These notes cover the following topics:

1. Brief review of back-propagation in DNN (BP)

2. a) Back-propagation through time (BPTT) in RNN.
   b) Vanishing gradient problem in RNN.

3. A short introduction of LSTM.

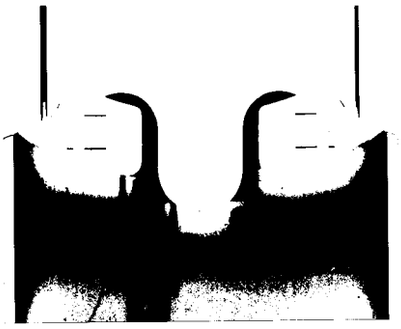
References:

[1] Bishop, "Neural networks for pattern recognition," 1995, Oxford University Press

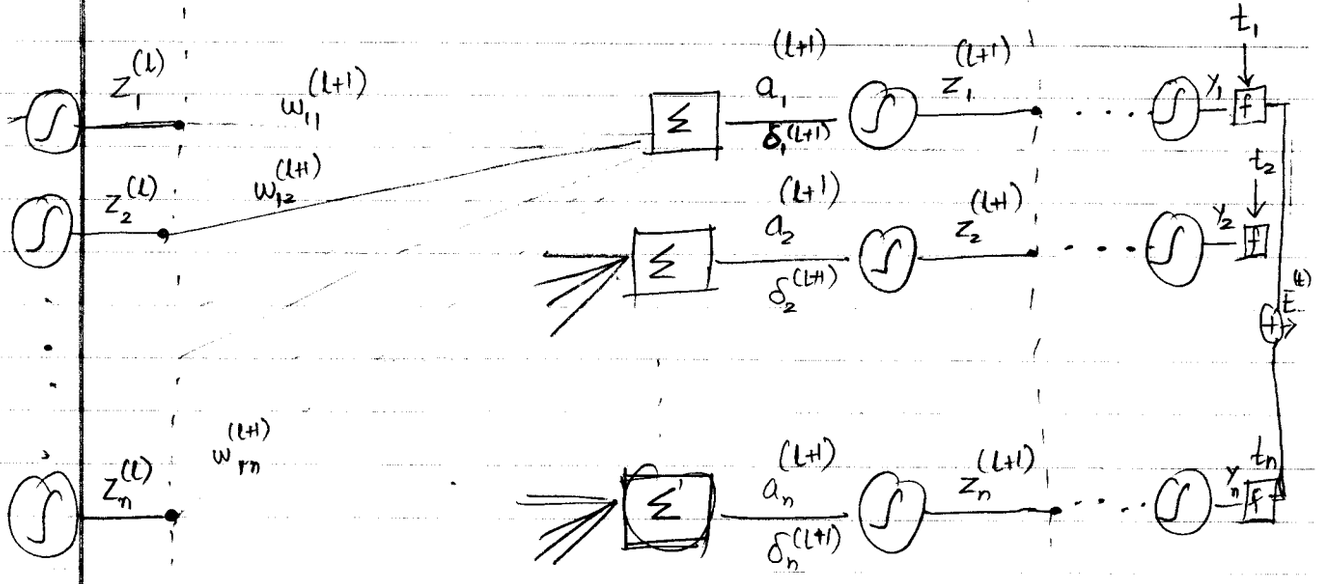
[2] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp 1735-1780, 1997

[3] A. Graves, A. R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," ICASSP 2013.

[4] H. Sak, A. Senior, F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," ArXiv-eprints, Feb 2014.



Bishop's notation: l (layer), $l+1$ (layer)



$$a_j^{(l+1)} = \sum_i w_{ji}^{(l+1)} z_i^{(l)} \qquad (w_{ji}: j = \text{o/p node}, i = \text{i/p node})$$

$$z_j^{(l+1)} = g\left(a_j^{(l+1)}\right) = g\left(\sum_i w_{ji}^{(l+1)} z_i^{(l)}\right)$$

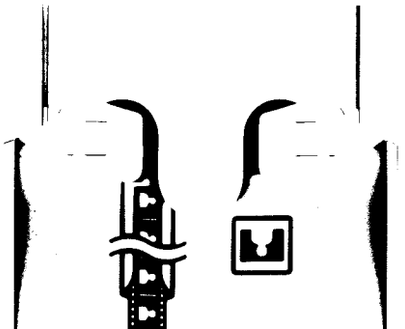
$$E_{tot} = \sum_{t=1}^{T} E^{(t)} \qquad (T = \#\text{ training examples})$$

$$\frac{\partial E^{(t)}}{\partial w_{ji}^{(l)}} = \frac{\partial E^{(t)}}{\partial z_j^{(l)}} \frac{\partial z_j^{(l)}}{\partial a_j^{(l)}} \frac{\partial a_j^{(l)}}{\partial w_{ji}^{(l)}}$$

$$= \underbrace{\delta_j^{(l)}}\; z_i^{(l-1)}$$

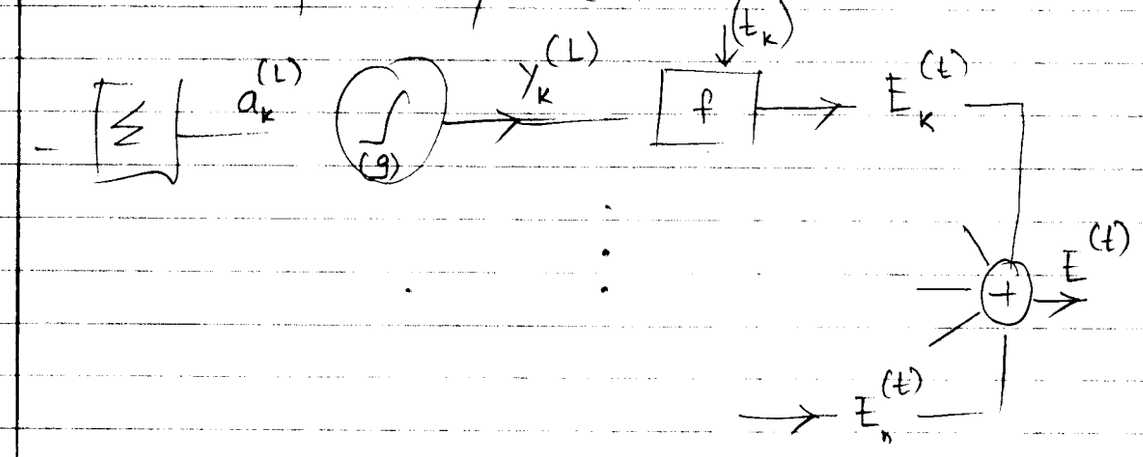
$$\frac{\partial E^{(t)}}{\partial w_{ji}^{(l)}} = \delta_j^{(l)} z_i^{(l-1)} \quad \text{①} \qquad \left[\text{where } \delta_j^{(l)} \triangleq \frac{\partial E^{(t)}}{\partial a_j^{(l)}}\right]$$

$z_i^{(l-1)}$: forward pass calculations, $\delta_j^{(l)}$: backward pass calculations



Calculate $\delta_j^{(l)}$:

Case I: Start with final layer (L): $\therefore l = L$



$$E^{(t)} = \sum_{k=1}^{n} E_k^{(t)} = \sum_{k=1}^{n} f\left(y_k^{(L)}, t_k\right) \quad -(2)$$

