Recurrent Neural Networks

Instructor: Xiaoqian Wang
9/30/2024

Slides prepared based on the Lectures slides of Sequence Modeling: Recurrent and Recursive Nets from https://www.deeplearningbook.org/lecture_slides.html

Discussion of MLP

- Large number of parameters
- Disregard sequential information
  - Text analysis
  - Speech recognition
  - Medical time series
  - Stock prices

Recurrent Neural Network

- Sequence Modeling
  - Repeat a same process multiple times
- Applicable to sequential input
- Everything else stays the same
  - Maximum likelihood, Back-propagation, etc.

Classical Dynamical Systems

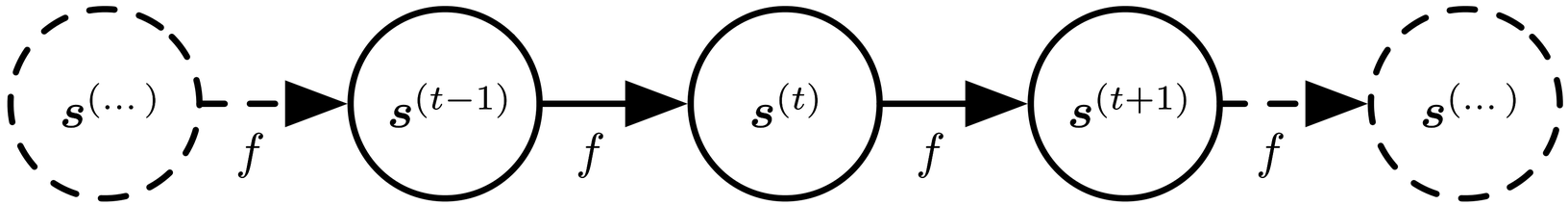


Figure 10.1

Unfolding Computation Graphs

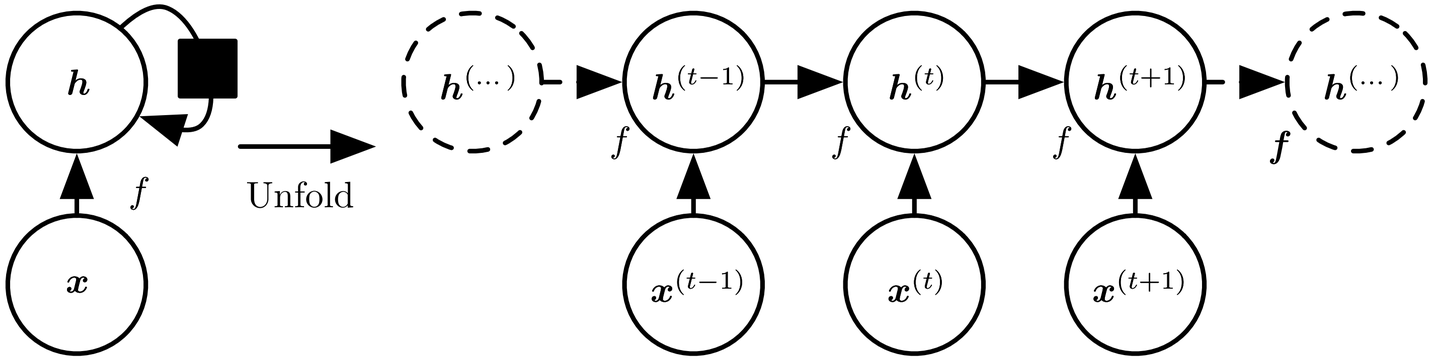


Figure 10.2

Recurrent Hidden Units

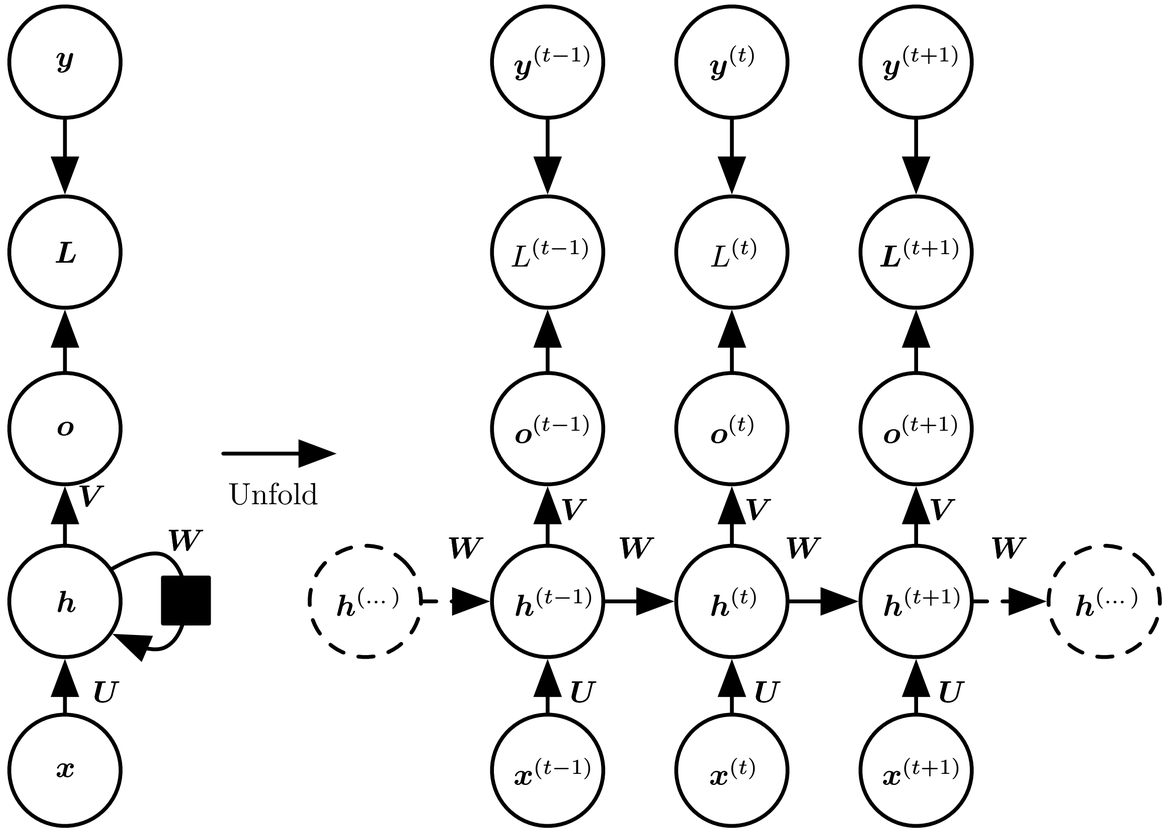
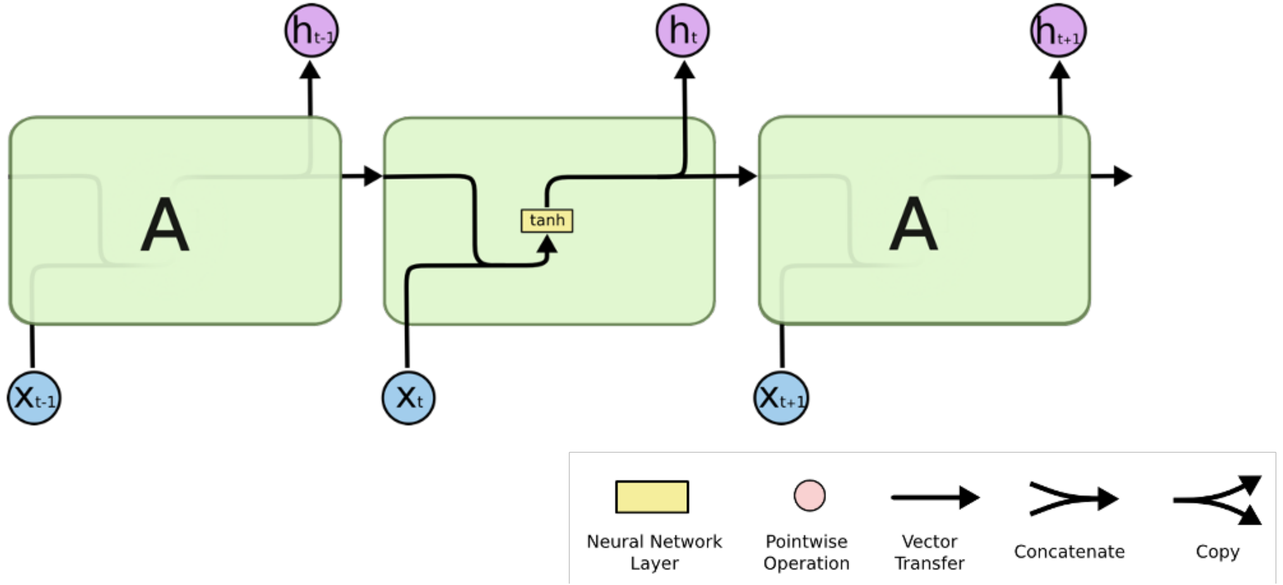


