Introduction to Deep Learning

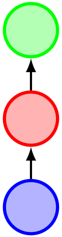
Arijit Mondal
Dept. of Computer Science & Engineering
Indian Institute of Technology Patna
arijit@iitp.ac.in

Recurrent Neural Network

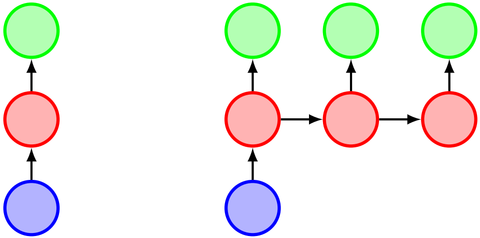
Introduction

- Recurrent neural networks are used for processing sequential data in general
  - Convolution neural network is specialized for image
- Capable of processing variable length input
- Shares parameters across different part of the model
  - Example: ”I went to IIT in 2017” or ”In 2017, I went to IIT”
  - For traditional machine learning models require to learn rules for different positions

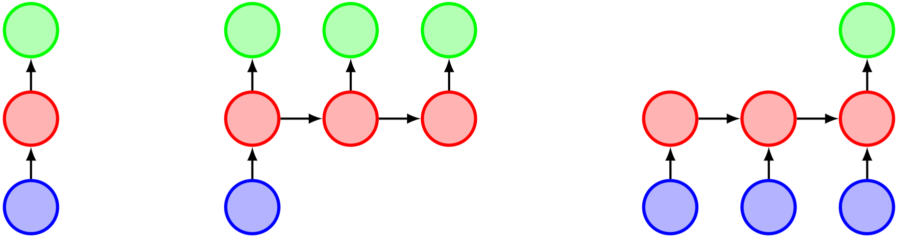
Types of applications



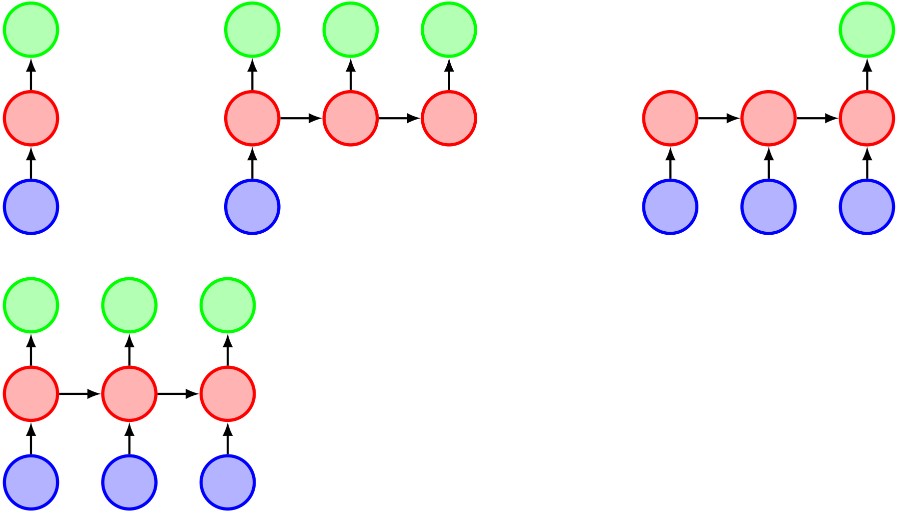
Types of applications