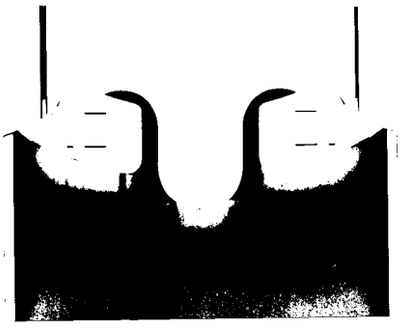
$$\delta_k^{(L)} = \frac{\partial E^{(t)}}{\partial a_k^{(L)}} = \frac{\partial E^{(t)}}{\partial y_k^{(L)}} \frac{\partial y_k^{(L)}}{\partial a_k^{(L)}} \quad -(3)$$

$$y_k^{(L)} = g\left(a_k^{(L)}\right) \Rightarrow \frac{\partial y_k^{(L)}}{\partial a_k^{(L)}} = g'\left(a_k^{(L)}\right) \quad -(4)$$

$$\frac{\partial E^{(t)}}{\partial y_k^{(L)}} = \frac{\partial f\left(y_k^{(L)}, t_k\right)}{\partial y_k^{(L)}} \quad -(5)$$

If $f\left(y_k^{(L)}, t_k\right) = \frac{1}{2}\left(y_k^{(L)} - t_k\right)^2$

$$\frac{\partial E}{\partial y_k^{(L)}} = 2\left(y_k - t_k\right)$$

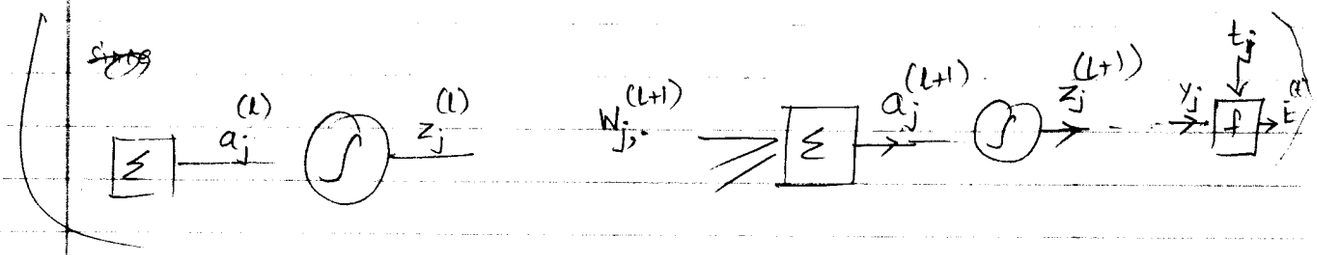


Substituting (4) & (5) in (3),

$$\delta_k^{(L)} = g'(a_k^{(L)}) \frac{\partial E^{(t)}}{\partial y_k^{(L)}} \quad - (6)$$

Case II For the remaining layers $(0 < l < L)$

$$\delta_j^{(l)} = \frac{\partial E^{(t)}}{\partial a_j^{(l)}} = \sum_{k=1}^{n} \frac{\partial E^{(t)}}{\partial a_k^{(l+1)}} \frac{\partial a_k^{(l+1)}}{\partial a_j^{(l)}} \quad - (7)$$



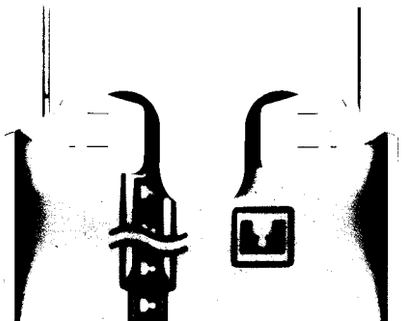
$$\frac{\partial E^{(t)}}{\partial a_j^{(l+1)}} = \delta_j^{(l+1)} \quad \text{(by definition)} \quad - (8)$$

We only need to compute $\frac{\partial a_k^{(l+1)}}{\partial a_j^{(l)}}$.

$$a_k^{(l+1)} = \sum_j w_{kj}^{(l+1)} z_j^{(l)}$$

$$= \sum_j w_{kj}^{(l+1)} g(a_j^{(l)})$$

For a particular j, $\frac{\partial a_k^{(l+1)}}{\partial a_j^{(l)}} = w_{kj}^{(l+1)} g'(a_j^{(l)}) \quad - (9)$



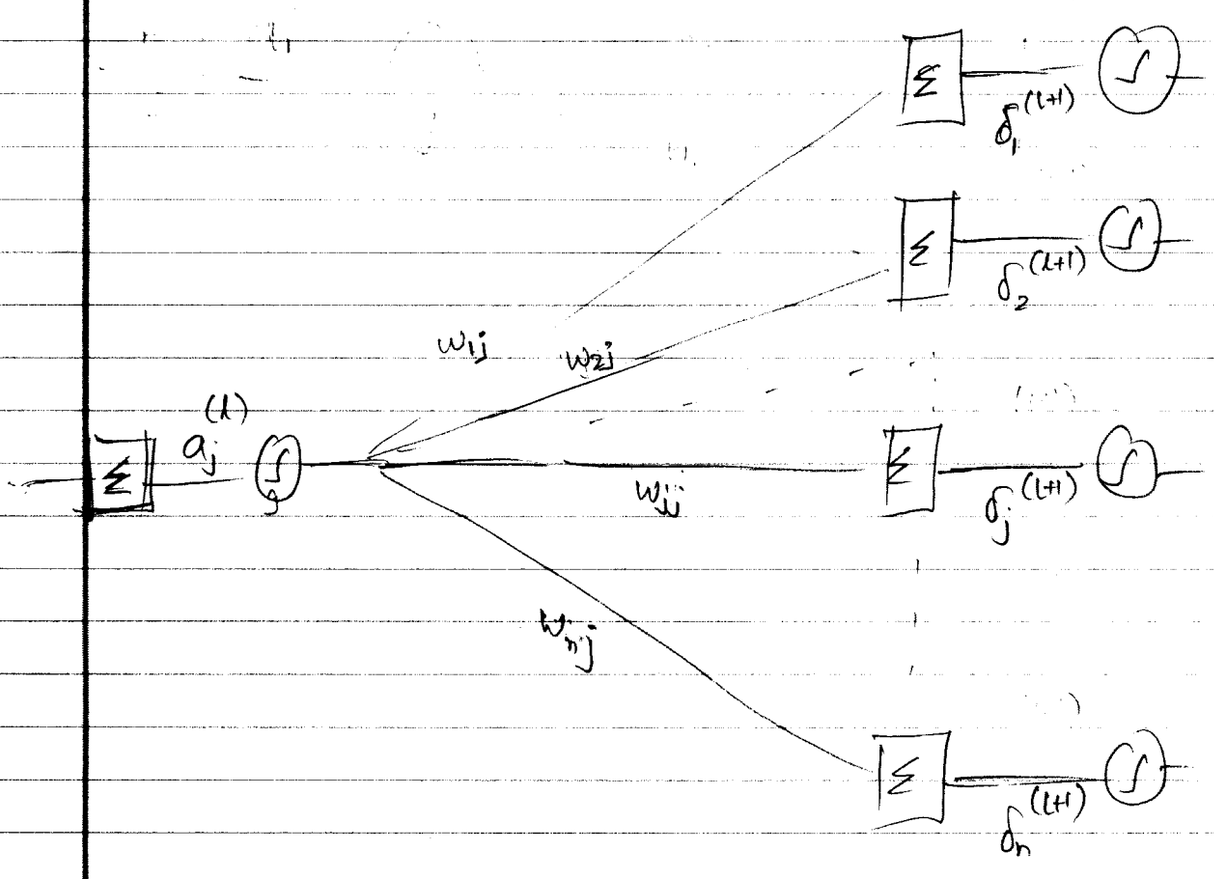
Substitute (8) and (9) in (7),

$$\delta_j^{(l)} = \sum_{k=1}^{n} \delta_k^{(l+1)} \; w_{kj}^{(l+1)} \; g'(a_j^{(l)})$$

$$\delta_j^{(l)} = g'(a_j^{(l)}) \sum_{k=1}^{n} w_{kj}^{(l+1)} \; \delta_k^{(l+1)} \quad - (10)$$

(Back-propagation equation)

