Natural Language Processing
CSE 325/425

Sihong Xie

Lecture 13:

- Recurrent neural networks (RNN)

Language model review

Bi-gram $P(w_t|w_{t-1}) = \frac{\text{Count } w_t}{\text{Count } w_{t-1}}$

n-gram $P(w_t|[w_{t-1},\ldots,w_{t-n+1}]) = \frac{\text{Count } [w_t,\ldots,w_{t-n+1}]}{\text{Count } [w_{t-1},\ldots,w_{t-n+1}]}$

Two issues

- Data sparsity: the occurrences of many $[w_t,\ldots,w_{t-n+1}]$ are zeros.
- Model complexity: number of parameters increases exponentially in *n*.

The two issues are related: if we want longer range dependencies, we increase *n*, then both the data sparsity and model complexity become worse.

The students walked in the room and asked the ___?___ about the quiz questions.

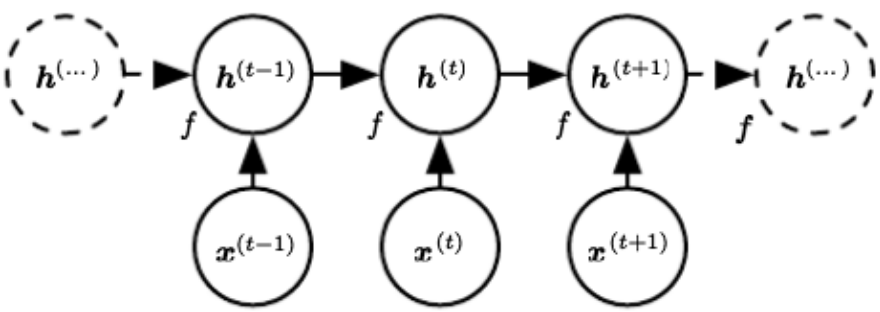
Address the issues using neural network

Don't store the *n*-grams, but use a fixed-size model to predict the *n*-grams.

- model complexity is fixed.
- no data sparsity issue (no *n*-gram is computed)
- can be generalized to unseen sequences.

Recurrent Neural Networks (RNN)

$$\boldsymbol{h}^{(t)} = f(\boldsymbol{h}^{(t-1)}, \boldsymbol{x}^{(t)}; \boldsymbol{\theta})$$



"*<u>Recurrent</u>*" because the next ***h*** is compute by the same function that calculates the previous ***h***.

$\boldsymbol{h}^{(t)}$	state summarizing what has happened before time step *t*. (theoretically speaking)
$\boldsymbol{\theta}$	a single model specifying how to transit to the next state (independent of *t*).

A running example

Recurrent Neural Networks

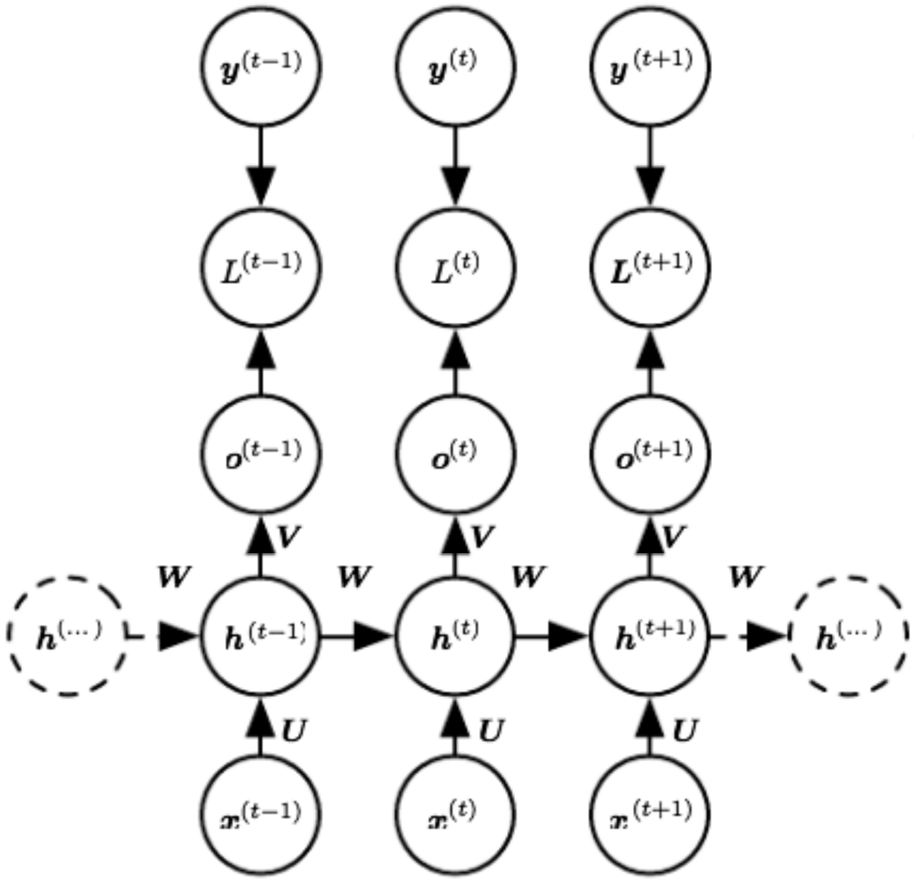
Output sequence
(e.g., POS tags)