Figure 10.3

A vanilla RNN can be made with linear and activation layers

- The RNN module can be written as $f_\theta(h_{\ell-1}, x_\ell) = (h_\ell, y_\ell)$
  - $h_\ell = \tanh(W_h h_{\ell-1} + W_x x_\ell + b_h)$
  - $y_\ell = W_y h_\ell + b_y = W_y \tanh(W_h h_{\ell-1} + W_x x_\ell) + b_y$
  - The parameters of the model are the weights and biases

$$\theta = (W_h, W_x, W_y, b_h, b_y)$$

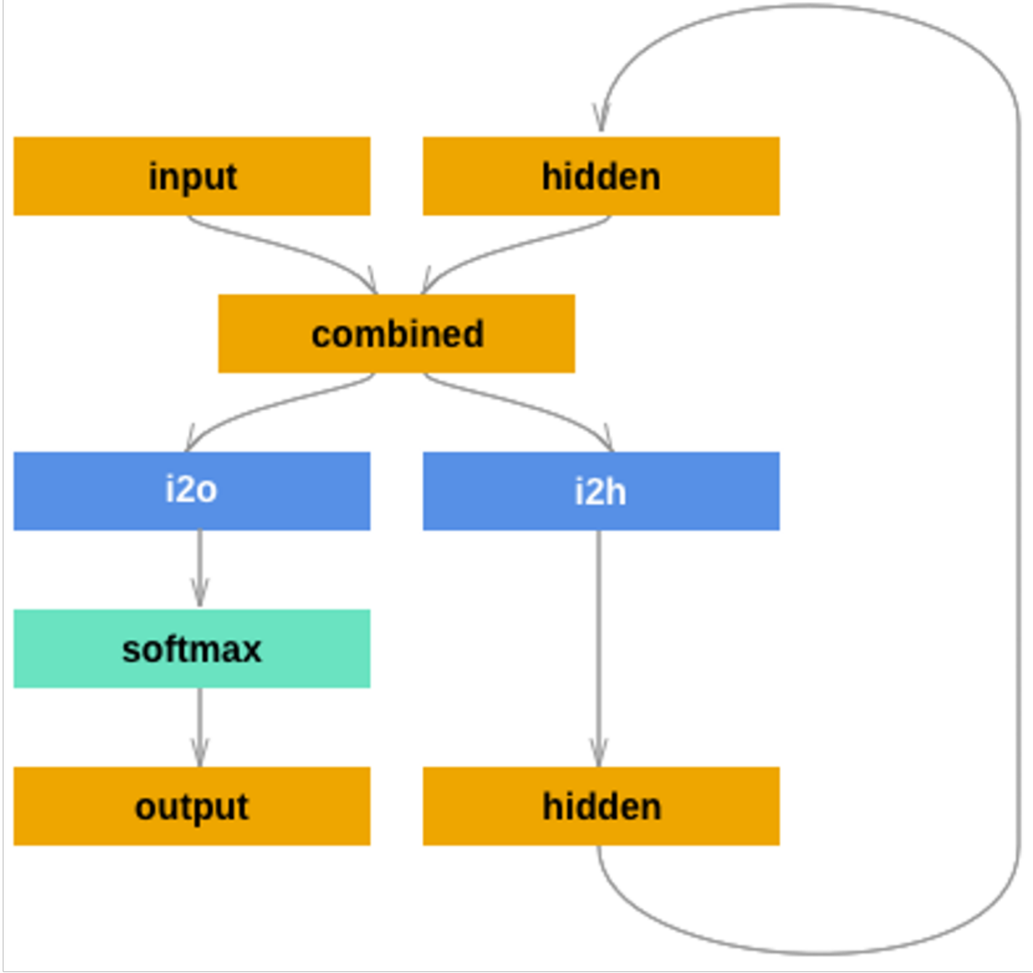


In these figures $t \equiv \ell$

Images and content adapted from https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Demo of RNN for sequence classification

- Demo and illustration figure from PyTorch tutorial https://pytorch.org/tutorials/intermediate/char_rnn_classification_tutorial.html