Back-prop diagram



From (10) (back prop), we know $\delta_j^{(l)}$ $\forall j, l$,

From forward pass, we know $z_j^{(l-1)}$

$$\frac{\partial E^{(t)}}{\partial w_{ji}^{(l)}} = z_i^{(l-1)} \delta_j^{(l)} \quad \text{(Repeat eqn. (11))}$$

$$\frac{\partial E}{\partial w_{ji}^{(l)}} = \frac{1}{T} \sum_{t=1}^{T} \frac{\partial E^{(t)}}{\partial w_{ji}^{(l)}} \quad -(11)$$

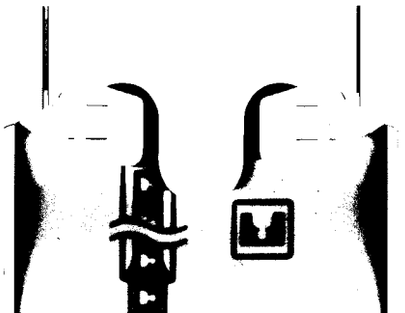
(12) (Gradient descent) $(\eta > 0)$

$$w_{ji}^{(l)} \leftarrow w_{ji}^{(l)} - \eta \frac{\partial E}{\partial w_{ji}^{(l)}}$$

$$= w_{ji}^{(l)} - \frac{\eta}{T} \sum_{t=1}^{T} \frac{\partial E^{(t)}}{\partial w_{ji}^{(l)}}$$

$$= w_{ji}^{(l)} - \frac{\eta}{T} \sum_{t=1}^{T} \left[z_i^{(l-1)} \delta_j^{(l)} \right]^{(t)}$$

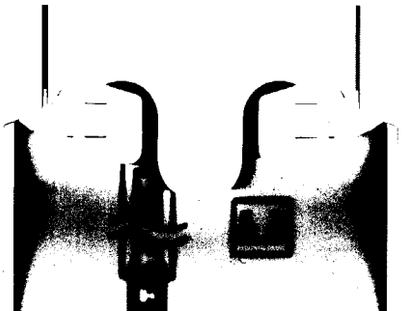
eqn. (1) for a pattern 't'.



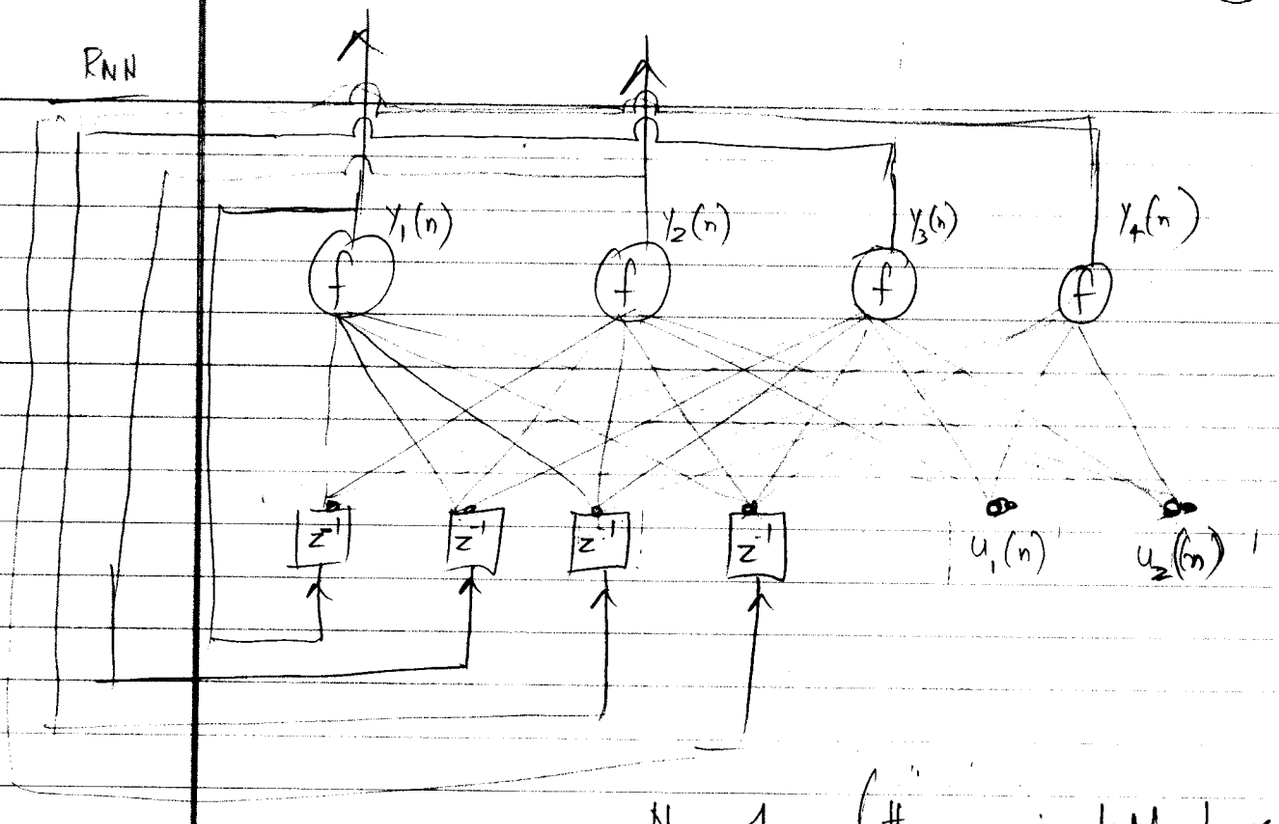
Recurrent Neural Networks

Notations:

	DNN (Bishop)	RNN	
o/p of node 'i' last layer:	y_i	$y_i(n)$	n = at time n.
i/p of 0th layer	$z_i^{(0)}$	$u_i(n)$	
Summed input	a_i	$s_i(n)$	
o/p node 'i' i/p node 'j'	w_{ij}	w_{ij}	
Act function	$g(\cdot)$	$f(\cdot)$	



RNN



$N = 4$ (# neurons in hidden layer)

$M = 2$ (external i/p s, u_1, u_2)

$L = 2$ (# of o/p)

Input

$$z_i(n) = \begin{cases} y_i(n-1), & 1 \le i \le N \\ u_{i-N}(n), & N+1 \le i \le N+M \end{cases} \quad -(1)$$

Summed Input

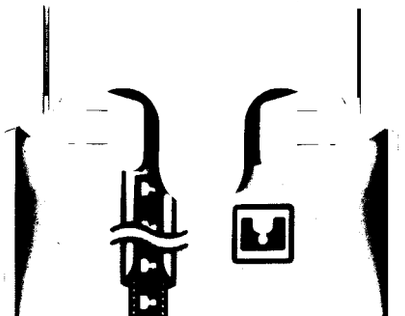
$$s_i(n) = \sum_{j=1}^{N+M} w_{ij}\, z_j(n) \quad -(2)$$

$$= \sum_{j=1}^{N} w_{ij}\, y_j(n-1) + \sum_{j=N+1}^{N+M} w_{ij}\, u_{j-N}(n)$$

Output

$$y_i(n) = f(s_i(n)) \quad -(3)$$

(or there could be another non-recurrent layer of affine xforms & sigmoid fns)



$$E[n_0, n] = \sum_{m=n_0}^{n} \sum_{k=1}^{L} e_k^2(m)$$

In DNN, we had $E^{(t)}$ for 1 training pattern which occurs at time t.