Loss function

Output units

Hidden states

Input sequence
(e.g., sentence)



Training data:

$$\{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(\tau)}\}, \{\boldsymbol{y}^{(1)}, \dots, \boldsymbol{y}^{(\tau)}\}$$

Trainable parameters:

$$\boldsymbol{\theta} = \{\boldsymbol{U}, \boldsymbol{W}, \boldsymbol{V}, \boldsymbol{b}, \boldsymbol{c}\}$$

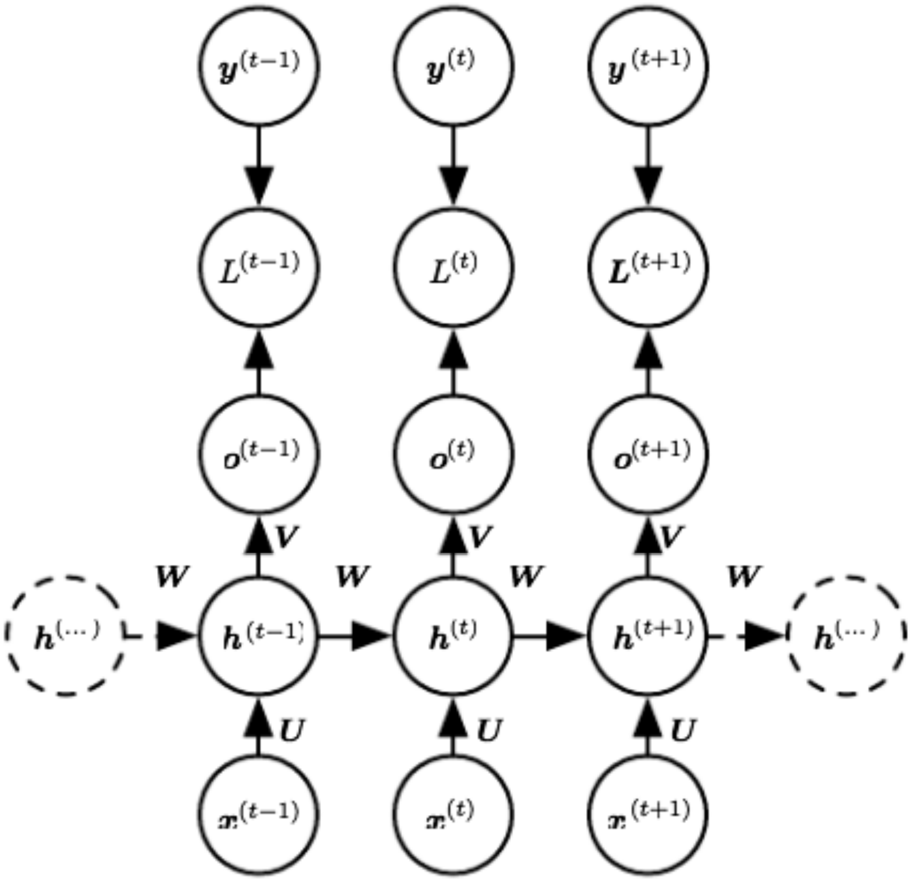
$\boldsymbol{V}$: maps from *h* to *o*

$\boldsymbol{W}$: maps from *h* to *h*

$\boldsymbol{U}$: maps from *x* to *h*

$\boldsymbol{b}, \boldsymbol{c}$: biases

RNN forward pass

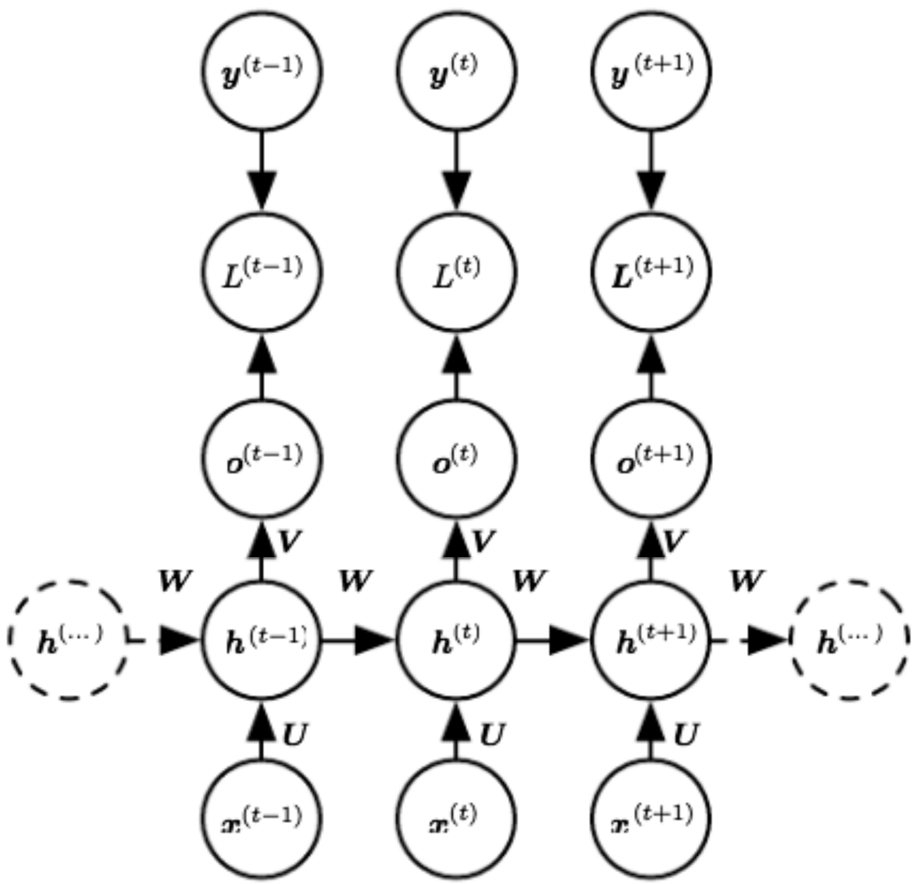


$$
\begin{aligned}
