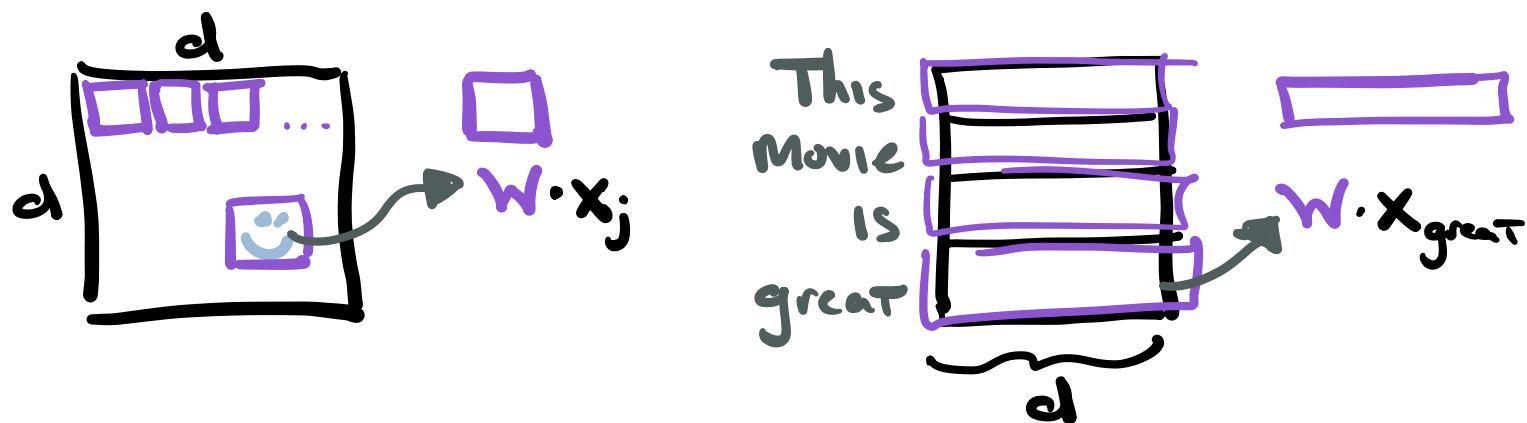


DS 4440 Recurrent Neural Networks

Last few lectures ... ConvNets



These filters are position invariant.

But data often is sequentially structured - e.g.

- Language
- Genetic Code
- Physiological data
- Stock prices

$$x_t \sim P(x_t | x_{t-1} \dots x_1)$$

$$y_t = f(x_t \dots x_1)$$

Key
Issue

How can we model
an arbitrary length
Sequence with a finite
Set of parameters?

Markov Assumption

$$P(x_t | x_{t-1} \dots x_1) = P(x_t | x_{t-1} \dots x_{t-k})$$

RNNs learn to induce a fixed length
representation of $x_1 \dots x_T$.

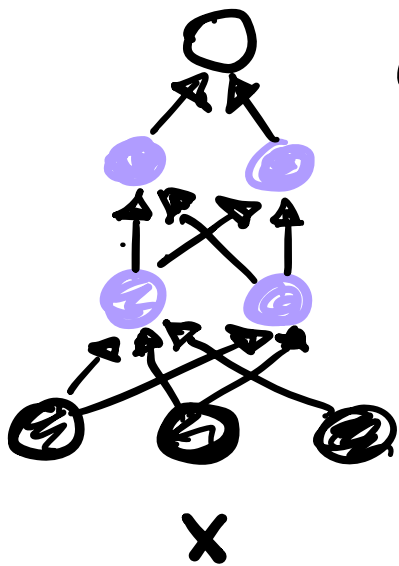
Example Learning to Count in
Variable length Sequences

$$\begin{bmatrix} [a \ b \ b] \\ [b \ b \ a \ a \ a \ b] \\ [a \ a \ b \ a \ b \ b \ b \ b] \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \left. \begin{array}{l} 1 \text{ if Count}(b) \\ \text{is even} \\ 0 \text{ otherwise} \end{array} \right\}$$

X Y

Unclear how we could model this with MLPs or ConvNets.
We need State.