In RNN, 1 training pattern occurs during time $[n_0, n]$.

$$E[n_0, n] = \sum_{t=n_0}^{n} E^{(t)}$$

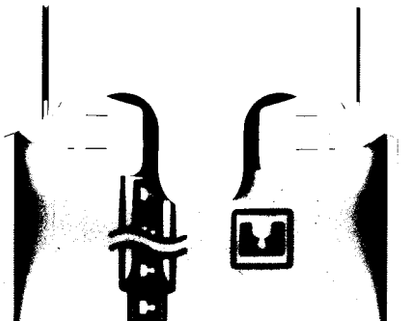
To calculate the δ's, we need to unfold the RNN in time. The unfolded RNN becomes a DNN with

layer 1 corresp. to n_0

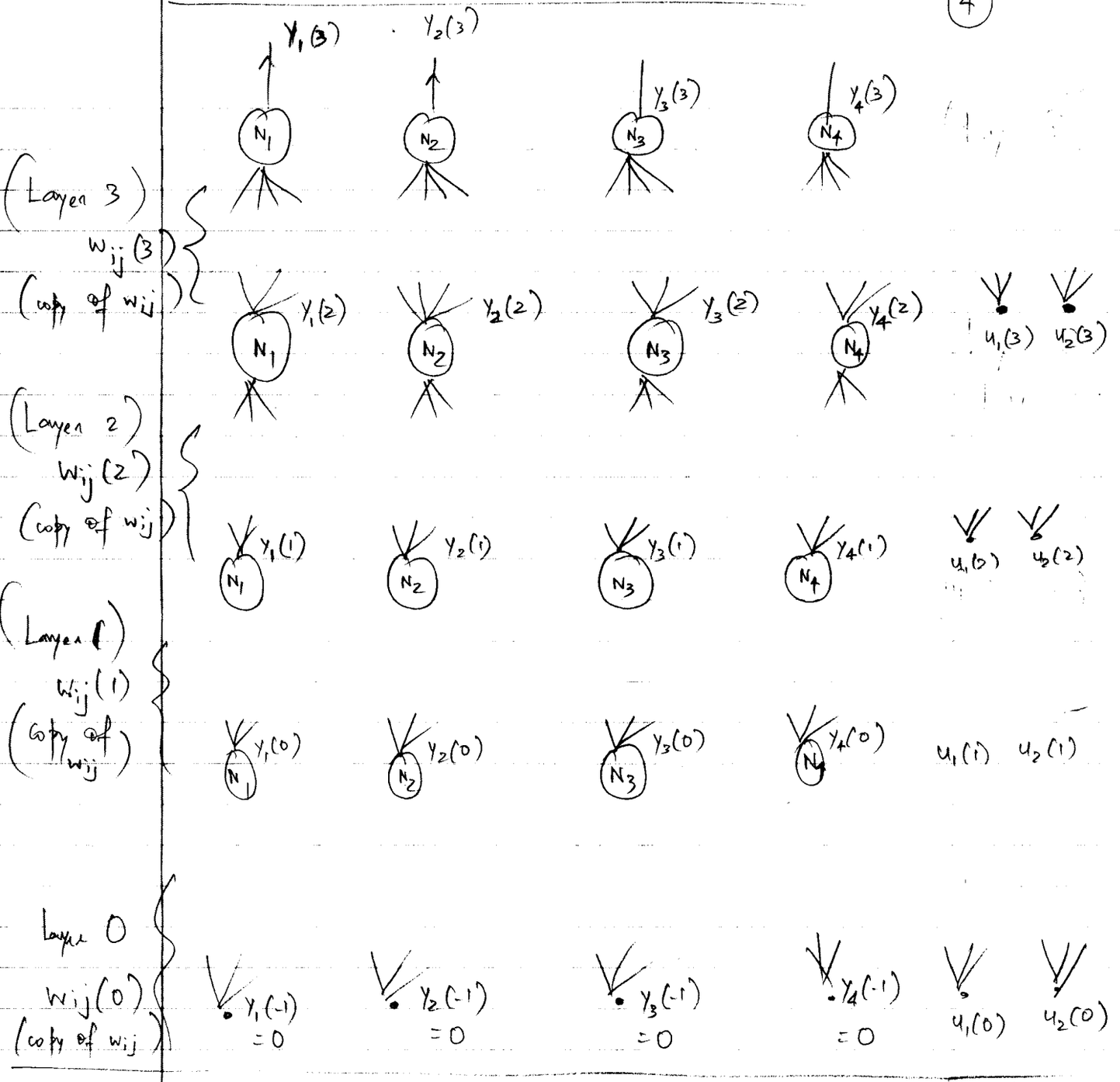
layer 2 — " — $n_0 + 1$

⋮

layer $(n - n_0 + 1)$ — " — n



Unfolded RNN in time [0, 3]



Input

$$z_i(n) = \begin{cases} y_i(n-1), & 1 \le i \le N \\ u_{i-N}(n), & N+1 \le i \le N+M \end{cases} \quad -(4)$$

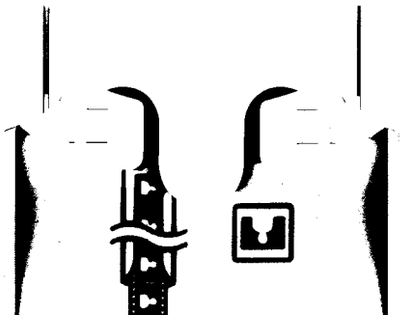
Summed I/P

$$s_i(n) = \sum_{j=1}^{N+M} w_{ij}(n)\, z_j(n) \quad -(5)$$

Output

$$y_i(n) = f(s_i(n)) \quad -(6)$$

(Or add another layer of affine xform & sigmoid)



Q: Why unfolding RNN?

Ans: Because easy to calculate δ's. Instead of $\delta_j^{(l+1)}$ (in DNN), we have $\delta_j(t+1)$ in unfolded RNN. Then calculating $\delta_j^{(t+1)}$ is easy using BPTT (back-prop through time).

Rewriting the summed i/p equations for RNN & unfolded RNN,

(RNN)
$$s_i(m) = \sum_{j=1}^{N+M} w_{ij}\, z_j(m) \quad - (3)$$

(Unfolded RNN)
$$s_i(m) = \sum_{j=1}^{N+M} w_{ij}(m)\, z_j(m) \quad - (5)$$

(3) = (5) since $w_{ij}(m) = w_{ij}$, $m \in [n_0, n_1]$

During fwd pass, w_{ij}'s are fixed. So, we get the same o/p in RNN & unfolded RNN for a given set of i/p

During back-prop, things are different for each layer

$$\frac{\partial E(n_0, n)}{\partial w_{ij}(m)} \neq \frac{\partial E(n_0, n)}{\partial w_{ij}(l)} \qquad m \neq l$$

since $$\frac{\partial E(n_0, n)}{\partial a_i(m)} \neq \frac{\partial E(n_0, n)}{\partial a_i(l)}$$

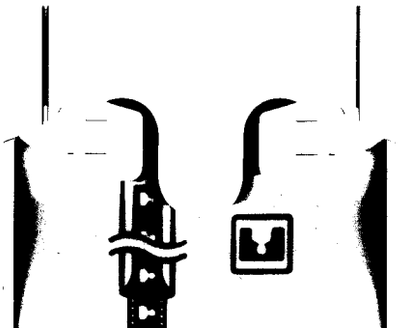
$$\frac{\partial E(n_0, n)}{\partial w_{ij}} = \text{Error gradient of RNN}$$

$$= \sum_{m=n_0}^{n} \frac{\partial E(n_0, n)}{\partial w_{ij}(m)} \underbrace{\frac{\partial w_{ij}(m)}{\partial w_{ij}}}_{=1 \text{ (since } w_{ij}(m) = w_{ij})}$$

$$= \sum_{m=n_0}^{n} \frac{\partial E(n_0, n)}{\partial w_{ij}(m)}$$

$$\boxed{\frac{\partial E(n_0, n)}{\partial w_{ij}} = \sum_{m=n_0}^{n} \delta_i^{(m)} z_j^{(m)}, \quad 1 \le i, j \le N+M} \quad (6)$$

(BPTT equation) (similar to BP but additionally calculated through all layers)



Qualitative Explanation

In RNN, $\frac{\partial E(n_0, n)}{\partial w_{ij}}$ = 'change' in $E(n_0)$ + $E(n_1)$ + ... $E(n)$ when you perturb w_{ij}.

∴ Need to calculate ΔE for each time step.