\boldsymbol{a}^{(t)} &= \boldsymbol{b} + \boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\boldsymbol{x}^{(t)}, \\
\boldsymbol{h}^{(t)} &= \tanh(\boldsymbol{a}^{(t)}), \\
\boldsymbol{o}^{(t)} &= \boldsymbol{c} + \boldsymbol{V}\boldsymbol{h}^{(t)}, \\
\hat{\boldsymbol{y}}^{(t)} &= \mathrm{softmax}(\boldsymbol{o}^{(t)}),
\end{aligned}
$$

activation function

probability distribution

RNN forward pass



Negative log likelihood (NLL) loss, or the "perplexity"

$$L\left(\{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(\tau)}\}, \{\boldsymbol{y}^{(1)}, \dots, \boldsymbol{y}^{(\tau)}\}\right)$$

$$= \sum_t L^{(t)}$$

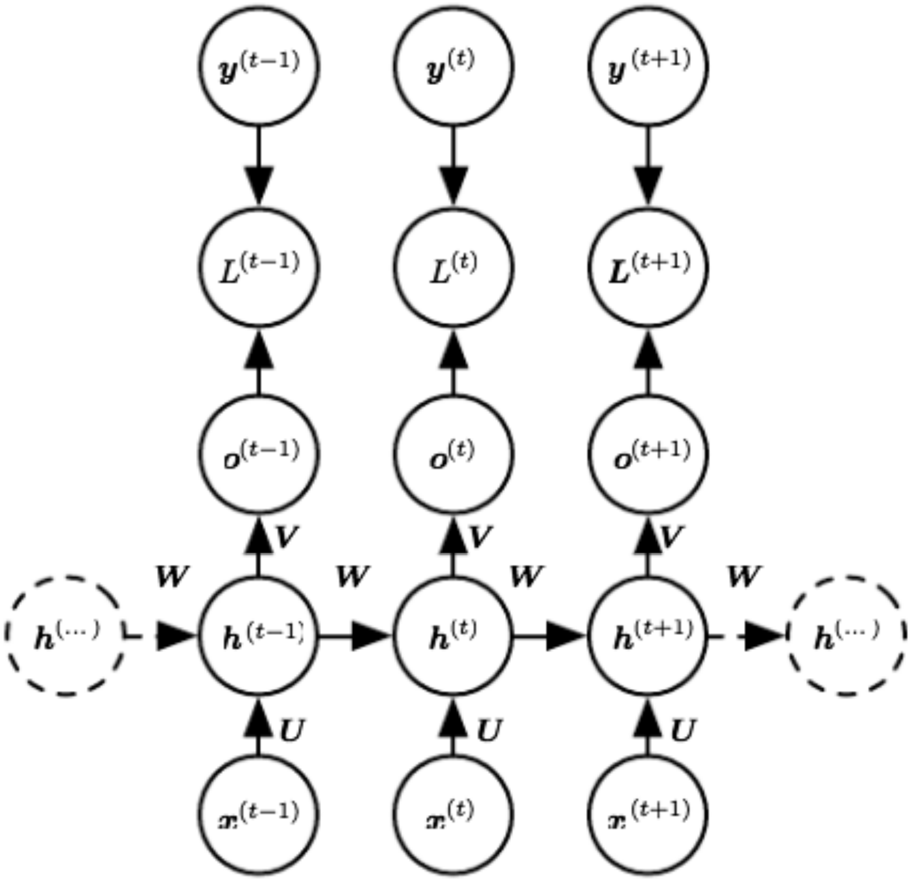
the ground truth label.