MLPs have hidden units

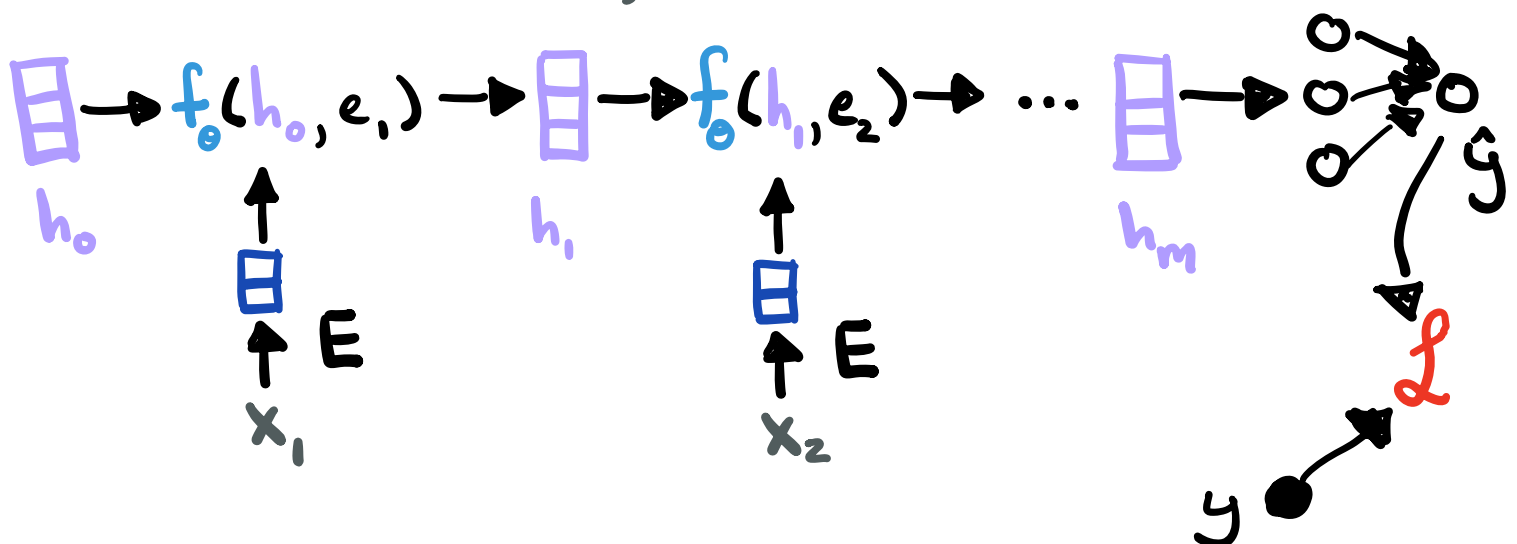


Q. Why doesn't This suffice?

A. Need to process X in order, update h at each step.

RNNs

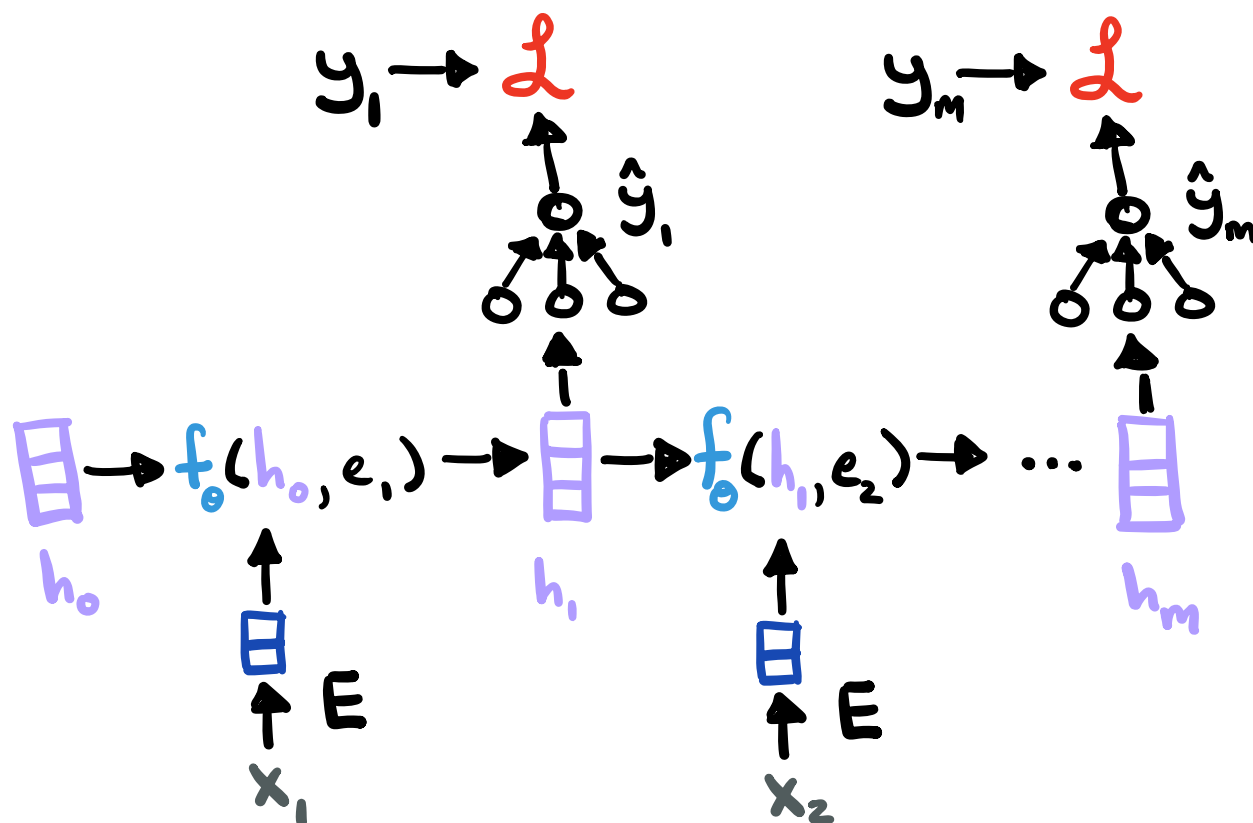
$$X = \{x_1, x_2, \dots, x_m\}, y \in \mathbb{R}$$



