Recurrent Neural Network

Outline

- Recurrent Neural Network (RNN)
  - Training of RNNs
    - BPTT
  - Visualization of RNN through Feed-Forward Neural Network
  - Usage
  - Problems with RNNs

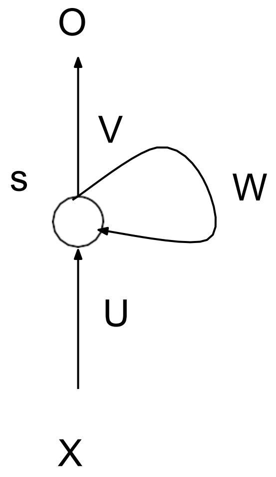
Recurrent Neural Network (RNN)

Basic definition:

A neural network with feedback connections.

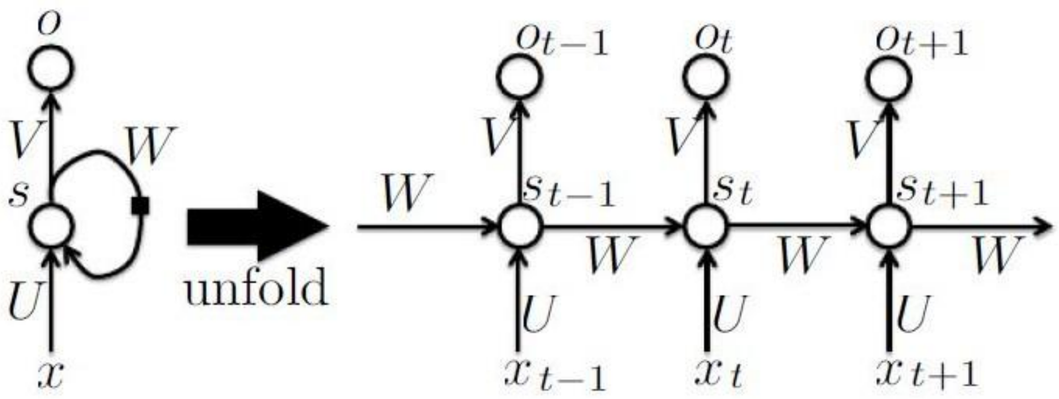
X: Input
O: Ouput
S: Hidden state

Weights: [U,V,W]
Learned during training



Recurrent Neural Network (RNN)

- Enable networks to do temporal processing
- Good at learning sequences
- Acts as memory unit



Memory

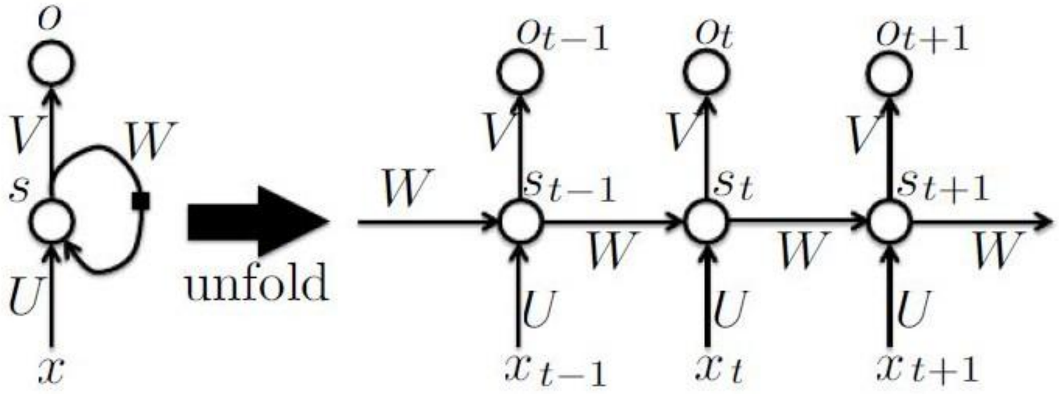
$$
\begin{aligned}
\boldsymbol{a}_t &= \boldsymbol{b} + \boldsymbol{W}\boldsymbol{s}_{t-1} + \boldsymbol{U}\boldsymbol{x}_t \\
\boldsymbol{s}_t &= \tanh(\boldsymbol{a}_t) \\
\boldsymbol{o}_t &= \boldsymbol{c} + \boldsymbol{V}\boldsymbol{s}_t \\
\boldsymbol{p}_t &= \mathrm{softmax}(\boldsymbol{o}_t)
\end{aligned}
$$

RNN - Example 1

Part-of-speech tagging:

- Given a sentence X, tag each word its corresponding grammatical class.