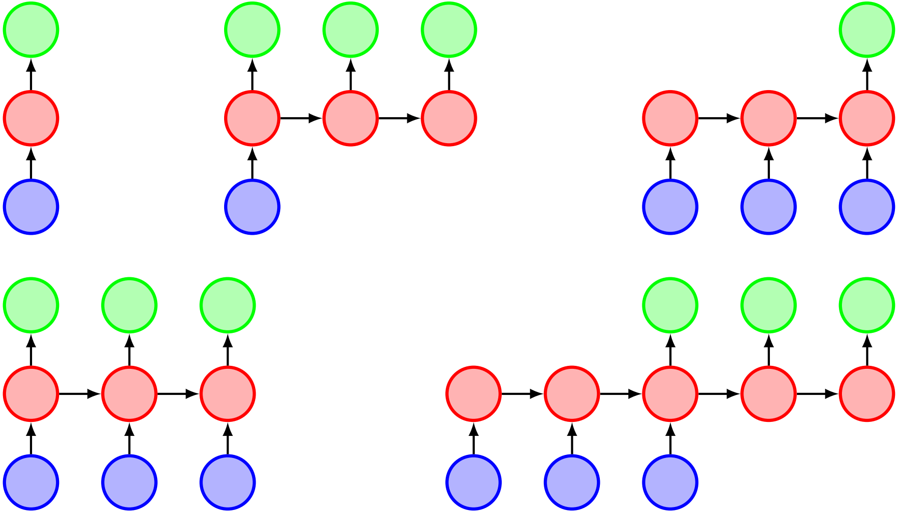
Types of applications



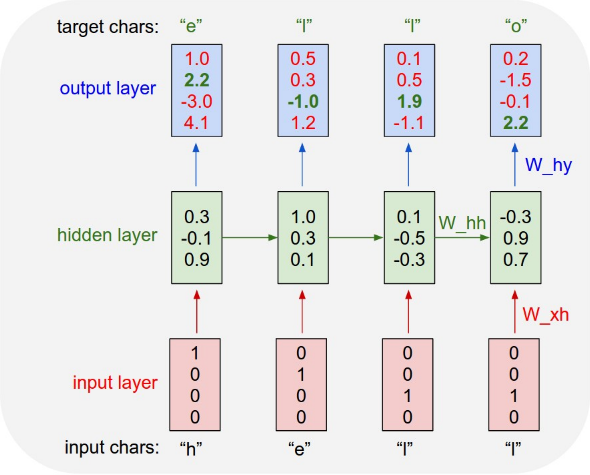
Types of applications



Types of applications



Example



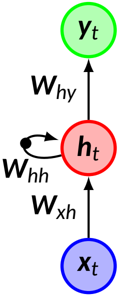
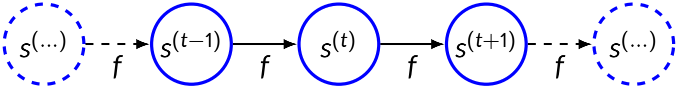


Image source: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

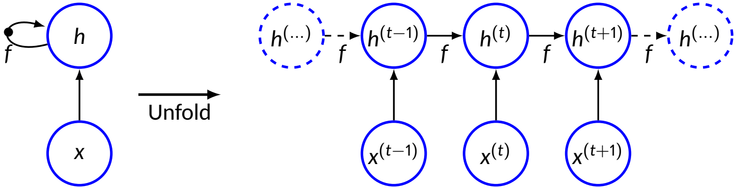
Computational graph

- **Formal way to represent the computation**
- **Unfolding the graph results in sharing of parameters**
- **Consider a system $\boldsymbol{s}^{(t)} = f(\boldsymbol{s}^{(t-1)}, \boldsymbol{\theta})$ where $\boldsymbol{s}^{(t)}$ denotes the state of the system**
  - **It is recurrent**
  - **For finite number of steps, it can be unfolded**
  - **Example: $s^{(3)} = f(s^{(2)}, \theta) = f(f(s^{(1)}, \theta), \theta)$**