$$= -\sum_t \log p_{\text{model}}\left(y^{(t)} \mid \{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(t)}\}\right)$$

$$\hat{\boldsymbol{y}}^{(t)} = \text{softmax}(\boldsymbol{o}^{(t)})$$

A running example

RNN back propagation



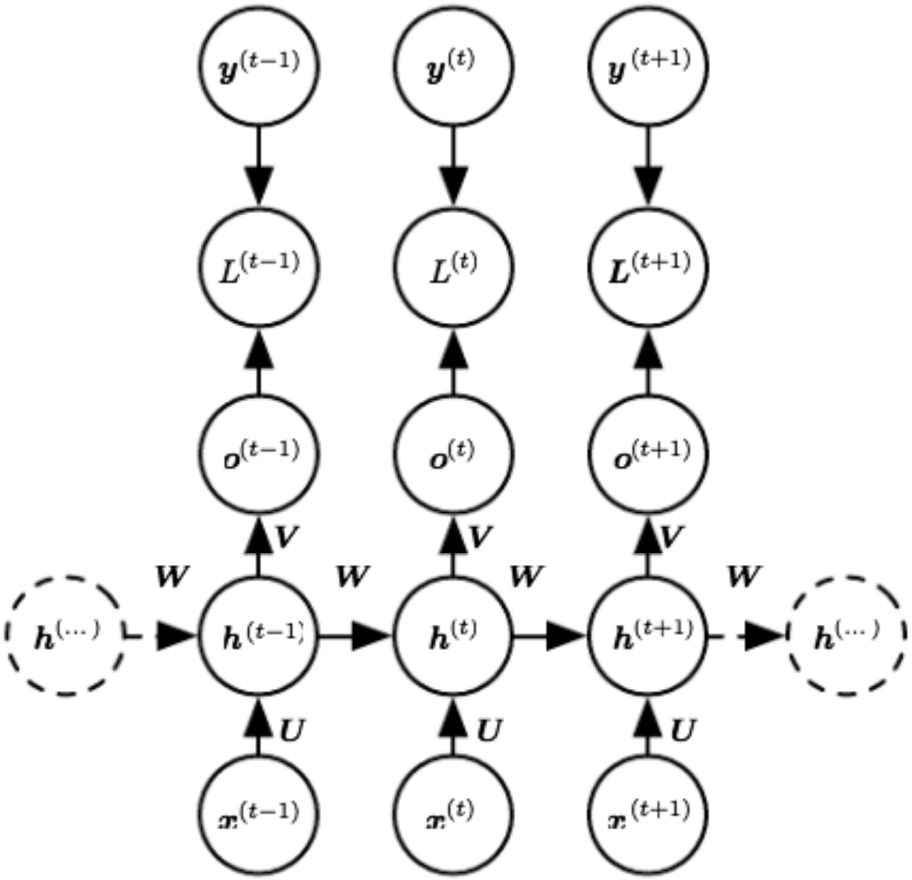
BPTT (Back Propagation through Time)

- Used for gradient descent training;
- A special name for RNN back-propagation;
- Need all information in the forward pass, making BPTT sequential and hard to parallelize.
- $\boldsymbol{\theta} = \{\boldsymbol{U}, \boldsymbol{W}, \boldsymbol{V}, \boldsymbol{b}, \boldsymbol{c}\}$ used in all steps.
- Two derivative rules applied:

$$\nabla_x(f(x) + g(x)) = \nabla_x f(x) + \nabla_x g(x)$$

$$\nabla_x(f(g(x)) = \nabla_g f(g(x)) \times \nabla_x g(x)$$

RNN back propagation



BPTT (Back Propagation through Time)

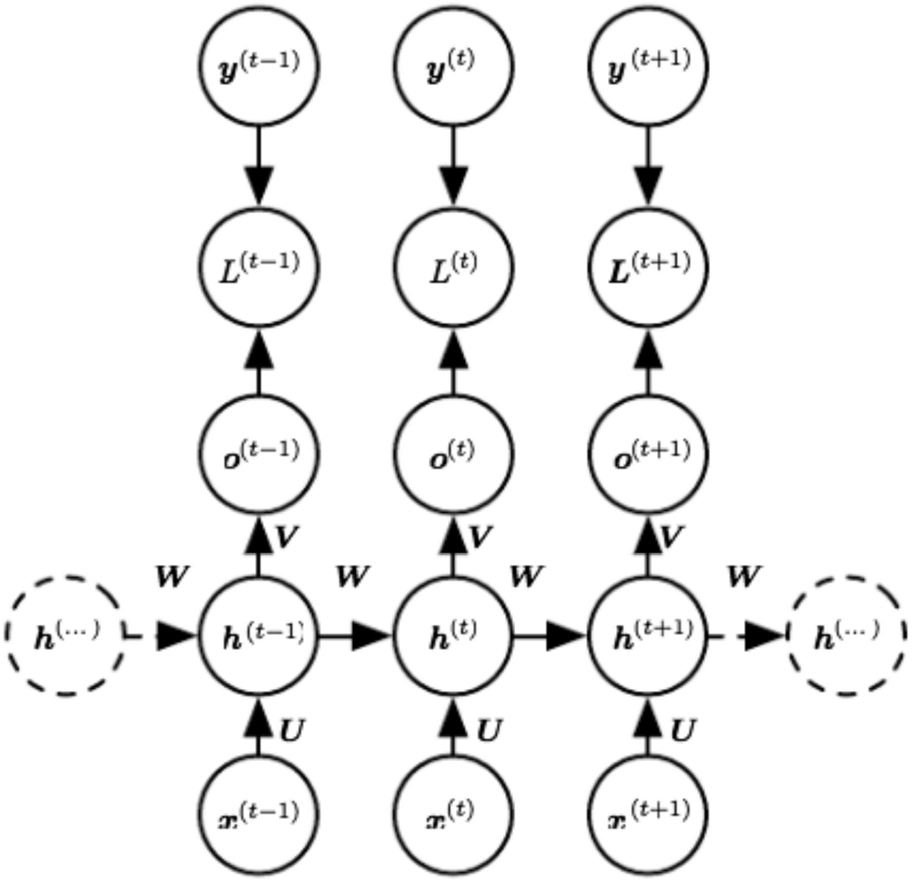
- focus on each step t (the final gradient is the sum of all gradients at each step.

