{error} for a single neuron can be calculated in the following way.

$$P_{error} = \frac{1}{\sqrt{2\pi}\sigma} \int_1^\infty \exp(-\frac{x^2}{2\sigma^2}) dx = \frac{1}{2}[1 - \text{erf}(\sqrt{N/2P})] \tag{23}$$

Note that in the limit $x \to \infty$, $1 - (erf)(x) \approx \frac{\exp(-x^2)}{\sqrt{\pi}x}$. Now if we requires that $P_{error} < 0.01/N$, we thus obtain

$$\log P_{error} \approx -\log 2 - N/2P - \frac{1}{2}\log \pi - \frac{1}{2}\log(N/2P) < \log 0.01 - \log N$$

Taking the leading order in N, we obtain

$$N/2P > \log N \tag{24}$$

Energy landscape

A more physical way to think about the dynamics in the Hopfield model is to consider an energy landscape, defined by the following function

$$E = -\frac{1}{2}\sum_{i,j} s_i w_{ij} s_j - \sum_i h_i s_i \tag{25}$$

It is easy to show that the update rules reduce the energy E during each iteration. To see this, note that energy difference during one iteration is given by

$$(s_i^* - s_i)\left(\sum_j w_{ij} s_j + h_i\right) \leq 0$$

Now by substituting the expression for the synaptic weight matrix, we have

$$E = -\frac{1}{2N}\sum_{ij} s_i \left(\sum_{\mu=1}^P \xi_i^\mu \xi_j^\mu\right) s_j = -\frac{1}{2N}\sum_{\mu=1}^P \sum_i \xi_i^\mu s_i \sum_j \xi_j^\mu s_j \tag{26}$$

Writing in vector terms, the energy function could be rewritten as

$$E = -\frac{1}{2N}\sum_{\mu=1}^{P}(\xi \cdot \mathbf{s})^2 \tag{27}$$

Generalization of the energy landscape

The idea of energy function can be generalized to the continuous firing rate model, as discussed previously. Let's consider the following equation

$$\tau_r \frac{d\mathbf{v}}{dt} = -\mathbf{v} + F(\mathbf{h} + \mathbf{M} \cdot \mathbf{v}), \tag{28}$$

where F, the activation function, specifies the relationship between steady state firing rate and the total synaptic input current. In the following, we shall show that such an energy function does exist under certain conditions. To this end, let us define

$$\mathbf{s} = F(\mathbf{h} + \mathbf{M} \cdot \mathbf{v}),$$

and the inverse function for F as F^{-1}. We are using a similar notation as that for the Hopfield network. The energy function is defined as

$$\begin{aligned} E = -\frac{1}{2}\sum_{ij} s_i M_{ij} s_j - \sum_i h_i s_i + \sum_i V(s_i), \\ V = \int_0^{s_i} F^{-1}(x)dx \end{aligned} \tag{29}$$

Next we shall prove that for symmetric and nonsingular weight matrix M, $\frac{dE}{dt} \leq 0$ and it is bounded from below for a biologically plausible activation function F. To show this, note that

$$\frac{dE}{dt} = \sum_i \frac{\partial E}{\partial s_i}\frac{\partial s_i}{\partial t} + \sum_j \frac{\partial E}{\partial s_j}\frac{\partial s_j}{\partial t}$$

Substituting the expression for E, we have

$$\frac{dE}{dt} = -\frac{1}{2}\left(\sum_i\sum_j M_{ij}s_j\frac{\partial s_i}{\partial t} + \sum_j\sum_i s_i M_{ij}\frac{\partial s_j}{\partial t}\right) - \sum_i h_i\frac{\partial s_i}{\partial t} + \sum_i F^{-1}(s_i)\frac{\partial s_i}{\partial t}$$

If the synaptic weight matrix is symmetric, $M_{ij} = M_{ji}$, the above equation is further simplified. Writing it down in the matrix form, we have

$$\frac{dE}{dt} = \left[-\mathbf{M}\mathbf{s} - \mathbf{h} + F^{-1}(\mathbf{s})\right] \cdot \frac{d\mathbf{s}}{dt}$$

We next derive the explicit expression for $\frac{d\mathbf{s}}{dt}$.

$$\frac{d\mathbf{s}}{dt} = F'\mathbf{M}\frac{d\mathbf{v}}{dt}$$

We can further express $\mathbf{v}$ in terms of $\mathbf{s}$ using the inverse function F^{-1}:

$$\mathbf{v} = \mathbf{M}^{-1}[F^{-1}(\mathbf{s}) - \mathbf{h}],$$

And

$$\tau_r \frac{d\mathbf{v}}{dt} = -\mathbf{v} + \mathbf{s} = -\mathbf{M}^{-1}[F^{-1}(\mathbf{s}) - \mathbf{h}] + \mathbf{s}.$$

Thus,

$$\tau_r \frac{d\mathbf{s}}{dt} = F'[-F^{-1}(\mathbf{s}) + \mathbf{h} + \mathbf{M}\mathbf{s}]$$

As a result, the gradient of the energy function is given by

$$\frac{dE}{dt} = -\frac{1}{\tau_r} F' \|\mathbf{h} + \mathbf{M}\mathbf{s} - F^{-1}(\mathbf{s})\|^2 \leq 0 \tag{30}$$

The energy function would be bounded from below if $\mathbf{s}$ does not diverge. Indeed, this is a biologically plausible assumption. No neurons can have excessively high firing rate. In practice, F has been chosen as a sigmoidal function so that its value saturates for high synaptic currents.