Sequence Input, Single Output

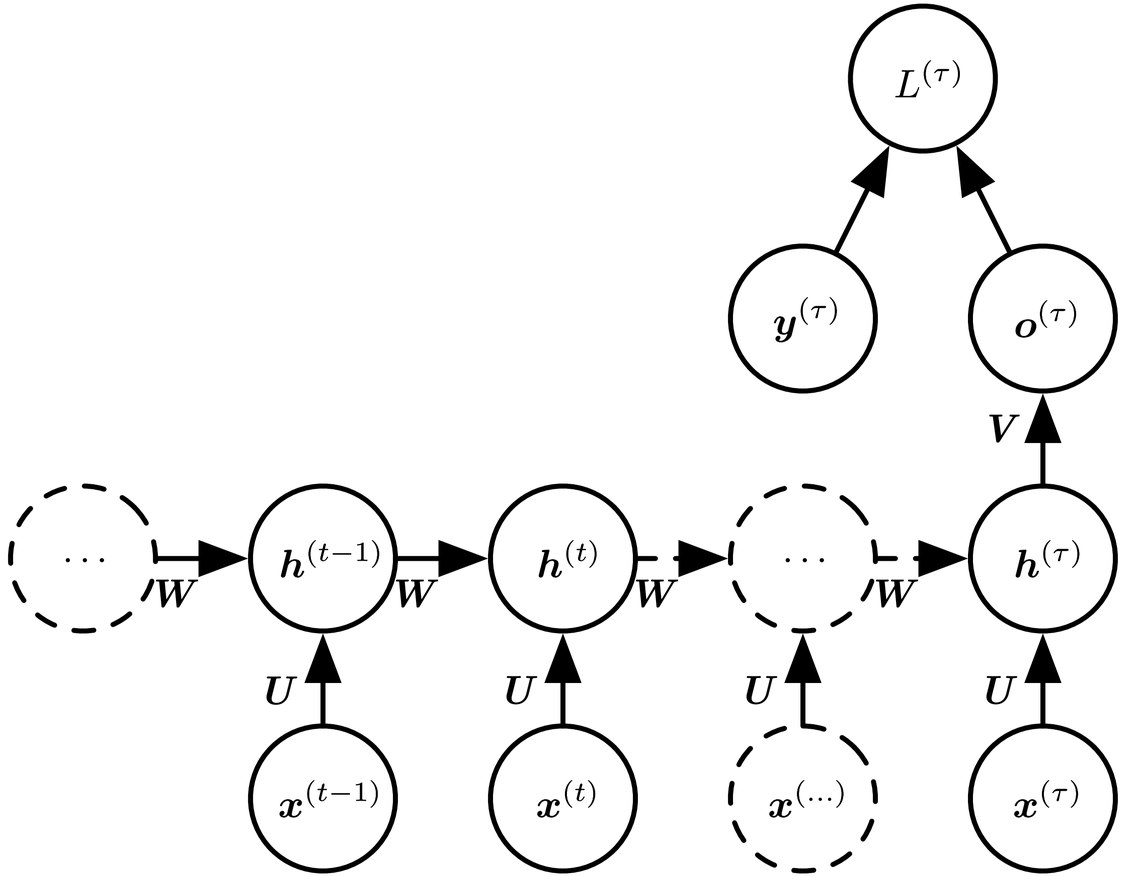


Figure 10.5

Vector to Sequence

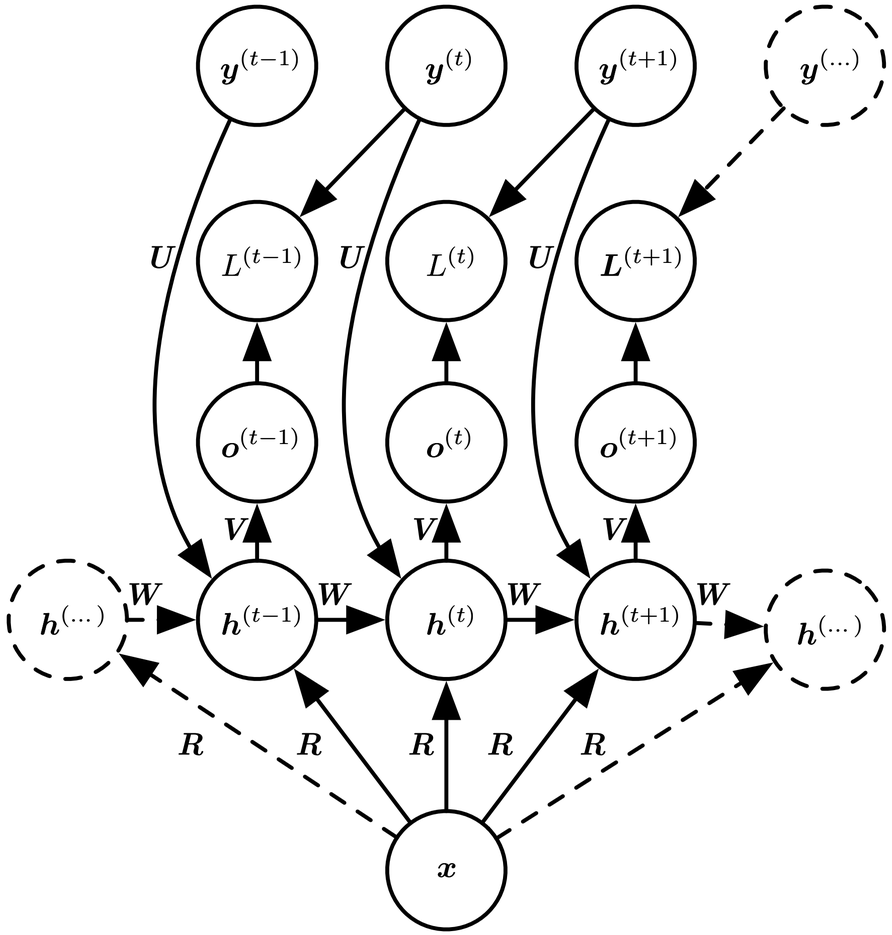


Figure 10.9

Sequence to Sequence Architecture

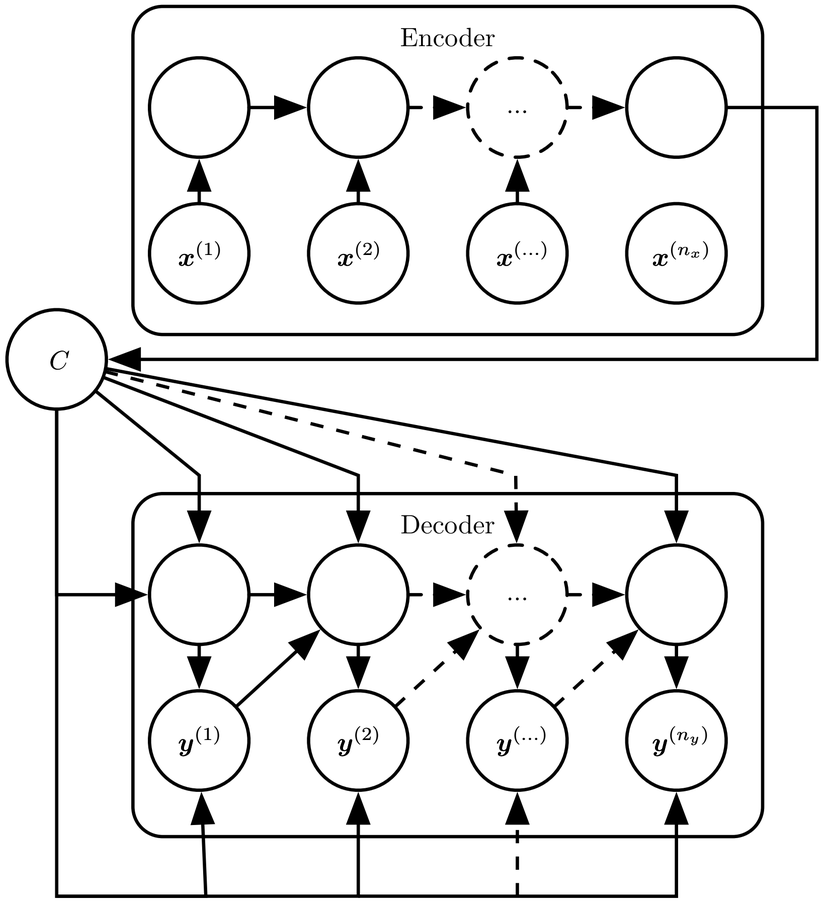


Figure 10.12

Bidirectional RNN