System with inputs

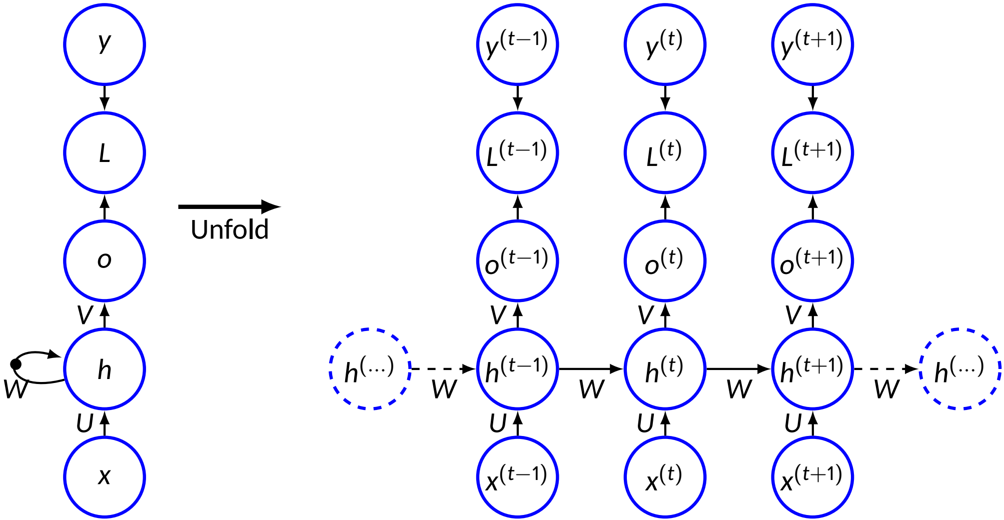
- A system will be represented as $\boldsymbol{s}^{(t)} = f(\boldsymbol{s}^{(t-1)}, \boldsymbol{x}^{(t)}, \boldsymbol{\theta})$
  - A state contains information of whole past sequence
- Usually state is indicated as hidden units such that $\boldsymbol{h}^{(t)} = f(\boldsymbol{h}^{(t-1)}, \boldsymbol{x}^{(t)}, \boldsymbol{\theta})$
- While predicting, network learn $\boldsymbol{h}^{(t)}$ as a kind of lossy summary of past sequence upto t
  - $\boldsymbol{h}^{(t)}$ depends on $(\boldsymbol{x}^{(t)}, \boldsymbol{x}^{(t-1)}, \ldots, \boldsymbol{x}^{(1)})$



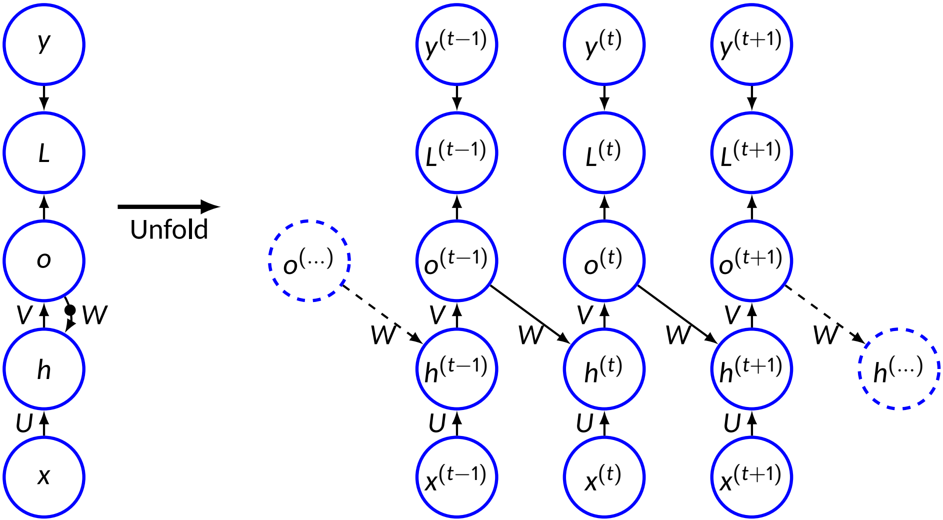
System with inputs (contd.)

- Unfolded recursion after t steps will be $\boldsymbol{h}^{(t)} = g^{(t)}(\boldsymbol{x}^{(t)}, \boldsymbol{x}^{(t-1)}, \ldots, \boldsymbol{x}^{(1)}) = f(\boldsymbol{h}^{(t-1)}, \boldsymbol{x}^{(t)}, \boldsymbol{\theta})$
- Unfolding process has some advantages
  - Regardless of sequence length, learned model has same input size
  - Uses the same transition function f with the same parameters at every time steps
- Can be trained with fewer examples
- Recurrent graph is succinct
- Unfolded graph illustrates the information flow