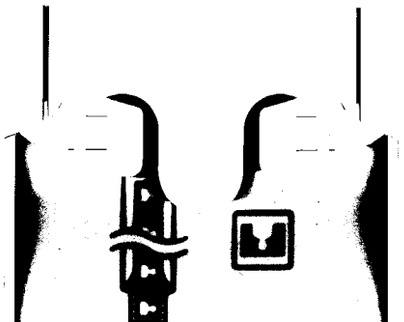
In unfolded RNN, when we perturb w_{ij}, we are perturbing $w_{ij}(m)$, $m = n_0, \ldots, n$ by the same amount. ∴ $\frac{\partial w_{ij}(m)}{\partial w_{ij}} = 1$

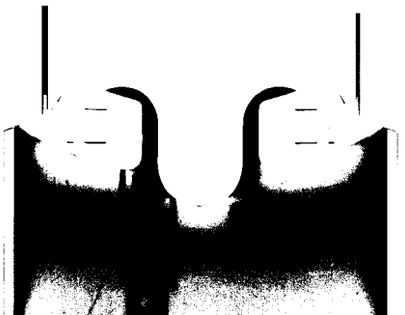
But to calculate change in $E(n)$, we need to compute $\frac{\partial E(n)}{\partial w_{ij}(n)}$ & $\frac{\partial E(n)}{\partial s_i(n)} = \delta_i(n)$

Similarly, to compute change in $E(n-1)$, we need $\frac{\partial E(n-1)}{\partial w_{ij}(n-1)}$ & $\frac{\partial E(n-1)}{\partial s_i(n-1)} = \delta_i(n-1)$

And, so on.

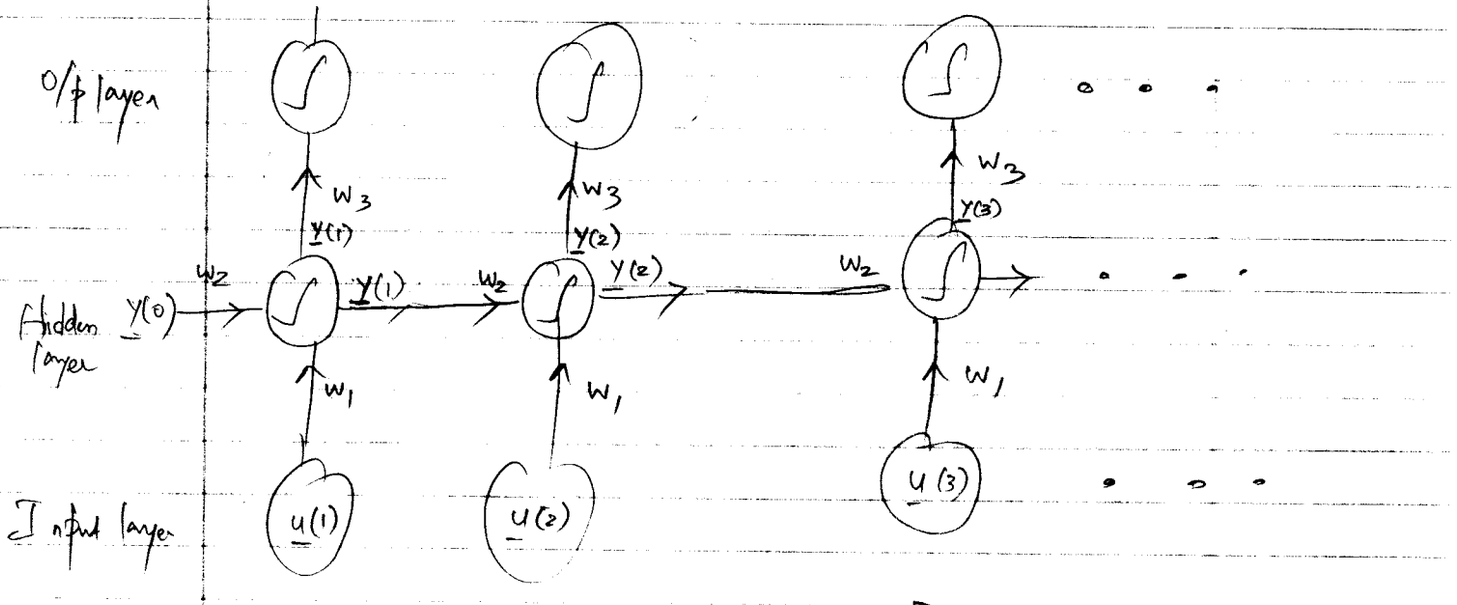
Finally we add all $\delta_i(m)$ from $m = n_0, \ldots, n$.



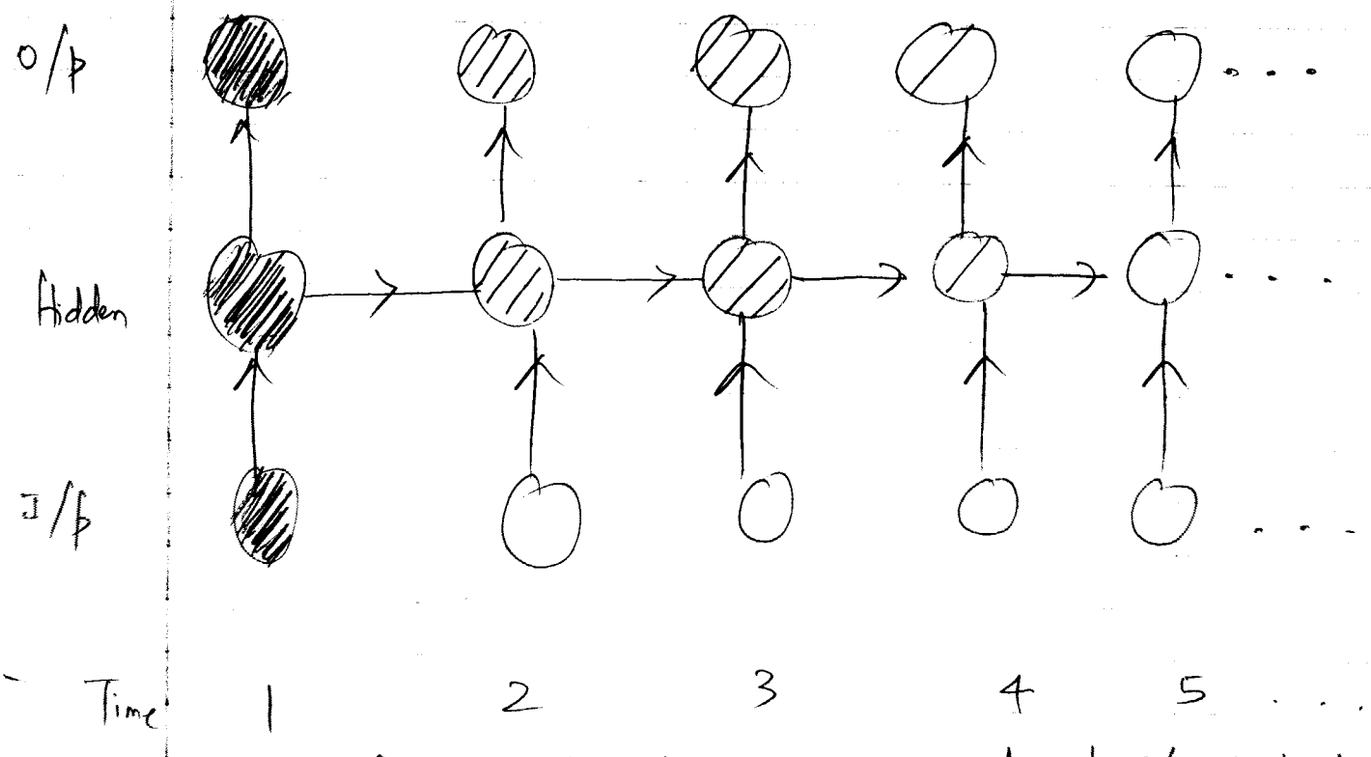


RNN: problem of vanishing (or explosive) gradients

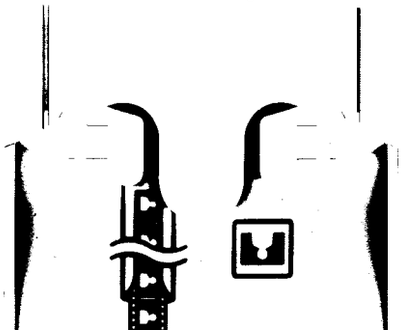
Another view of unfolded RNN:



Vanishing Gradient (Darker ⟺ More sensitive)



Sensitivity of hidden & o/p layer activations due to i/p at $t=1$
↓ as time ↑. N/w forgets i/p at $t=1$.