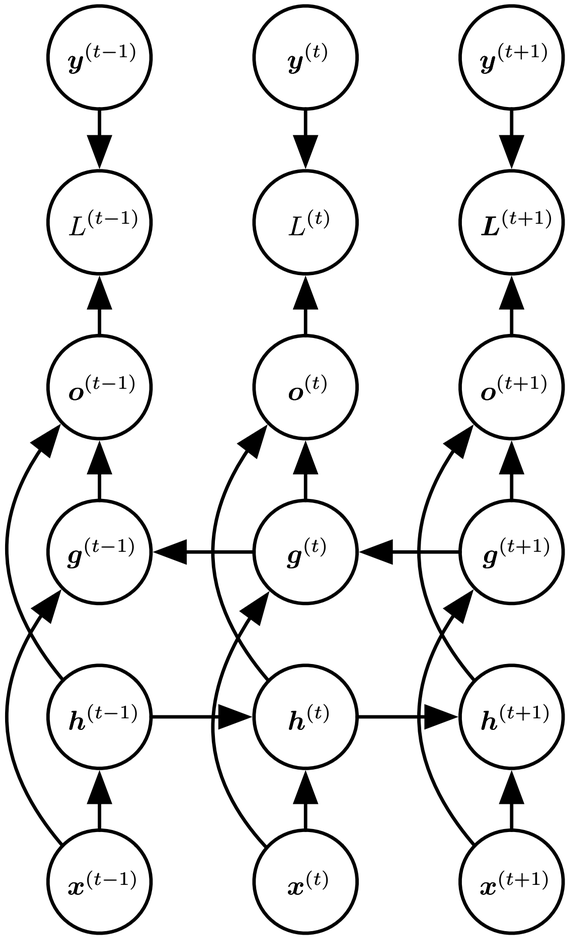
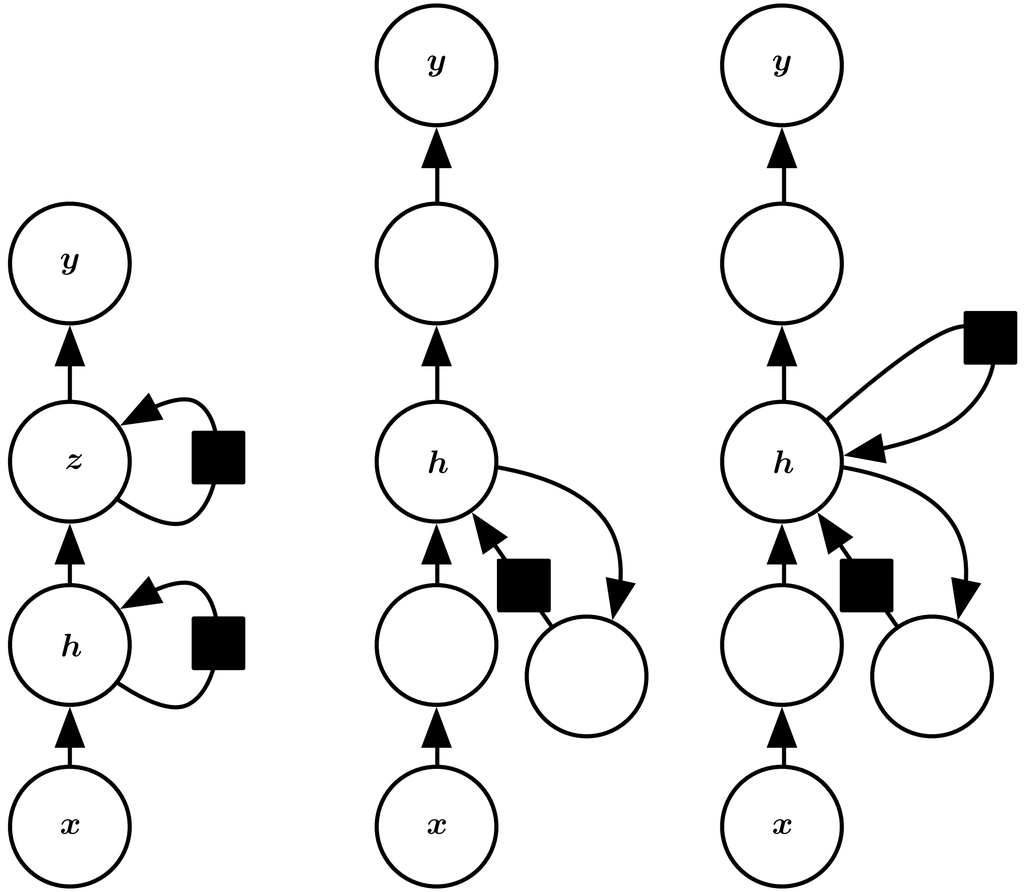


Figure 10.11

Deep RNNs



(a) (b) (c)

Figure 10.13

Vanishing and exploding gradients

- Vanishing or exploding gradients are caused by recursive definition of hidden state
- For simplicity, let's assume that $w_x = 0$ so that we see the core issue.