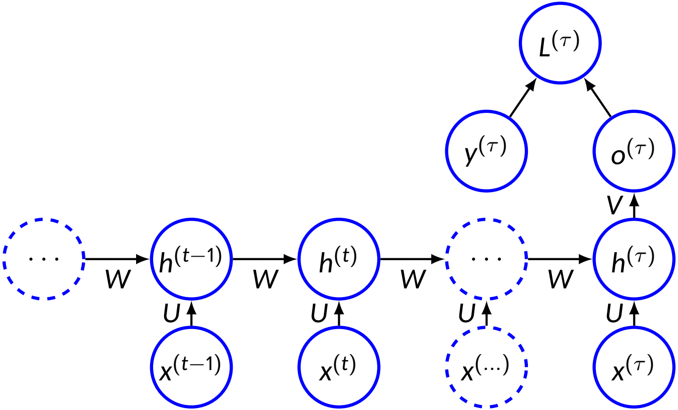
Recurrent connection in hidden units



Output to hidden unit connection



Sequence processing

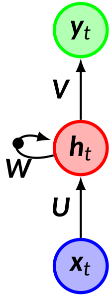


Recurrent neural network

- Function computable by a Turing machine can be computed by such recurrent network of finite size
- tanh is usually chosen as activation function for hidden units
- Output can be considered as discrete, so $\boldsymbol{o}$ gives unnormalized log probabilities
- Forward propagation begins with initial state $\boldsymbol{h}^0$
- So we have,
  - $\boldsymbol{a}^{(t)} = \boldsymbol{b} + \boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\boldsymbol{x}^{(t)}$
  - $\boldsymbol{h}^{(t)} = \tanh(\boldsymbol{a}^{(t)})$
  - $\boldsymbol{o}^{(t)} = \boldsymbol{c} + \boldsymbol{V}\boldsymbol{h}^{(t)}$
  - $\hat{\boldsymbol{y}}^{(t)} = \text{softmax}(\boldsymbol{o}^{(t)})$
- Input and output have the same length

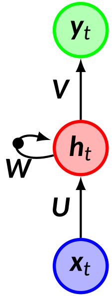
Backpropagation through time

- The network will be unfolded and gradient will be back propagated
- Number of stages need to be decided
- Issue in gradient computation
  - Vanishing gradients
  - Exploding gradients



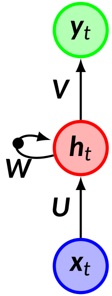
Backpropagation through time

- **The network will be unfolded and gradient will be back propagated**
- **Number of stages need to be decided**
- **Issue in gradient computation**
  - **Vanishing gradients**
  - **Exploding gradients**
- **Loss function**
  - $E_t = \frac{1}{2}\sum_{k=1}^{\text{out}}(\hat{y}_k - y_k)^2,$



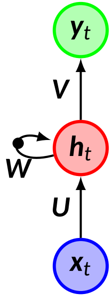
Backpropagation through time

- The network will be unfolded and gradient will be back propagated
- Number of stages need to be decided
- Issue in gradient computation
  - Vanishing gradients
  - Exploding gradients
- Loss function
  - $E_t = \frac{1}{2}\sum_{k=1}^{\text{out}}(\hat{y}_k - y_k)^2, E = \frac{1}{2}\sum_{t=1}^{\tau}\sum_{k=1}^{\text{out}}(\hat{y}_{tk} - y_{tk})^2$



Backpropagation through time

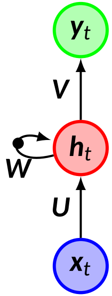
- The network will be unfolded and gradient will be back propagated
- Number of stages need to be decided
- Issue in gradient computation
    - Vanishing gradients
    - Exploding gradients
- Loss function
    - $E_t = \frac{1}{2}\sum_{k=1}^{\text{out}}(\hat{y}_k - y_k)^2$, $E = \frac{1}{2}\sum_{t=1}^{\tau}\sum_{k=1}^{\text{out}}(\hat{y}_{tk} - y_{tk})^2$
    - $E = -\sum_{t=1}^{\tau}\sum_{k=1}^{\text{out}}[\hat{y}_{tk}\ln y_{tk} + (1-\hat{y}_{tk})\ln(1-y_{tk})]$