$$\frac{\partial L}{\partial L^{(t)}} = 1$$

$$(\nabla_{\boldsymbol{o}^{(t)}} L)_i = \frac{\partial L}{\partial o_i^{(t)}} = \frac{\partial L}{\partial L^{(t)}} \frac{\partial L^{(t)}}{\partial o_i^{(t)}} = \hat{y}_i^{(t)} - \mathbf{1}_{i=y^{(t)}}$$

(an element of $\nabla_{\boldsymbol{o}^{(t)}} L$)

RNN backprop (base case)



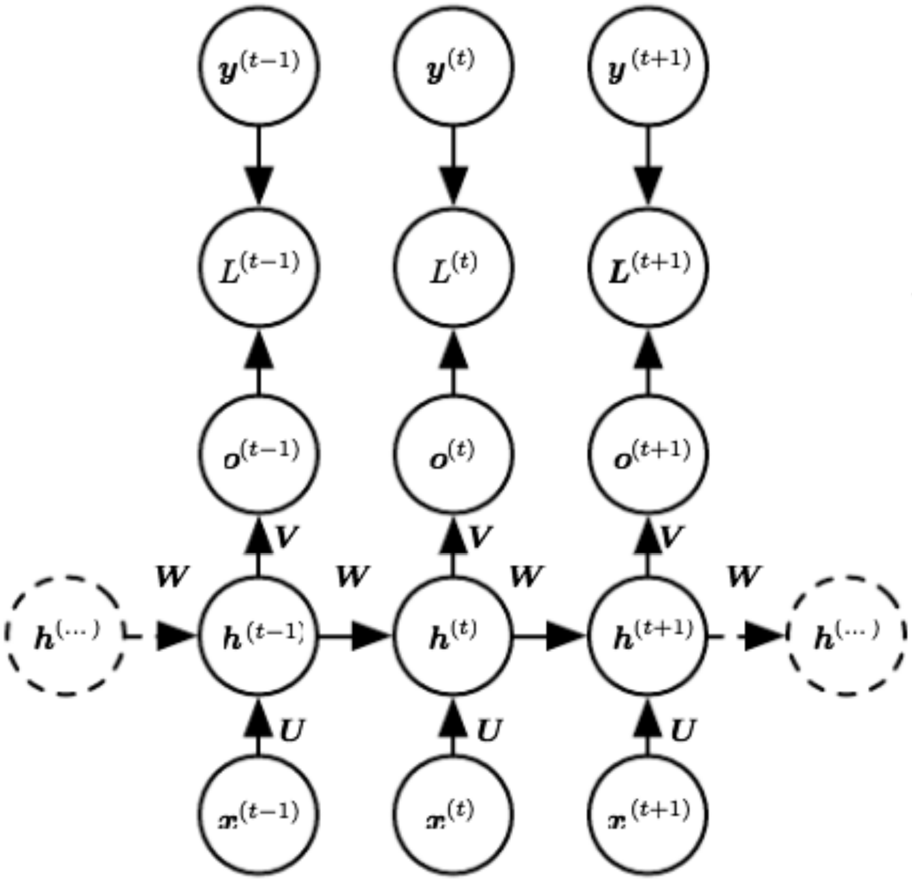
BPTT (Back Propagation through Time)

- Base case: at the final step τ

$$\nabla_{\boldsymbol{h}^{(\tau)}} L = \boldsymbol{V}^{\top} \nabla_{\boldsymbol{o}^{(\tau)}} L.$$

since $\boldsymbol{o}^{(\tau)} = V\boldsymbol{h}^{(\tau)} + \boldsymbol{c}$

RNN backprop (recurrent)



BPTT (Back Propagation through Time)