- The last prediction is as follows:

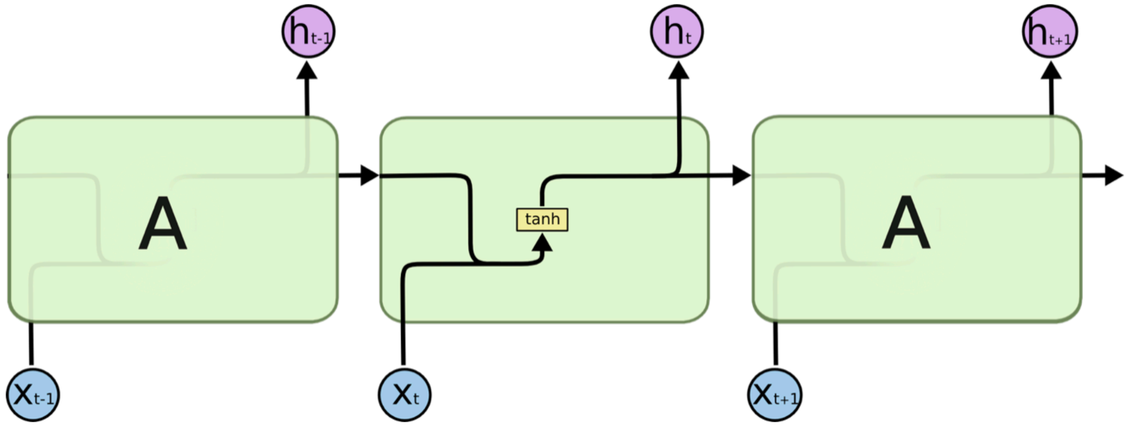
$$\begin{aligned}\hat{y}_L &= w_y h_L + w_x x_L = w_y h_L = w_y(w_h h_{L-1} + w_x x_{L-1}) \\ &= w_y w_h h_{L-1} = w_y w_h^2 h_{L-2} = \cdots = w_y \mathbf{w_h^L} h_0\end{aligned}$$