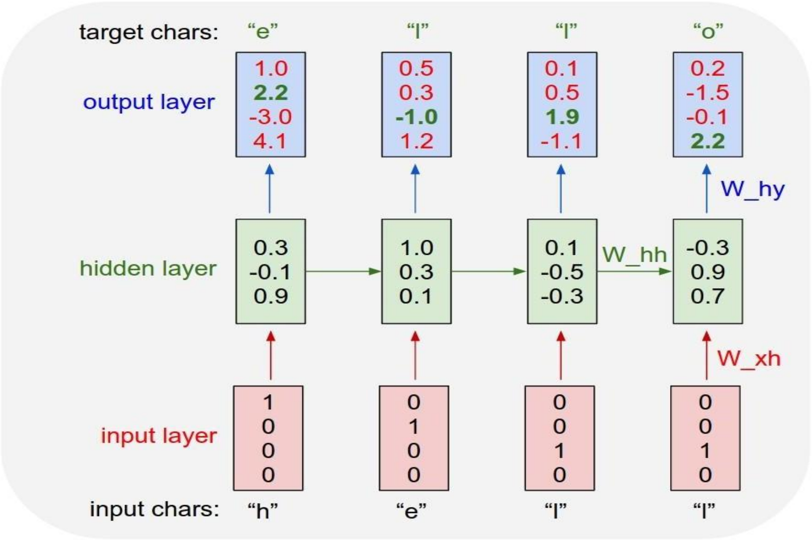
O = [PRP VBP NNS]



X = [I love mangoes]

RNN - Example 2

Character level language model:



- Given previous and current characters, predict next the character in the sequence.

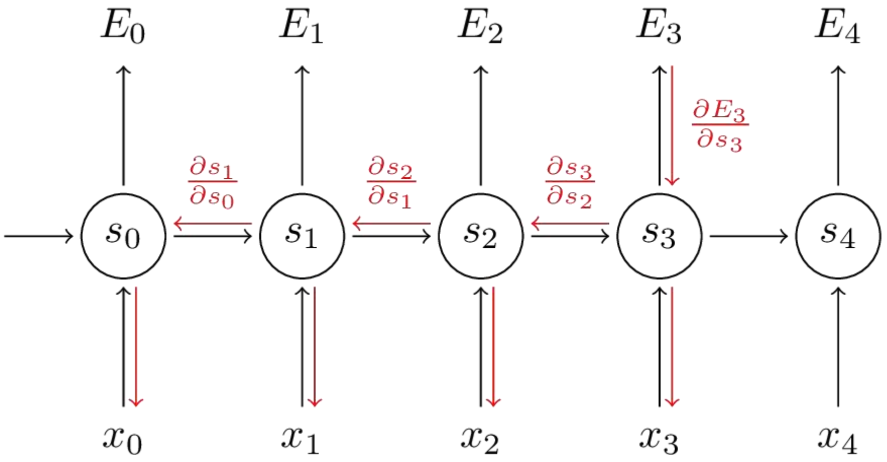
Let

- **Vocabulary: [h,e,l,o]**
- **One-hot representations**
  - **h = [1 0 0 0]**
  - **e = [0 1 0 0]**
  - **l = [0 0 1 0]**
  - **o = [0 0 0 1]**

Training of RNNs

How to train RNNs?

- Typical FFN
  - Backpropagation algorithm
- RNNs
  - A variant of backpropagation algorithm namely **Back-Propagation Through Time (BPTT)**.

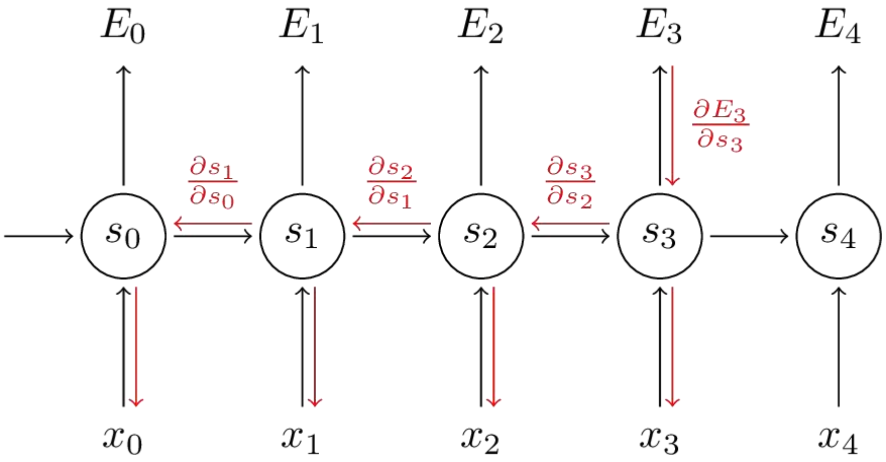


BackPropagation Through Time (BPTT)