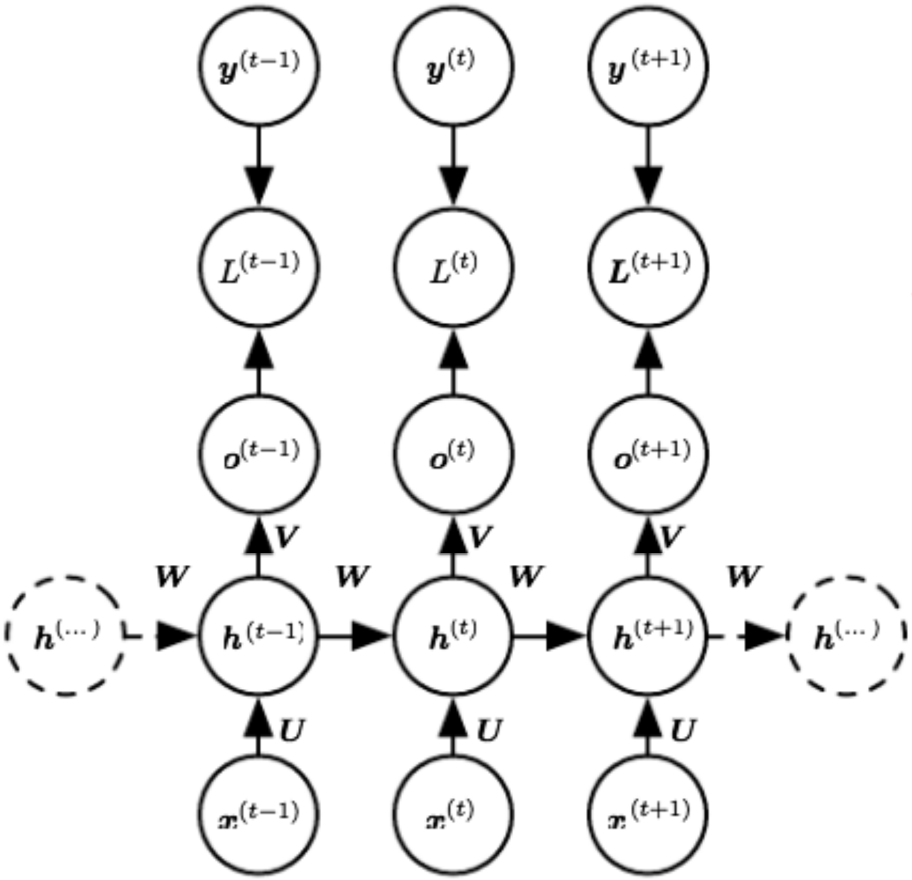
- Recursively, at any step $1 \leq t < \tau$

$$\nabla_{\boldsymbol{h}^{(t)}} L = \left(\frac{\partial \boldsymbol{h}^{(t+1)}}{\partial \boldsymbol{h}^{(t)}}\right)^{\top} (\nabla_{\boldsymbol{h}^{(t+1)}} L) + \left(\frac{\partial \boldsymbol{o}^{(t)}}{\partial \boldsymbol{h}^{(t)}}\right)^{\top} (\nabla_{\boldsymbol{o}^{(t)}} L)$$

$$= \boldsymbol{W}^{\top} \text{diag}\left(1 - \left(\boldsymbol{h}^{(t+1)}\right)^{2}\right) (\nabla_{\boldsymbol{h}^{(t+1)}} L) + \boxed{\boldsymbol{V}^{\top} (\nabla_{\boldsymbol{o}^{(t)}} L)}$$

$$\boldsymbol{o}^{(t)} = V \boldsymbol{h}^{(t)} + \boldsymbol{c}$$

RNN backprop (recurrent)



BPTT (Back Propagation through Time)

- Recursively, at any step $1 \leq t < \tau$

$$\nabla_{\boldsymbol{h}^{(t)}} L = \left(\frac{\partial \boldsymbol{h}^{(t+1)}}{\partial \boldsymbol{h}^{(t)}}\right)^{\top} (\nabla_{\boldsymbol{h}^{(t+1)}} L) + \left(\frac{\partial \boldsymbol{o}^{(t)}}{\partial \boldsymbol{h}^{(t)}}\right)^{\top} (\nabla_{\boldsymbol{o}^{(t)}} L)$$

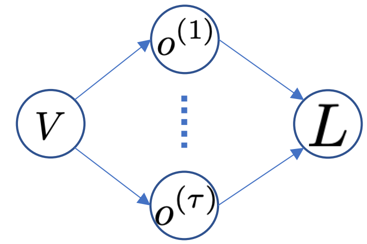
$$= \boxed{\boldsymbol{W}^{\top} \mathrm{diag}\left(1 - \left(\boldsymbol{h}^{(t+1)}\right)^2\right) (\nabla_{\boldsymbol{h}^{(t+1)}} L)} + \boldsymbol{V}^{\top} (\nabla_{\boldsymbol{o}^{(t)}} L)$$