- The gradient of MSE loss for the last term is:

$$\frac{d}{dw_y}\ell(y, \hat{y}_L) = \frac{d}{dw_y}\|y - \hat{y}_L\|_2^2 = 2(y - \hat{y}_L)\frac{d\hat{y}_L}{dw_y} = 2(y - \hat{y}_L)\mathbf{w_h^L} h_0$$

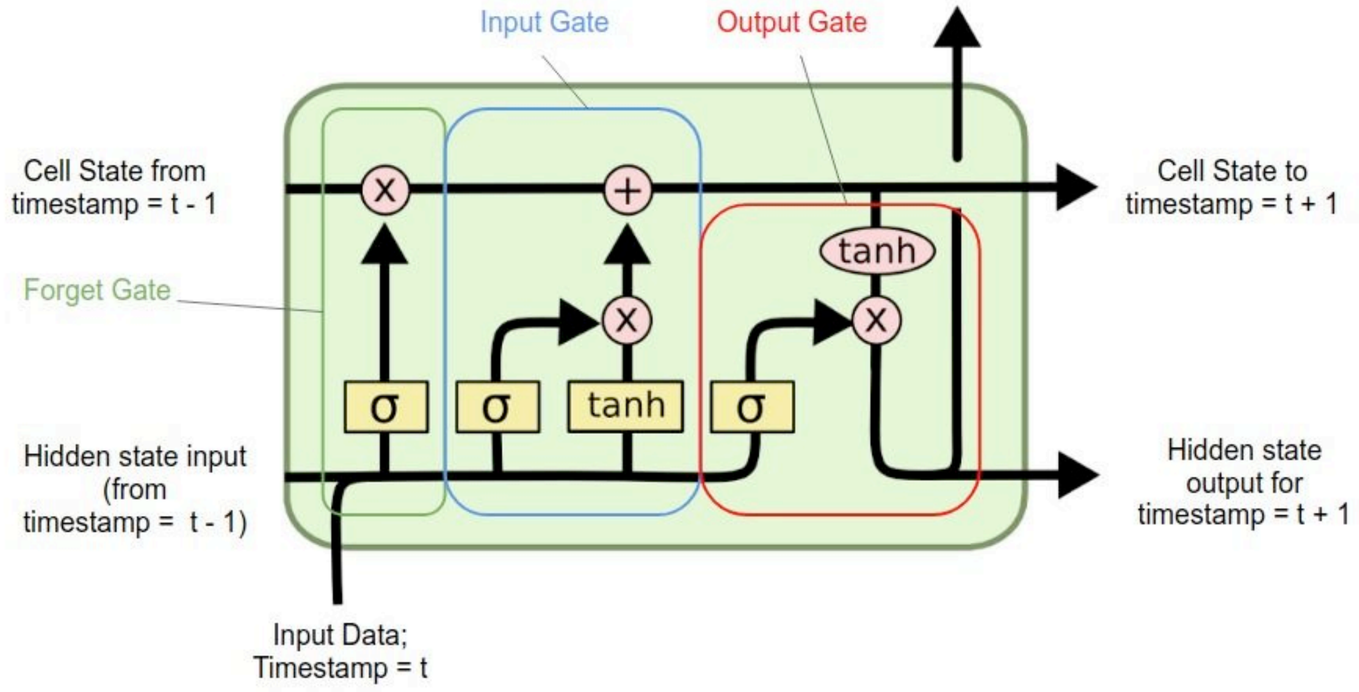
- If $w_h > 1$, then the gradient exponentially increases w.r.t. sequence length L
- If $w_h < 1$, then the gradient exponentially decreases w.r.t. sequence length L

Long short-term memory (LSTM): address vanishing gradient problem and enable learning long-term dependency