Error for an instance = Sum of errors at each time step of the instance

Gradient of error

$$\frac{\partial E}{\partial W} = \sum_t \frac{\partial E_t}{\partial W}$$



BackPropagation Through Time (BPTT)

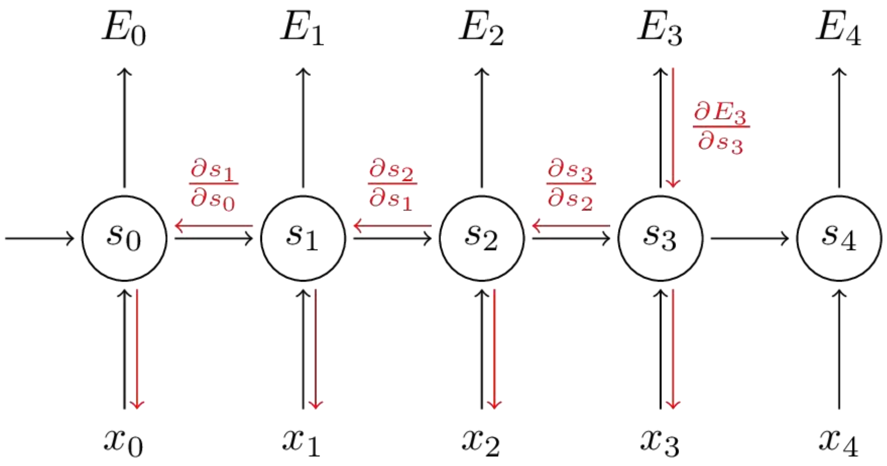
For V

$$\frac{\partial E_3}{\partial V} = \frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial V}$$

For W (Similarly for U)

$$\frac{\partial E_3}{\partial W} = \frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial s_3}\frac{\partial s_3}{\partial W}$$

$$\frac{\partial E_3}{\partial W} = \sum_{k=0}^{3}\frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial s_3}\frac{\partial s_3}{\partial s_k}\frac{\partial s_k}{\partial W}$$



When to use RNNs

Usage

- Depends on the problems that we aim to solve.
- Typically good for sequence processings.
- Some sort of memorization is required.

Bit reverse problem

- Problem definition:
  - **Problem 1:** Reverse a binary digit.
    - 0 → 1   and   1 → 0
  - **Problem 2:** Reverse a sequence of binary digits.
    - 0 1 0 1 0 0 1   →   1 0 1 0 1 1 0
    - Sequence: Fixed or Variable length
  - **Problem 3:** Reverse a sequence of bits over time.
    - 0 1 0 1 0 0 1   →   1 0 1 0 1 1 0
  - **Problem 4:** Reverse a bit if the current i/p and previous o/p are same.

Input sequence 1 1 0 0 1 0 0 0 1 1

Output sequence 1 0 1 0 1 0 1 0 1 0

Data

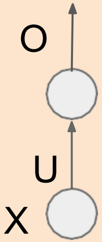
Let

- **Problem 1**
  - I/p dimension: **1 bit** O/p dimension: **1 bit**
- **Problem 2**
  - Fixed
    - I/p dimension: **10 bit** O/p dimension: **10 bit**
  - Variable: Pad each sequence upto max sequence length: **10**
    - Padding value: **-1**
    - I/p dimension: **10 bit** O/p dimension: **10 bit**
- **Problem 3 & 4**
  - Dimension of each element of I/p (X) : **1 bit**
  - Dimension of each element of O/p (O) : **1 bit**
  - Sequence length : **10**

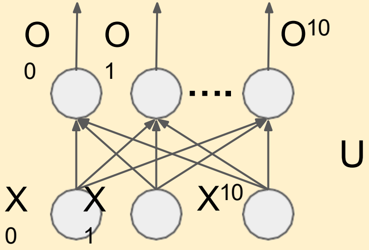
No. of I/p neurons = I/p dimension
No. of O/p neurons = O/p dimension