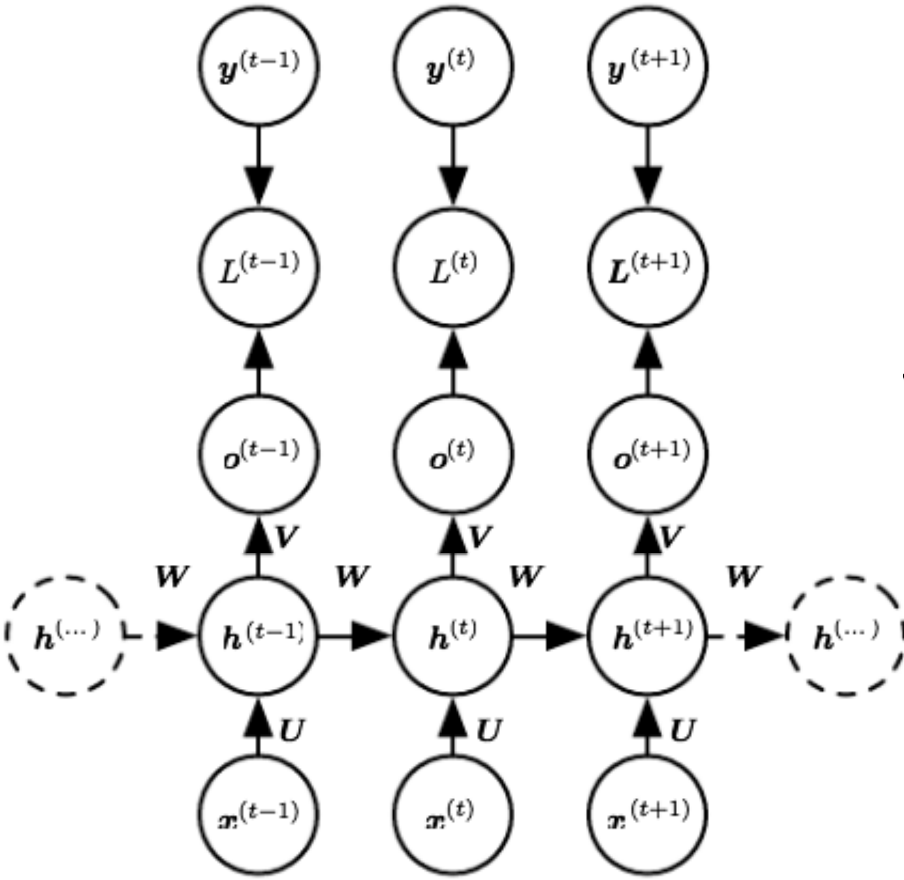
$$\boldsymbol{h}^{(t+1)} = \tanh(\boldsymbol{a}^{(t+1)})$$ (element-wise)

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$\boldsymbol{a}^{(t+1)} = W\boldsymbol{h}^{(t)} + U\boldsymbol{x}^{(t+1)} + \boldsymbol{b}$$

RNN backprop (params)





BPTT (Back Propagation through Time)

- at any step $1 \le t < \tau$

since $\boldsymbol{o}^{(t)} = V\boldsymbol{h}^{(t)} + \boldsymbol{c}$

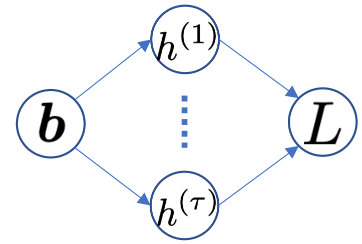
$$\nabla_{\boldsymbol{V}} L^{(t)} = (\nabla_{\boldsymbol{o}^{(t)}} L^{(t)})\boldsymbol{h}^{(t)\top} \qquad \nabla_{\boldsymbol{c}} L^{(t)} = \nabla_{\boldsymbol{o}^{(t)}} L^{(t)}$$

accumulate over time

$$\nabla_{\boldsymbol{V}} L = \sum_t \sum_i \left(\frac{\partial L}{\partial o_i^{(t)}}\right) \nabla_{\boldsymbol{V}^{(t)}} o_i^{(t)} = \sum_t (\nabla_{\boldsymbol{o}^{(t)}} L)\, \boldsymbol{h}^{(t)\top}$$

$$\nabla_{\boldsymbol{c}} L = \sum_t \left(\frac{\partial \boldsymbol{o}^{(t)}}{\partial \boldsymbol{c}}\right)^{\top} \nabla_{\boldsymbol{o}^{(t)}} L = \sum_t \nabla_{\boldsymbol{o}^{(t)}} L$$

RNN backprop (params)

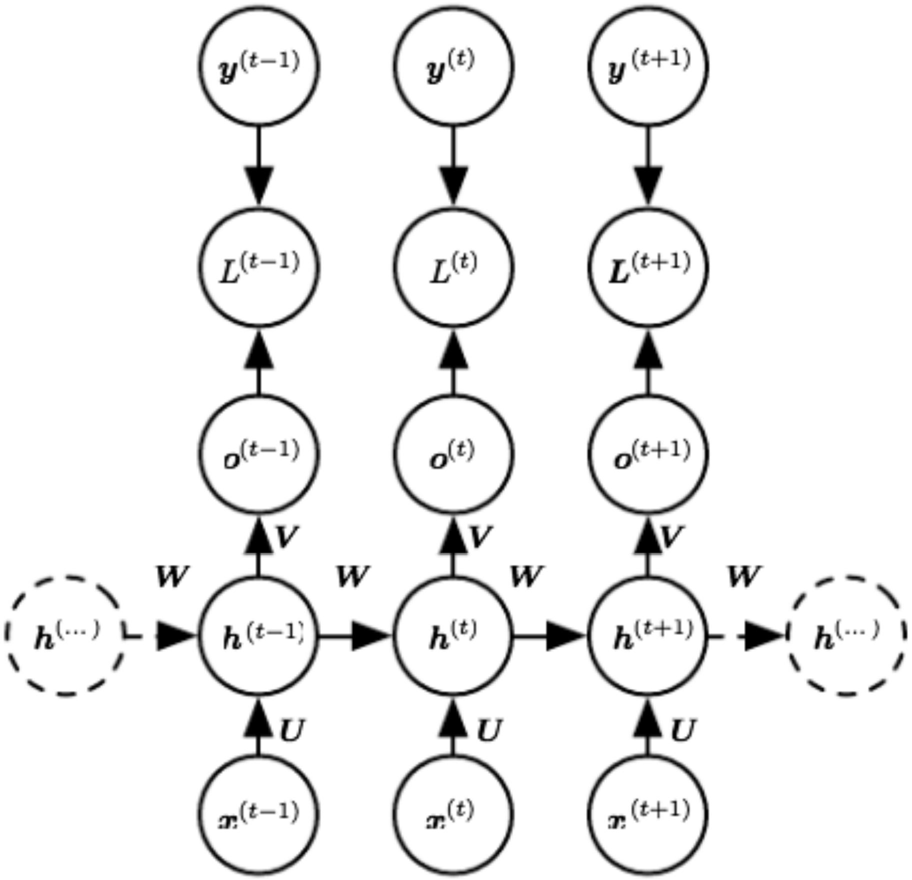


BPTT (Back Propagation through Time)