Recall, back-prop eqn:,

$$\delta_j(t-1) = f_j'\left(s_j(t-1)\right) \sum_i w_{ij}\, \delta_i(t) \quad - (7)$$

$$\frac{\partial E}{\partial w_{ij}(t-1)} \propto \eta\, \delta_j(t-1)\, y_i(t-2)$$

η ~ learning rate

∴ Differentiate (7) w.r.t. $\delta_i(t)$, (t = final time)

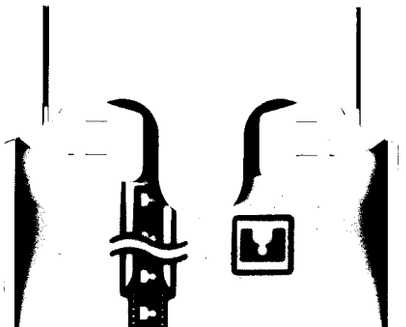
$$\frac{\partial \delta_j(t-1)}{\partial \delta_i(t)} = f_j'\left(s_j(t-1)\right) w_{ij} \quad - (8)$$

The back-prop eqn. to compute $\delta_j(t-q)$, i.e. 'q' time steps behind is,

$$\delta_j(t-q) = f_j'\left(s_j(t-q)\right) \sum_i w_{ij}\, \delta_i(t-q+1) \quad - (9)$$

∴ Differentiate (9) w.r.t. $\delta_i(t)$, (t = final time)

$$\frac{\partial \delta_j(t-q)}{\delta_i(t)} = f_j'\left(s_j(t-q)\right) \sum_i w_{ij} \frac{\partial \delta_i(t-q+1)}{\partial \delta_i(t)} \quad - (10)$$



$\therefore$ Gradient of the errors can be computed using the back-prop method (as in (10))

$\therefore$ Putting (8) and (10) together we have (11)

$$\frac{\partial \delta_j(t-q)}{\partial \delta_i(t)} = \begin{cases} f_j'\left(s_j(t-1)\right) w_{ij}, & q=1 \\ f_j'\left(s_j(t-q)\right) \sum_i w_{ij} \dfrac{\partial \delta_i(t-q+1)}{\partial \delta_i(t)}, & q>1 \end{cases} \quad -(11)$$

For $q=2$,

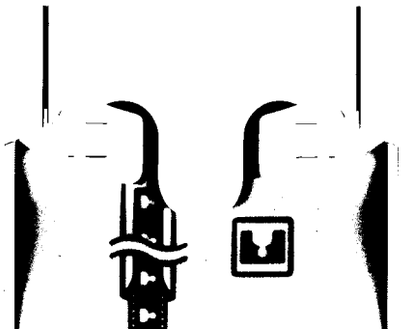
$$\frac{\partial \delta_j(t-2)}{\partial \delta_i(t)} = \sum_k f_k'\left(s_k(t-1)\right) w_{ik} \, f_j'\left(s_j(t-2)\right) w_{kj}$$

Replace 'i' by l_0 and 'j' by l_2, 'k' by l_1

$$\frac{\partial \delta_j(t-2)}{\partial \delta_i(t)} = \sum_{l_1} f_{l_1}'\left(s_{l_1}(t-1)\right) w_{l_0 l_1} \, f_{l_2}'\left(s_{l_2}(t-2)\right) w_{l_1 l_2}$$

From this, we can write the back-prop for 'q' time steps behind as,

$$\frac{\partial \delta_j(t-q)}{\partial \delta_i(t)} = \sum_{l_1=1}^{\#\text{neurons}} \sum_{l_2=1}^{\#\text{neurons}} \cdots \sum_{l_{q-1}}^{\#\text{neurons}} \prod_{m=1}^{q} f_{l_m}'\left(s_{l_m}(t-m)\right) w_{l_{m-1} l_m} \quad -(12)$$



Therefore (12) looks like :

$$\underbrace{(+ \quad + \quad \cdots \quad +)}_{n \text{ terms}} \underbrace{(+ \quad \cdots \quad +)}_{\text{next } n \text{ terms}} \cdots \quad (q-1) \text{ times}$$

$\therefore$ There are totally $\underbrace{n \cdot n \cdot \cdots}_{(q-1) \text{ times}} = n^{q-1}$ terms

and each term of those n^{q-1} terms look like this:

$$\boxed{f'(\ )\ w_{ij}}$$

$$\therefore \quad (f_j' w_{ij} + f_k' w_{ik} + \cdots)(f_{(\cdot)}' w_{(\cdot)} + \cdots) \cdots$$

After multiplying, we have

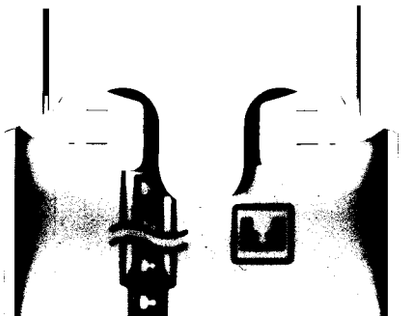
$$(12) = \underbrace{f_{(\cdot)}' w_{(\cdot)}}_{1} \quad \underbrace{f_{(\cdot)}' w_{(\cdot)}}_{2} \quad \cdots \quad \underbrace{f_{(\cdot)}' w}_{(q-1)} + \cdots$$

Case I

If

$$\left| f_{l_m}'\left(s_{l_m}(t-m)\right) w_{l_{m-1} l_m} \right| > 1 \quad \forall m$$

and $m^* = \underset{m}{\operatorname{argmax}} \left| f_{l_m}'\left(s_{l_m}(t-m)\right) w_{l_{m-1} l_m} \right|$



then

$$\left| f'_{l_{m^*}} \left(s_{l_{m^*}} (t - m^*) \right) w_{l_{m^*-1}, l_{m^*}} \right| \uparrow \text{ exponentially}$$

with q (Π has '$q-2$' coefficients each > 1)

This is explosive gradient case

Case II: If

$$\left| f'_{l_m} \left(s_{l_m} (t - m) \right) w_{l_{m-1}, l_m} \right| < 1 \quad \forall m,$$

then

$$\left| f'_{l_{m^*}} \left(s_{l_{m^*}} (t - m^*) \right) w_{l_{m^*-1}, l_{m^*}} \right| \downarrow \text{ exponentially}$$

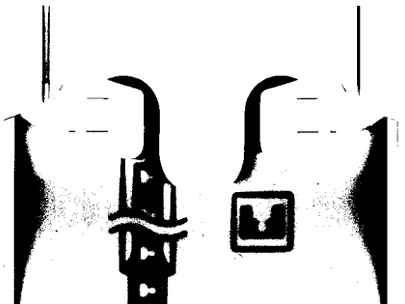
with q (Π has '$q-2$' coefficients each < 1)

This is the vanishing gradient case.

Analysis of $f'_{(\cdot)}$, $w_{(\cdot)}$:

Let $g = \left| f'_{l_m} \left(s_{l_m} (t - m) \right) w_{l_{m-1}, l_m} \right|$ - (13)

If $f = \sigma(\cdot)$, then $f' = f(1 - f)$



Therefore, for logistic sigmoid, (13) can be re written as

$$g = \left(f_{l_m}(s_{l_m}) - f_{l_m}^2(s_{l_m}) \right) w_{l_{m-1} l_m}$$

We want to find the $w^*_{l_{m-1} l_m}$ for which (13) attains its max.