f_θ is shared.

Sequence Tagging: $\hat{y}$ at each τ .

$$X = \{x_1, x_2, \dots, x_m\}, \quad y = \{y_1, y_2, \dots, y_m\}$$



Formally (batched)

$$x_\tau \in \mathbb{R}^{b \times d}$$

batch size
input dims

$$h_\tau \in \mathbb{R}^{b \times h}$$

hidden size

$$h_{\tau+1} \leftarrow \sigma(x_\tau W_x + h_\tau W_h + b_h)$$

(bxd)(dxh) (bxh)(hxh) (1xh)

Can RNNs Track b parity?

Suppose $a \rightarrow 0$ $b \rightarrow 1$

Let P_t be the parity of b 's so far, so
far, so $P_t \leftarrow P_{t-1} \text{ XOR } x_t$

Recall from lecture 3
we will need a nonlinearity

Now let $W_x = [1 \ 1]$ $\sigma = \text{ReLU}$

$$W_h = \begin{bmatrix} 1 & 1 \\ -2 & -2 \end{bmatrix} \quad b_h = [0 \ -1]$$

$$x_t W_x = [x_t \ x_t]$$

State change:

$$\begin{aligned} h_{t-1} W_h &= [h_{t-1,1} \ h_{t-1,2}] \begin{bmatrix} 1 & 1 \\ -2 & -2 \end{bmatrix} \\ &= [\underbrace{h_{t-1,1} - 2h_{t-1,2}}_{P_{t-1}}, \underbrace{h_{t-1,1} - 2h_{t-1,2}}_{P_{t-1}}] \end{aligned}$$

Add bias

$$x_t W_x + h_{t-1} W_h + b_h =$$

$$[x_t + P_{t-1}, x_t + P_{t-1}^{-1}]$$

$$h_t = [\text{ReLU}(x_t + P_{t-1}), \text{ReLU}(x_t + P_{t-1}^{-1})]$$

Consider input $\begin{matrix} [a & b & b] \\ 0 & 1 & 1 \end{matrix}$

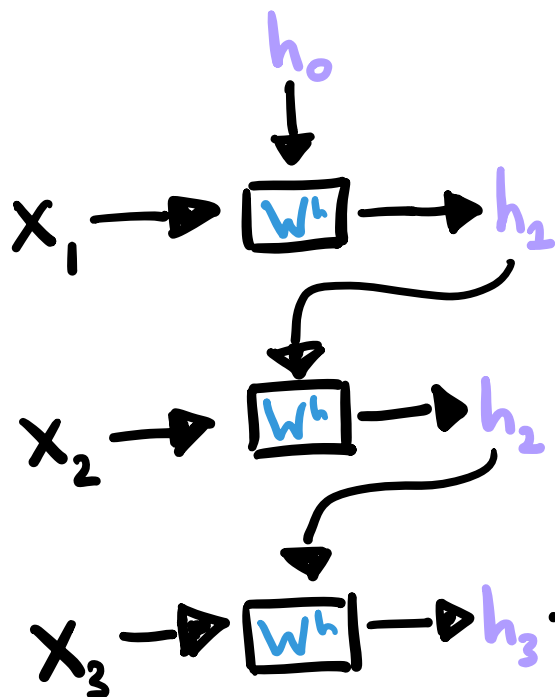
$$\text{let } h_0 = [0 \ 0]$$

τ	x_τ	$s = P_{\tau-1} + x_\tau$	h_τ	$P_\tau = h_{\tau,1} - 2h_{\tau,2}$	y_τ
0	-	-	$[0 \ 0]$	0	1
1	0_a	0	$[0 \ 0]$	0	1
2	1_b	1	$[1 \ 0]$	1	0
3	1_b	2	$[2 \ 1]$	0	1