- at any step $1 \le t < \tau$

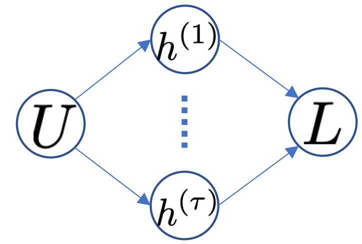
since $\boldsymbol{h}^{(t)} = \tanh(\boldsymbol{a}^{(t)})$

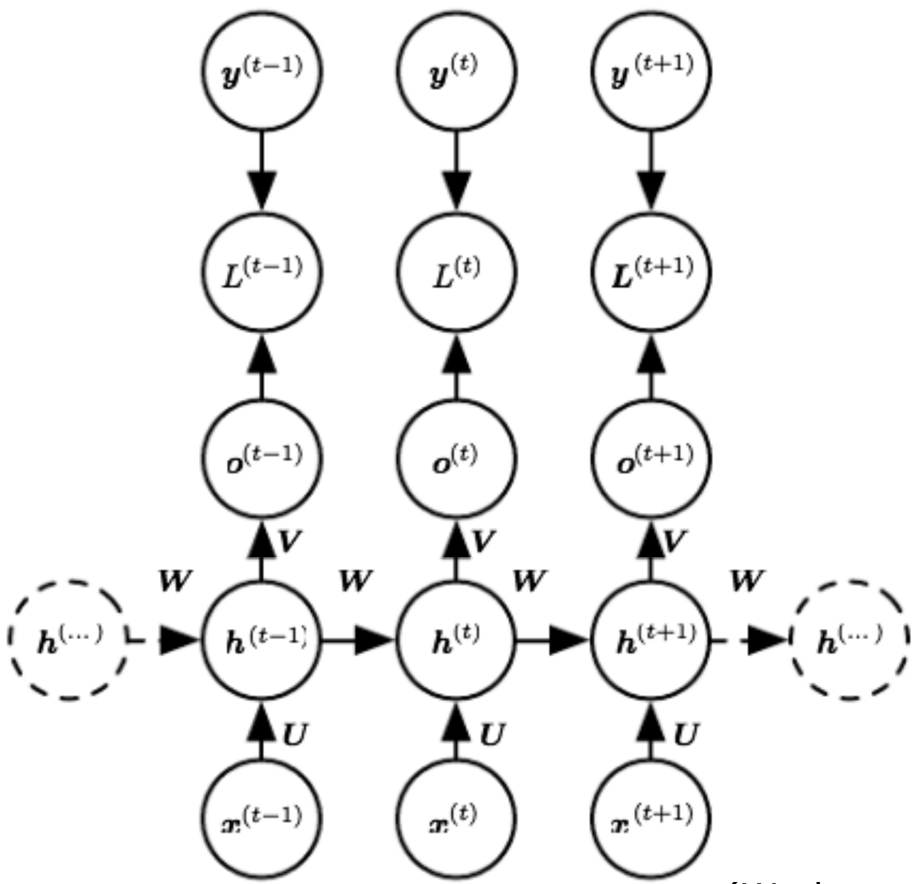
$$\boldsymbol{a}^{(t)} = \boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\boldsymbol{x}^{(t)} + \boldsymbol{b}$$



$$\nabla_{\boldsymbol{b}} L = \sum_t \left(\frac{\partial \boldsymbol{h}^{(t)}}{\partial \boldsymbol{b}^{(t)}}\right)^\top \nabla_{\boldsymbol{h}^{(t)}} L = \sum_t \operatorname{diag}\left(1 - \left(\boldsymbol{h}^{(t)}\right)^2\right) \nabla_{\boldsymbol{h}^{(t)}} L$$

RNN backprop (params)





BPTT (Back Propagation through Time)

- at any step $1 \leq t < \tau$

since $\boldsymbol{h}^{(t)} = \tanh(\boldsymbol{a}^{(t)})$

$$\boldsymbol{a}^{(t)} = \boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\boldsymbol{x}^{(t)} + \boldsymbol{b}$$

$$\nabla_{U} L = \sum_t \operatorname{diag}\left(1 - \left(\boldsymbol{h}^{(t)}\right)^2\right)(\nabla_{\boldsymbol{h}^{(t)}} L)\, \boldsymbol{x}^{(t)^\top}$$

(We have done $\nabla_{\boldsymbol{h}^{(t-1)}} L$, and leave $\nabla_{\boldsymbol{W}} L$ as an exercise.)