Network Architecture

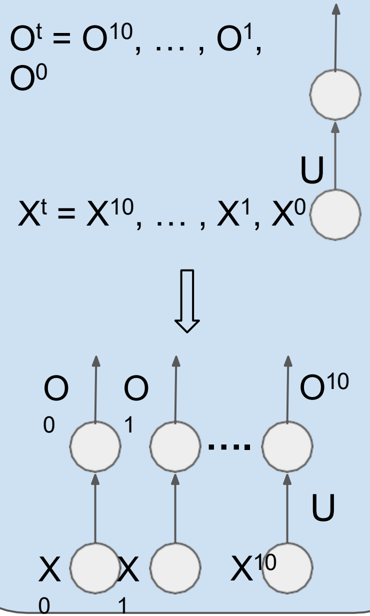
Problem 1:
- I/p neurons = 1
- O/p neurons = 1



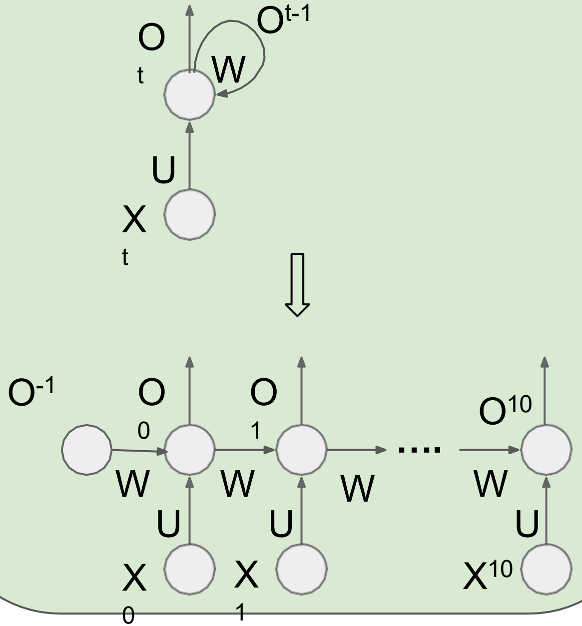
Problem 2: Fixed & Variable
- I/p neurons = 10
- O/p neurons = 10