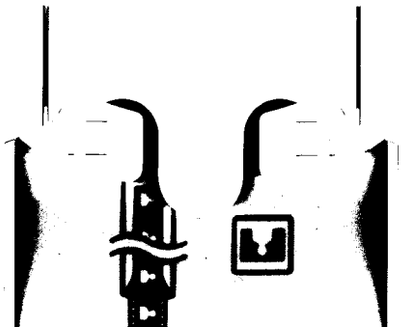
$$\frac{\partial g}{\partial w_{l_m l_{m-1}}} = \left(f'_{l_m}(s_{l_m}) y_{l_{m-1}} - 2 f_{l_m}(s_{l_m}) f'_{l_m}(s_{l_m}) y_{l_{m-1}} \right) w_{l_m l_m} + f'_{l_m}(s_{l_m}) = 0$$

$$\Rightarrow \quad y_{l_{m-1}} \left(1 - 2 f_{l_m}(s_{l_m}) \right) w_{l_{m-1} l_m} + 1 = 0$$

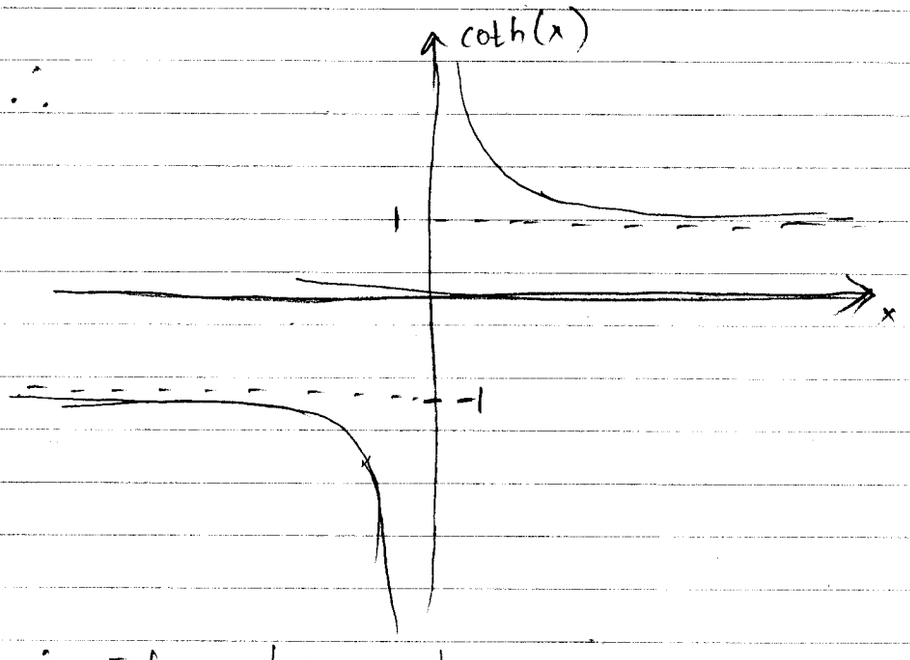
$$\Rightarrow \quad w_{l_{m-1} l_m} = \frac{1}{y_{l_{m-1}}} \frac{1}{2 f_{l_m}(s_{l_m}) - 1}$$

$$= \frac{1}{y_{l_{m-1}}} \frac{1}{\frac{2}{1+e^{-s_{l_m}}} - 1} = \frac{1}{y_{l_{m-1}}} \frac{1+e^{-s_{l_m}}}{1-e^{-s_{l_m}}}$$

$$= \frac{1}{y_{l_{m-1}}} \frac{e^{\frac{1}{2}s_{l_m}} + e^{-\frac{1}{2}s_{l_m}}}{e^{\frac{1}{2}s_{l_m}} - e^{-\frac{1}{2}s_{l_m}}} = \frac{1}{y_{l_{m-1}}} \coth\left(\frac{1}{2} s_{l_m}\right)$$



$$w^*_{l_{m-1} l_m} = \frac{1}{y_{l_{m-1}}} \coth\left(\frac{1}{2} s_{l_m}\right)$$



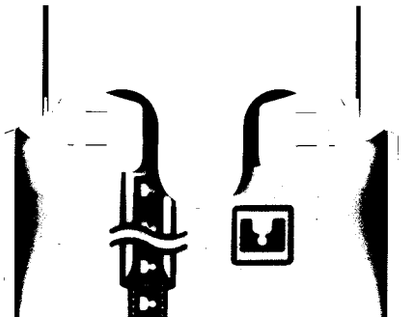
$\therefore$ If $|w_{l_{m-1} l_m}| \to \infty \iff s_{l_m} \to 0$.

$$g = f'_{l_m}(s_{l_m})\, w_{l_{m-1} l_m}$$

$$= f_{l_m}(s_{l_m})\left(1 - f_{l_m}(s_{l_m})\right) w_{l_{m-1} l_m}$$

$$= \frac{1}{1+e^{-s_{l_m}}} \cdot \frac{e^{-s_{l_m}}}{1+e^{-s_{l_m}}} \, w_{l_{m-1} l_m}$$

$$= \frac{1}{\left(1+e^{-s_{l_m}}\right)^2} \cdot \frac{w_{l_{m-1} l_m}}{e^{s_{l_m}}} \simeq \frac{1}{(1+0)^2} \cdot \frac{\infty}{\text{faster } \infty \text{ (?)}} \to 0$$



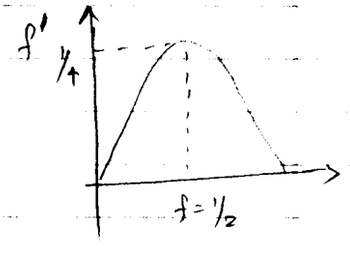
Now, what if $|w^*_{l_{m-1}l_m}|$ is small?

Note $f'_{l_m} = f(1-f) = h$

$\frac{\partial h}{\partial f} = 1 - 2f = 0 \Rightarrow f = 1/2$

$\frac{\partial^2 h}{\partial f^2} = -2 < 0 \quad \therefore \quad f = 1/2$ at max

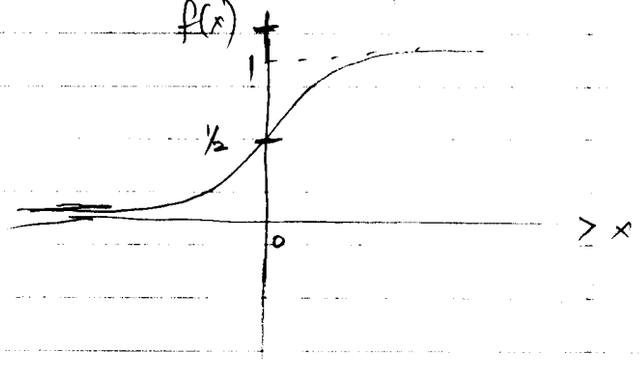
$\left(f'_{l_m}\right)_{max} = \frac{1}{2}\left(1 - \frac{1}{2}\right) = \frac{1}{4}.$



$f = \frac{1}{2} \Rightarrow \frac{1}{2} = \frac{1}{1+e^{-x}}$

$\Rightarrow e^{-x} = 1$

$\Rightarrow x = 0$

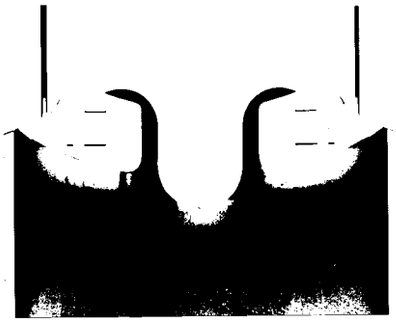


'f' achieves steepest slope at $x = 1/2$.