y_τ is just $1 - P_\tau$. $W_0 = \begin{bmatrix} -1 \\ 2 \end{bmatrix}$ $b_0 = 1$

$$\hat{y} = h_\tau W_0 + b_0 = -h_{\tau,1} + 2h_{\tau,2} + 1 = 1 - P_\tau$$

Backprop through time



Assume Simple Model

$$\hat{y} = W^o \cdot h_3$$

$$\nabla_{W^o} \mathcal{L} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \nabla_{W^o} \hat{y}$$

What about ∇_{W^h} ?

$$\nabla_{W^h} = \sum_{t=1}^T \nabla_{W^h} h_t \cdot \nabla_{h_t} \mathcal{L}$$

$$\nabla_{h_3} \mathcal{L} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \nabla_{h_3} \hat{y} = W^o \in \mathbb{R}^d$$

But ∇_{h_2} More Complicated

$$\nabla_{h_2} L = (\nabla_{h_2} h_3) \cdot (\nabla_{h_3} L)$$

$$\downarrow$$

$$d \begin{bmatrix} \frac{\partial h_{3,1}}{\partial h_{2,1}} & \dots & \frac{\partial h_{3,1}}{\partial h_{2,d}} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_{3,d}}{\partial h_{2,1}} & \dots & \frac{\partial h_{3,d}}{\partial h_{2,d}} \end{bmatrix} \text{ Jacobian}$$

d

(See Colab Notebook)

Problem long sequences require repeat multiplies. Consider

$$\nabla_{h_1} L = (\nabla_{h_1} h_2) \cdot (\nabla_{h_2} h_3) \cdot (\nabla_{h_3} L)$$

... And This is a Toy example.

Gradients ∇ Tend to **Explode** or VANISH.

Can address Explosions via gradient Clipping:

If $\text{Norm}(\nabla_{\theta}) \geq \tau$

$$\nabla_{\theta} \leftarrow \nabla_{\theta} / 2$$

For INTUITION: Explosions with Scalars

State = Single number. No σ .

$$h_t = \omega h_{t-1}$$

$\rightarrow \nabla_{h_t}$ gets a factor of ω per step!

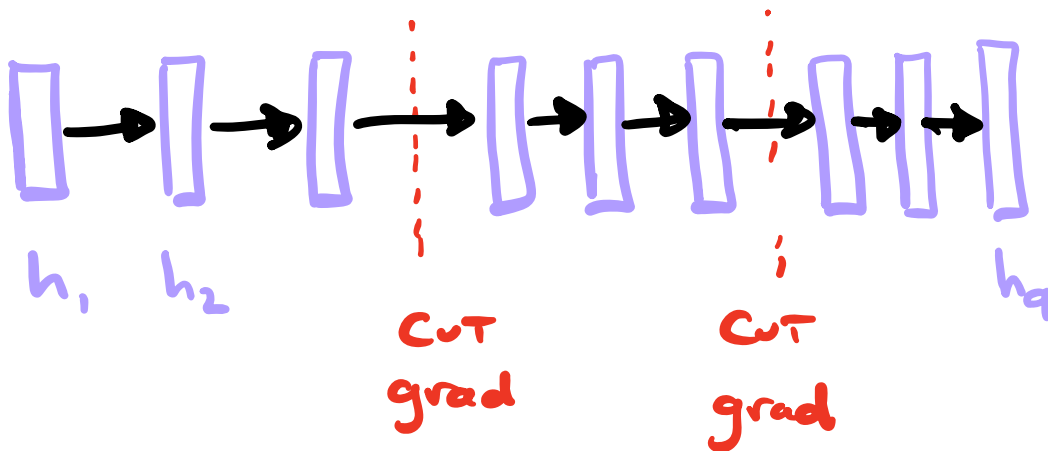
	<u>T=10</u>	<u>50</u>	<u>100</u>	
$\omega = .9$	.39	$5.7 \cdot 10^{-3}$	$3 \cdot 10^{-5}$	Vanish
$\omega = 1$	1	1	1	
$\omega = 1.1$	2.4	$1.1 \cdot 10^2$	$1.3 \cdot 10^4$	Explode!

Clipping can help with **explosion**,
not **vanishing**.

(gating)
(Next Time: Architectures to help!)

Another related issue: Memory.

Have to hold on to all activations
as we move $\rightarrow$ for weight update.



Truncated BPTT passes state forward
but `.detach()`'s grad.