Problem 3:
- I/p neurons = 1
- O/p neurons = 1
- Seq len = 10



Problem 4:
- I/p neurons = 1
- O/p neurons = 1
- Seq len = 10



Different configurations of RNNs

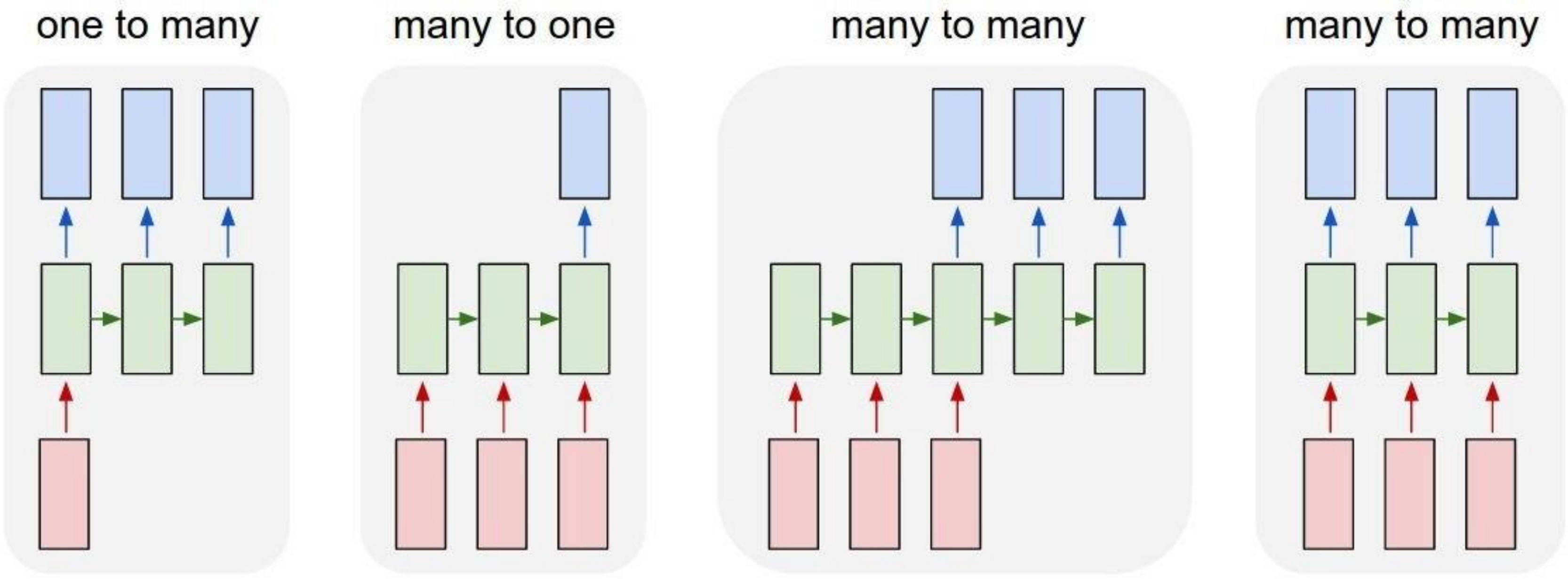
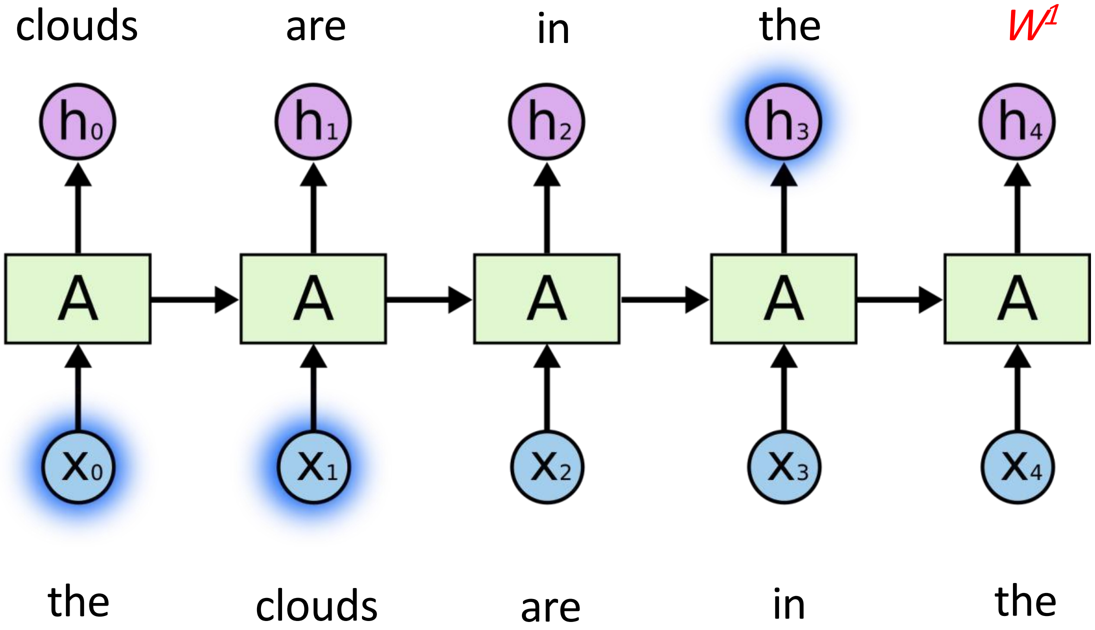


Image Captioning | **Sentiment Analysis** | **Machine Translation** | **Language modelling**