Backpropagation through time

- **Basic equations**

$$
\begin{aligned}
\boldsymbol{h}_t &= \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}\phi(\boldsymbol{h}_{t-1}) \\
\boldsymbol{y}_t &= \boldsymbol{V}\phi(\boldsymbol{h}_t)
\end{aligned}
$$



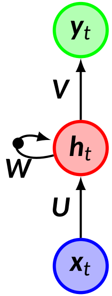
Backpropagation through time

- **Basic equations**

$$\begin{aligned} \boldsymbol{h}_t &= \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}\phi(\boldsymbol{h}_{t-1}) \\ \boldsymbol{y}_t &= \boldsymbol{V}\phi(\boldsymbol{h}_t) \end{aligned}$$

- **Gradient**

$$\frac{\partial E}{\partial \boldsymbol{W}}$$



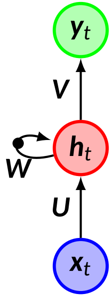
Backpropagation through time

- **Basic equations**

$$\begin{aligned} \boldsymbol{h}_t &= \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}\phi(\boldsymbol{h}_{t-1}) \\ \boldsymbol{y}_t &= \boldsymbol{V}\phi(\boldsymbol{h}_t) \end{aligned}$$

- **Gradient**

$$\frac{\partial E}{\partial \boldsymbol{W}} = \sum_{t=1}^{\tau} \frac{\partial E_t}{\partial \boldsymbol{W}}$$



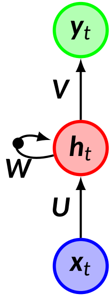
Backpropagation through time

- **Basic equations**

$$\begin{aligned} \boldsymbol{h}_t &= \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}\phi(\boldsymbol{h}_{t-1}) \\ \boldsymbol{y}_t &= \boldsymbol{V}\phi(\boldsymbol{h}_t) \end{aligned}$$

- **Gradient**

$$\frac{\partial E}{\partial \boldsymbol{W}} = \sum_{t=1}^{\tau} \frac{\partial E_t}{\partial \boldsymbol{W}} = \sum_{t=1}^{\tau} \sum_{k=1}^{t} \frac{\partial E_t}{\partial \boldsymbol{y}_t}$$



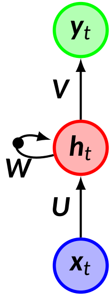
Backpropagation through time

- **Basic equations**

$$\begin{aligned} \boldsymbol{h}_t &= \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}\phi(\boldsymbol{h}_{t-1}) \\ \boldsymbol{y}_t &= \boldsymbol{V}\phi(\boldsymbol{h}_t) \end{aligned}$$

- **Gradient**

$$\frac{\partial E}{\partial \boldsymbol{W}} = \sum_{t=1}^{\tau} \frac{\partial E_t}{\partial \boldsymbol{W}} = \sum_{t=1}^{\tau} \sum_{k=1}^{t} \frac{\partial E_t}{\partial \boldsymbol{y}_t} \frac{\partial \boldsymbol{y}_t}{\partial \boldsymbol{h}_t}$$



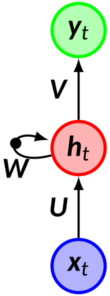
Backpropagation through time

- **Basic equations**

$$\begin{aligned} \boldsymbol{h}_t &= \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}\phi(\boldsymbol{h}_{t-1}) \\ \boldsymbol{y}_t &= \boldsymbol{V}\phi(\boldsymbol{h}_t) \end{aligned}$$

- **Gradient**

$$\frac{\partial E}{\partial \boldsymbol{W}} = \sum_{t=1}^{\tau} \frac{\partial E_t}{\partial \boldsymbol{W}} = \sum_{t=1}^{\tau} \sum_{k=1}^{t} \frac{\partial E_t}{\partial \boldsymbol{y}_t} \frac{\partial \boldsymbol{y}_t}{\partial \boldsymbol{h}_t} \frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_k}$$



