

Learning trajectories by gradient descent

Sebastian Seung

9.641 Lecture 21: November 27, 2000

1 Backpropagation learning of trajectories

The backpropagation algorithm for learning trajectories is the following iterative procedure:

- Starting from the initial condition $x(0)$, integrate the network

$$\dot{x} + x = f(Wx + b) \quad (1)$$

forward in time to $t = T$, all the while keeping track of the error $d - x$ between the desired and actual trajectories, as well as $D(t) = \text{diag}(f'(Wx(t) + b))$.

- Starting from the final condition $y(T) = 0$, integrate the dual network

$$-\dot{y} + y = W^T D y + d - x \quad (2)$$

backward in time to $t = 0$.

- Make the updates

$$\Delta W = \eta \int_0^T dt D y x^T \quad (3)$$

$$\Delta b = \eta \int_0^T dt D y \quad (4)$$

We will show later that this procedure performs gradient descent on the cost function

$$E = \int_0^T dt \frac{1}{2} |x - d|^2$$

subject to the constraint (1).

Backprop can deal with hidden neurons. If there is no desired value for a neuron during some time interval, then the error term for that neuron is omitted from (2) during that time interval. The initial conditions of the hidden neurons can be held fixed, or they can also be optimized in the algorithm.

Note that backprop solves the problem of temporal credit assignment by going backwards in time. It is necessary to go backwards because of the final condition $y(T) = 0$. It is also possible to integrate y forward in time, but then it would be necessary to iterate to satisfy the final condition.

Unfolding the network in time.
Teacher-forcing.

2 Lagrange multiplier derivation

Consider the functional

$$S[x(t), y(t)] = \int_0^T dt \left\{ \frac{1}{2} |x - d|^2 + y^T [\dot{x} + x - f(Wx + b)] \right\}$$

If $x(t)$ and $y(t)$ are changed by small amounts $\delta x(t)$ and $\delta y(t)$, then the change in S is

$$\delta S = \int_0^T dt \left[\frac{\delta S}{\delta x(t)} \delta x(t) + \frac{\delta S}{\delta y(t)} \delta y(t) \right]$$

by definition of the functional derivatives $\delta S/\delta x(t)$ and $\delta S/\delta y(t)$.

Let's impose the boundary condition $\delta x(0) = 0$, so that we consider only trajectories $x(t)$ with the same starting point. Then we say that S is stationary with respect to variations in x and y if $\delta S = 0$, i.e., $\delta S/\delta x(t) = 0$ for all $t > 0$ and $\delta S/\delta y(t) = 0$ for all t .

From

$$\frac{\delta S}{\delta y(t)} = \dot{x} + x - f(Wx + b)$$

it follows that $x(t)$ is a trajectory of (1) if S is stationary. Furthermore, $S = E$ at a stationary point. Therefore, the problem of minimizing E subject to the constraint (1) is equivalent to minimizing the value of S at a stationary point.

Let's calculate the functional derivative $\delta S/\delta x$. The variation δS caused by a small change δx is

$$\delta S = \int_0^T dt [(x - d)^T \delta x + y^T \delta \dot{x} + y^T (\delta x - DW \delta x)]$$

Integrating by parts, we find

$$\delta S = \int_0^T dt [(x - d)^T \delta x - \dot{y}^T \delta x + (y^T - y^T DW) \delta x] + [y^T \delta x]_0^T \quad (5)$$

$$= \int_0^T dt (x - d - \dot{y} + y - W^T Dy)^T \delta x + y^T \delta x]_0^T \quad (6)$$

Therefore the functional derivative is

$$\frac{\delta S}{\delta x(t)} = x - d - \dot{y} + y - W^T Dy + y + y(T) \delta(t - T) - y(0) \delta(t)$$

If we choose y such that

$$-\dot{y} + y = W^T D y + d - x$$

and impose the final condition $y(T) = 0$, then $\delta S / \delta x(t) = 0$ for $t > 0$.

Now we see that the forward and backward pass place us at a stationary point of S . Since the functional derivatives of S are zero, the gradient of S with respect to W and b includes only the explicit dependence, and not the implicit dependence on the trajectories:

$$\Delta W = -\eta \frac{\partial S}{\partial W} = \eta \int_0^T dt D y x^T$$

3 Real-time recurrent learning

The real-time recurrent learning (RTRL) algorithm involves integration of the equations

$$\frac{dx_k}{dt} + x_k = f \left(\sum_l W_{kl} x_l + b_k \right) \quad (7)$$

$$\frac{dp_{ij}^k}{dt} + p_{ij}^k = f'(\hat{x}_k) \left[\sum_l W_{kl} p_{ij}^l + \delta_{ik} x_j \right] \quad (8)$$

$$\frac{dW_{ij}}{dt} = \eta \sum_k (d_k - x_k) p_{ij}^k \quad (9)$$

with the initial condition $p_{ij}^l(0) = 0$ and the definition $\hat{x}_k \equiv \sum_l W_{kl} x_l + b_k$.

The problem of temporal credit assignment is solved with the variables p_{ij}^k , which measure the effect of W_{ij} on x_k . No backward pass is necessary.

4 Derivation

Gradient descent on the squared error

$$E = \int_0^T dt \sum_k \frac{1}{2} (d_k - x_k)^2$$

is

$$\Delta W_{ij} = -\eta \frac{\partial E}{\partial W_{ij}} \quad (10)$$

$$= \eta \int_0^T dt \sum_k (d_k - x_k) \frac{\partial x_k}{\partial W_{ij}} \quad (11)$$

To calculate $\partial x_k / \partial W_{ij}$, differentiate the equation

$$\frac{dx_k}{dt} + x_k = f \left(\sum_l W_{kl} x_l + b_k \right)$$

to obtain

$$\frac{d}{dt} \frac{\partial x_k}{\partial W_{ij}} + \frac{\partial x_k}{\partial W_{ij}} = f'(\hat{x}_k) \sum_l \left[W_{kl} \frac{\partial x_l}{\partial W_{ij}} + \delta_{ik} \delta_{jl} x_l \right] \quad (12)$$

$$= f'(\hat{x}_k) \left[\sum_l W_{kl} \frac{\partial x_l}{\partial W_{ij}} + \delta_{ik} x_j \right] \quad (13)$$

Assuming $x_k(0)$ is fixed, and therefore does not depend on the synaptic weights, then $\partial x_k / \partial W_{ij} = 0$. Therefore we can compute $\partial x_k / \partial W_{ij}$ by the above differential equation for p_{ij}^k with the initial condition $p_{ij}^k = 0$.

Gradient descent is

$$\delta W_{ij} = \eta \int_0^T dt \sum_k (d_k - x_k) p_{ij}^k$$

This is the batch form of learning. But it is also possible to make the weight changes online while the trajectory is being computed.