Problems with RNNs

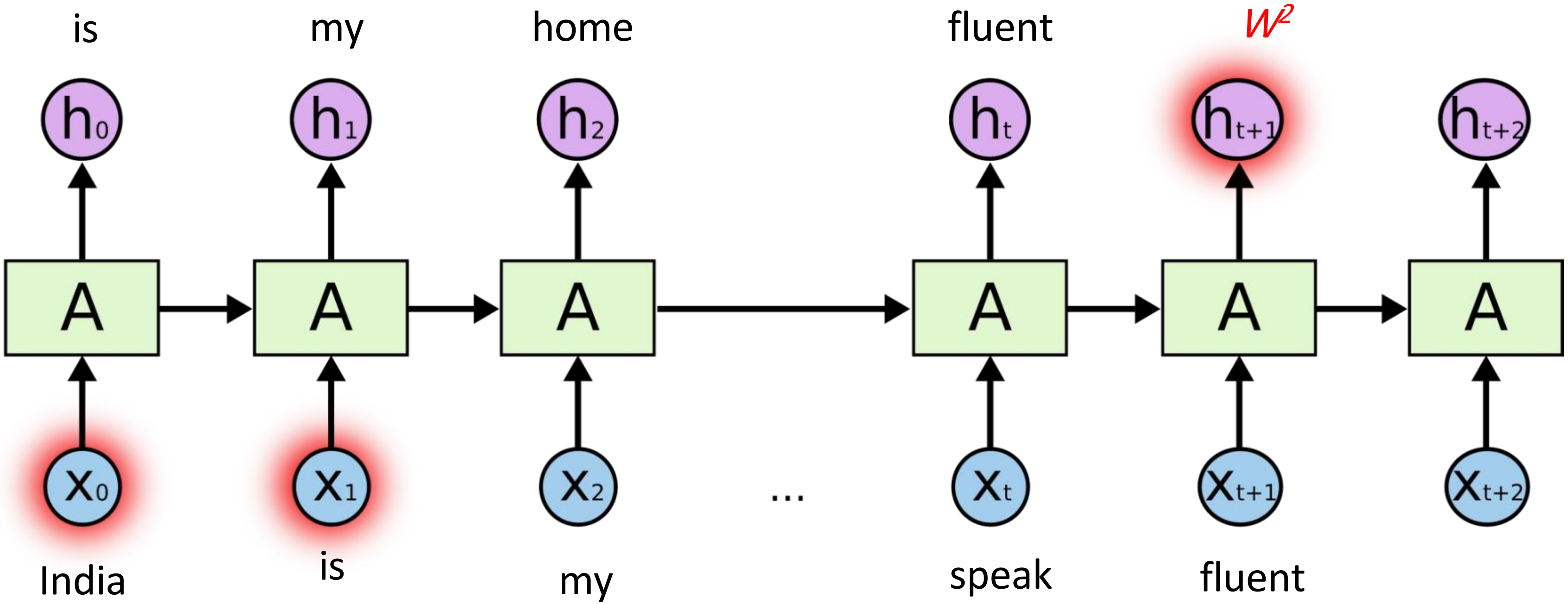
Language modelling: Example - 1

- “the clouds are in the *sky*”



Language modelling: Example - 2

- “India is my home country. I can speak fluent *Hindi.*”



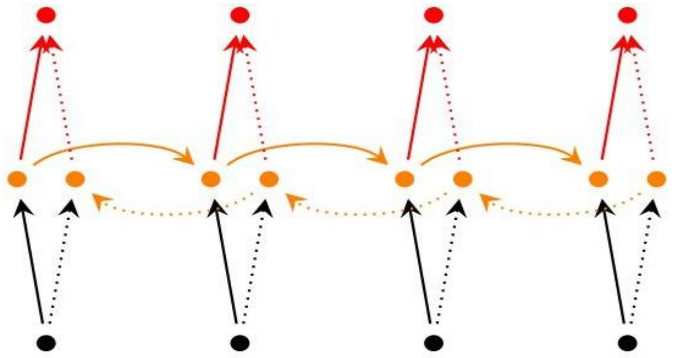
Vanishing/Exploding gradients

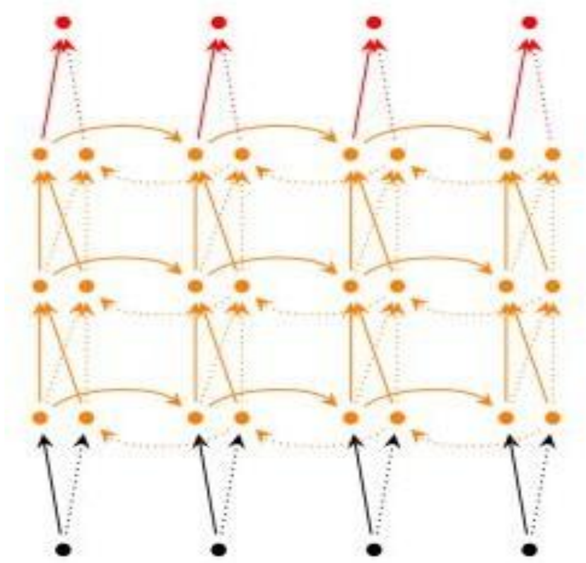
- Cue word for the prediction
  - Example 1: ***sky*** → ***clouds*** [3 units apart]
  - Example 2: ***hindi*** → ***India*** [9 units apart]

- As the sequence length increases, it becomes hard for RNNs to learn "long-term dependencies."
  - **Vanishing gradients:** If weights are small, gradient shrinks exponentially. Network stops learning.
  - **Exploding gradients:** If weights are large, gradient grows exponentially. Weights fluctuate and become unstable.

RNN extensions

- Bi-directional RNN
- Deep (Bi-directional) RNN





Thank You!