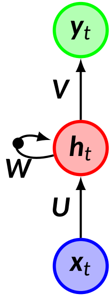
Backpropagation through time

- **Basic equations**

$$\boldsymbol{h}_t = \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}\phi(\boldsymbol{h}_{t-1})$$
$$\boldsymbol{y}_t = \boldsymbol{V}\phi(\boldsymbol{h}_t)$$

- **Gradient**

$$\frac{\partial E}{\partial \boldsymbol{W}} = \sum_{t=1}^{\tau} \frac{\partial E_t}{\partial \boldsymbol{W}} = \sum_{t=1}^{\tau} \sum_{k=1}^{t} \frac{\partial E_t}{\partial \boldsymbol{y}_t} \frac{\partial \boldsymbol{y}_t}{\partial \boldsymbol{h}_t} \frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_k} \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{W}}$$



Backpropagation through time

- **Basic equations**

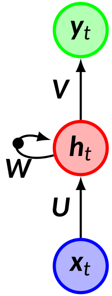
$$\begin{aligned}\boldsymbol{h}_t &= \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}\phi(\boldsymbol{h}_{t-1}) \\ \boldsymbol{y}_t &= \boldsymbol{V}\phi(\boldsymbol{h}_t)\end{aligned}$$

- **Gradient**

$$\frac{\partial E}{\partial \boldsymbol{W}} = \sum_{t=1}^{\tau} \frac{\partial E_t}{\partial \boldsymbol{W}} = \sum_{t=1}^{\tau}\sum_{k=1}^{t} \frac{\partial E_t}{\partial \boldsymbol{y}_t}\frac{\partial \boldsymbol{y}_t}{\partial \boldsymbol{h}_t}\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_k}\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{W}}$$

- **Now we have,**

$$\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_k}$$



Backpropagation through time

- **Basic equations**

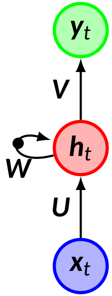
$$\begin{aligned} \boldsymbol{h}_t &= \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}\phi(\boldsymbol{h}_{t-1}) \\ \boldsymbol{y}_t &= \boldsymbol{V}\phi(\boldsymbol{h}_t) \end{aligned}$$

- **Gradient**

$$\frac{\partial E}{\partial \boldsymbol{W}} = \sum_{t=1}^{\tau} \frac{\partial E_t}{\partial \boldsymbol{W}} = \sum_{t=1}^{\tau} \sum_{k=1}^{t} \frac{\partial E_t}{\partial \boldsymbol{y}_t} \frac{\partial \boldsymbol{y}_t}{\partial \boldsymbol{h}_t} \frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_k} \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{W}}$$

- **Now we have,**

$$\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_k} = \prod_{i=k+1}^{t} \frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{h}_{i-1}}$$



Backpropagation through time

- **Basic equations**

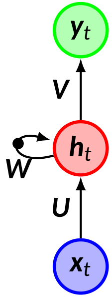
$$\begin{aligned} \boldsymbol{h}_t &= \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}\phi(\boldsymbol{h}_{t-1}) \\ \boldsymbol{y}_t &= \boldsymbol{V}\phi(\boldsymbol{h}_t) \end{aligned}$$

- **Gradient**

$$\frac{\partial E}{\partial \boldsymbol{W}} = \sum_{t=1}^{\tau} \frac{\partial E_t}{\partial \boldsymbol{W}} = \sum_{t=1}^{\tau}\sum_{k=1}^{t} \frac{\partial E_t}{\partial \boldsymbol{y}_t}\frac{\partial \boldsymbol{y}_t}{\partial \boldsymbol{h}_t}\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_k}\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{W}}$$

- **Now we have,**

$$\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_k} = \prod_{i=k+1}^{t} \frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{h}_{i-1}} = \prod_{i=k+1}^{t} \boldsymbol{W}^T \textbf{diag}[\phi'(\boldsymbol{h}_{i-1})]$$