The steepest slope is $\left(f'\right)_{max} = 1/4$.

$\therefore \quad g = \left|f'_{l_m}(s_{l_m})\, w_{l_{m-1}l_m}\right| < 1$

$= \left|\frac{1}{4} w_{l_{m-1}l_m}\right| < 1 \Rightarrow |w_{l_{m-1}l_m}| < 4.0$



$\therefore$ When (a) $|w_{l_{m-1}l_m}| \to \infty \Rightarrow g \to 0$

(b) $|w_{l_{m-1}l_m}| < 4 \Rightarrow g \to$ value less than 1.

Very Large weights, error vanishes

Small weights, error vanishes

To avoid this problem, we need

$$f'_{l_m}(s_{l_m})\, w_{l_m l_m} = 1$$

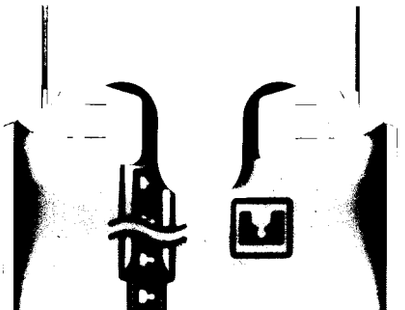
(assume we are interest in the weight connecting 'l_m' to 'l_m')

$$\Rightarrow \int f'_{l_m}(s_{l_m})\, w_{l_m l_m}\, d(s_{l_m}) = \int d(s_{l_m})$$

$$\Rightarrow f_{l_m}(s_{l_m})\, w_{l_m l_m} = s_{l_m}$$

$$\Rightarrow f_{l_m}(s_{l_m}) = \frac{s_{l_m}}{w_{l_m l_m}}$$

$\therefore f_{l_m}(s_{l_m}) \propto s_{l_m}$ $\therefore f(\,)$ has to be linear



In addition, unit 'j' activation has to be constant :

$$y_j(t+1) = f_j\left(s_j(t+1)\right) = f_j\left(w_{jj}\, y_j(t)\right)$$

$$= f_j\left(w_{jj}\, \frac{s_j(t)}{w_{jj}}\right)$$

$$= y_j(t)$$

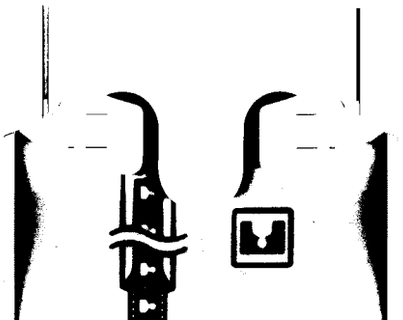
$$\therefore \quad y_j(t+1) = y_j(t)$$

∴ Constant Error Carousel we need

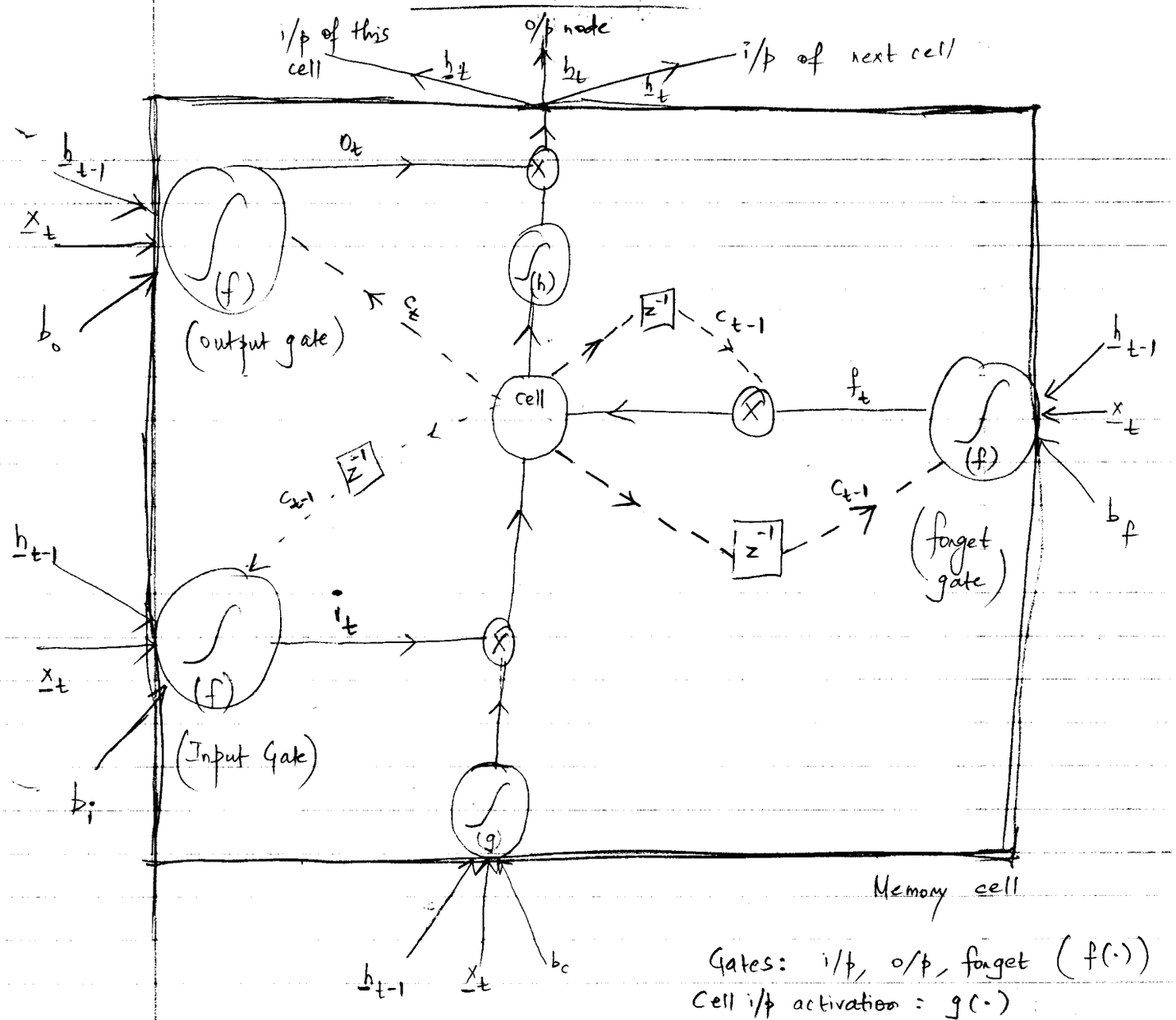
(a) $y_j(t) = f_j(s_j(t))$ ~~&~~ $s_j(t)$ (linear)

(b) $y_j(t+1) = y_j(t)$ (constant)

This is implemented in LSTM cell.



LSTM block



Gates: i/p, o/p, forget ($f(\cdot)$)

Cell i/p activation : $g(\cdot)$

Cell o/p activation : $h(\cdot)$

$$i_t = \sigma\left(W_{ix}\, \underline{x}_t + W_{ih}\, \underline{h}_{t-1} + W_{ic}\, \underline{c}_{t-1} + b_i \right)$$

$$f_t = \sigma\left(W_{fx}\, \underline{x}_t + W_{fh}\, \underline{h}_{t-1} + W_{fc}\, \underline{c}_{t-1} + b_f \right)$$

$$c_t = f_t\, c_{t-1} + i_t\, g\left(W_{cx}\, x_t + W_{ch}\, \underline{h}_{t-1} + b_c \right)$$

$$o_t = \sigma\left(W_{ox}\, \underline{x}_t + W_{oh}\, \underline{h}_{t-1} + W_{oc}\, \underline{c}_t + b_o \right)$$

$$h_t = o_t \cdot h(c_t)$$

$$y_t = W_{yh}\, \underline{h}_t + b_y$$