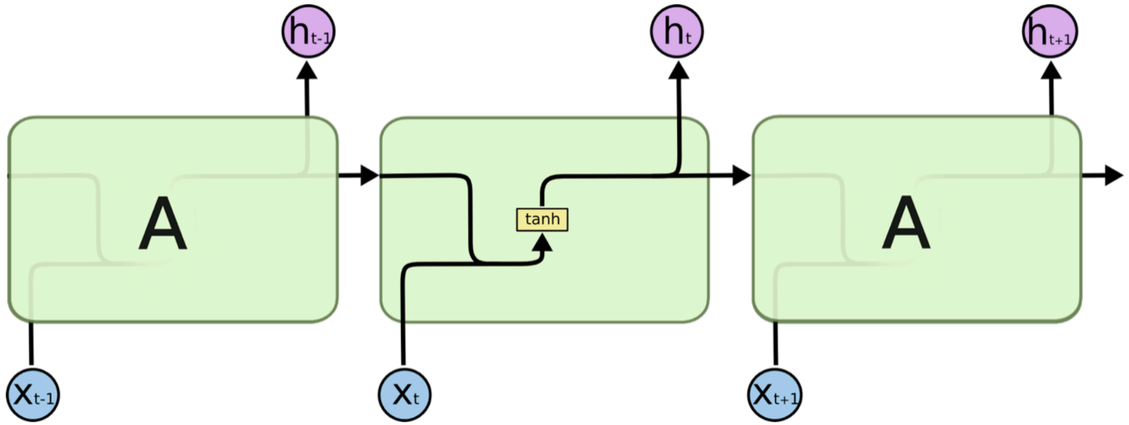
The repeating module in a standard RNN contains a single layer.

Figures from https://colah.github.io/posts/2015-08-Understanding-LSTMs/



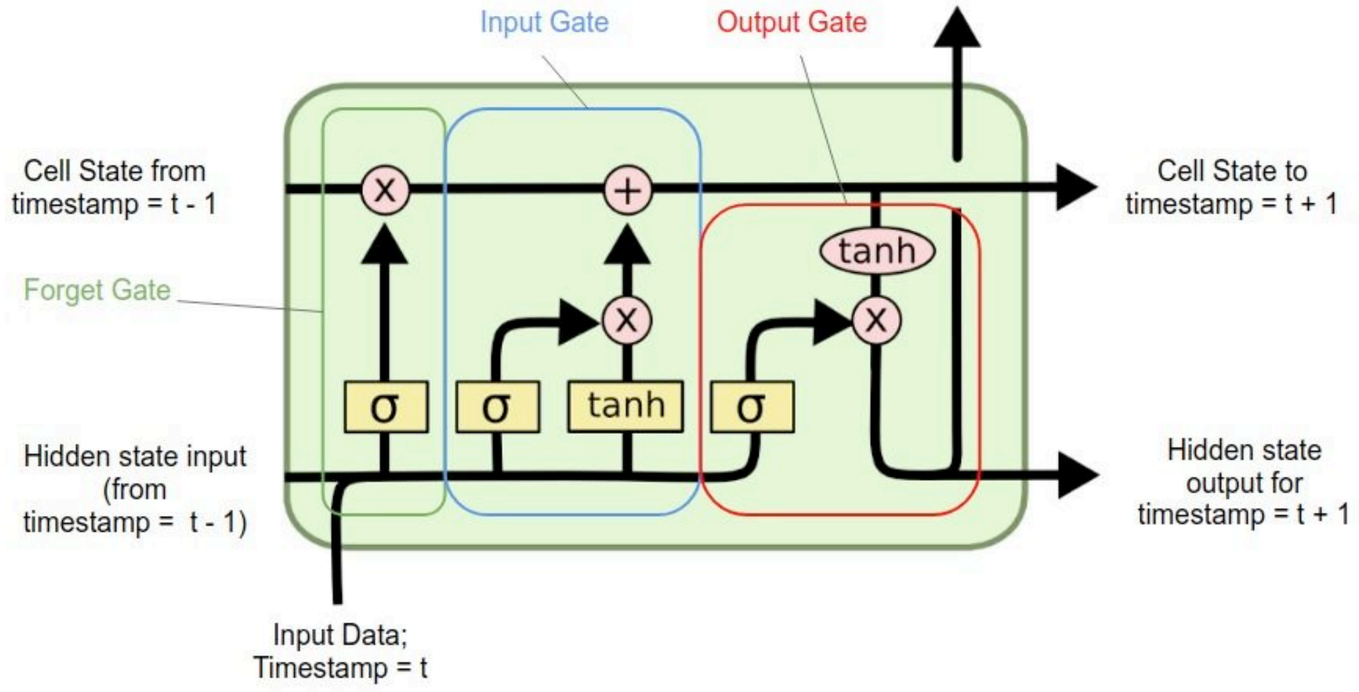
Figures from https://www.turing.com/kb/comprehensive-guide-to-lstm-rnn

Long short-term memory (LSTM): address vanishing gradient problem and enable learning long-term dependency



The repeating module in a standard RNN contains a single layer.

Figures from https://colah.github.io/posts/2015-08-Understanding-LSTMs/



Figures from https://www.turing.com/kb/comprehensive-guide-to-lstm-rnn

- Sequential processing
- Hard to train