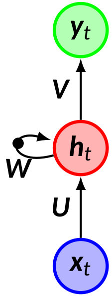


Backpropagation through time

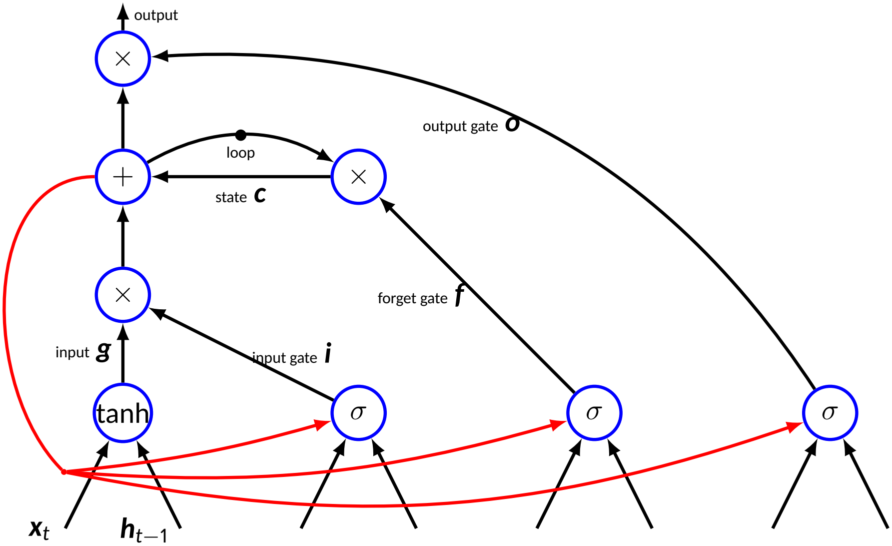
- **Issues in gradient**

$$\left\|\frac{\partial \mathbf{h}_i}{\partial \mathbf{h}_{i-1}}\right\| \leq \left\|\mathbf{W}^T\right\| \left\|\mathbf{diag}[\phi'(\mathbf{h}_{i-1})]\right\| \leq \lambda_{\mathbf{w}} \lambda_\phi$$

$$\left\|\frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_k}\right\| \leq (\lambda_{\mathbf{w}} \lambda_\phi)^{t-k}$$



LSTM


output
output gate $\boldsymbol{o}$
loop
state $\boldsymbol{c}$
forget gate $\boldsymbol{f}$
input $\boldsymbol{g}$
input gate $\boldsymbol{i}$
tanh
σ
σ
σ
$\boldsymbol{x}_t$
$\boldsymbol{h}_{t-1}$

LSTM

- **Mathematical relation**

$$\boldsymbol{i}_t = \sigma(\theta_{xi}\boldsymbol{x}_t + \theta_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}_i)$$

$$\boldsymbol{f}_t = \sigma(\theta_{xf}\boldsymbol{x}_t + \theta_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{b}_f)$$

$$\boldsymbol{o}_t = \sigma(\theta_{xo}\boldsymbol{x}_t + \theta_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{b}_o)$$

$$\boldsymbol{g}_t = \tanh(\theta_{xg}\boldsymbol{x}_t + \theta_{hg}\boldsymbol{h}_{t-1} + \boldsymbol{b}_g)$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t)$$

LSTM

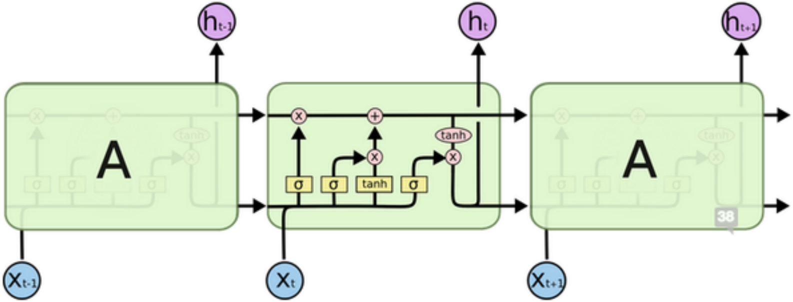


Image source:colah.github.io