MUDSS

Data Science Society

Advanced Workshop 6: Recurrent Neural Network

OVERVIEW

- Introduction to RNN
- RNN Model
- RNN Training
- Vanishing Gradient & LSTM
- Shortcomings and Adaptations
- Code Example

ASSUMED KNOWLEDGE

- Basic Neural Network (week 11)
  - Feedforward
  - Backpropagation
  - Gradient Descent (week 8)
- Activation Function - SoftMax, TanH
- Cross-Entropy Loss (week 9)
- PyTorch fundamental (week 7)

WHY RNN ?

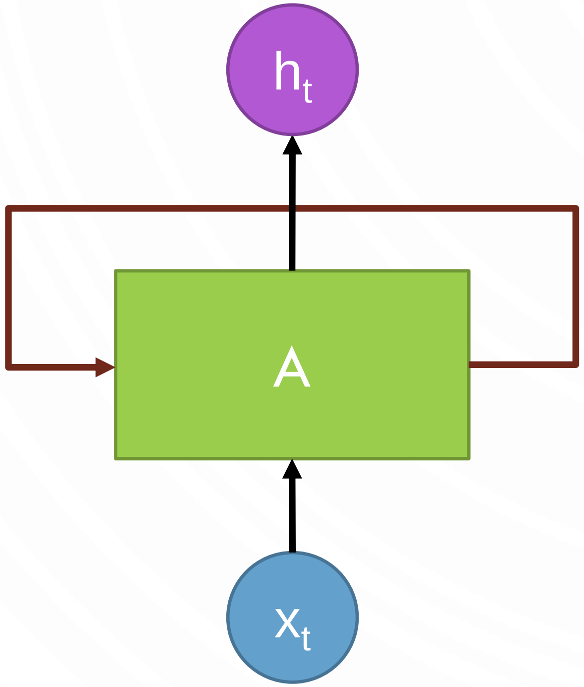
WHY RNN ?

"He grew up in France, and he can speak fluent ______"

Traditional NN → ???

RNN → French !

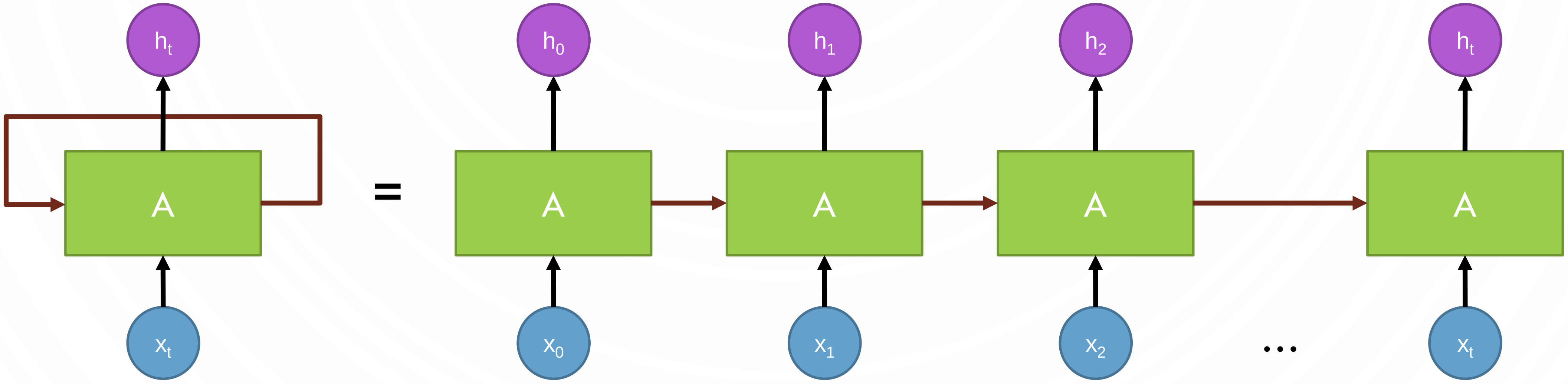
WHAT IS A RNN ?



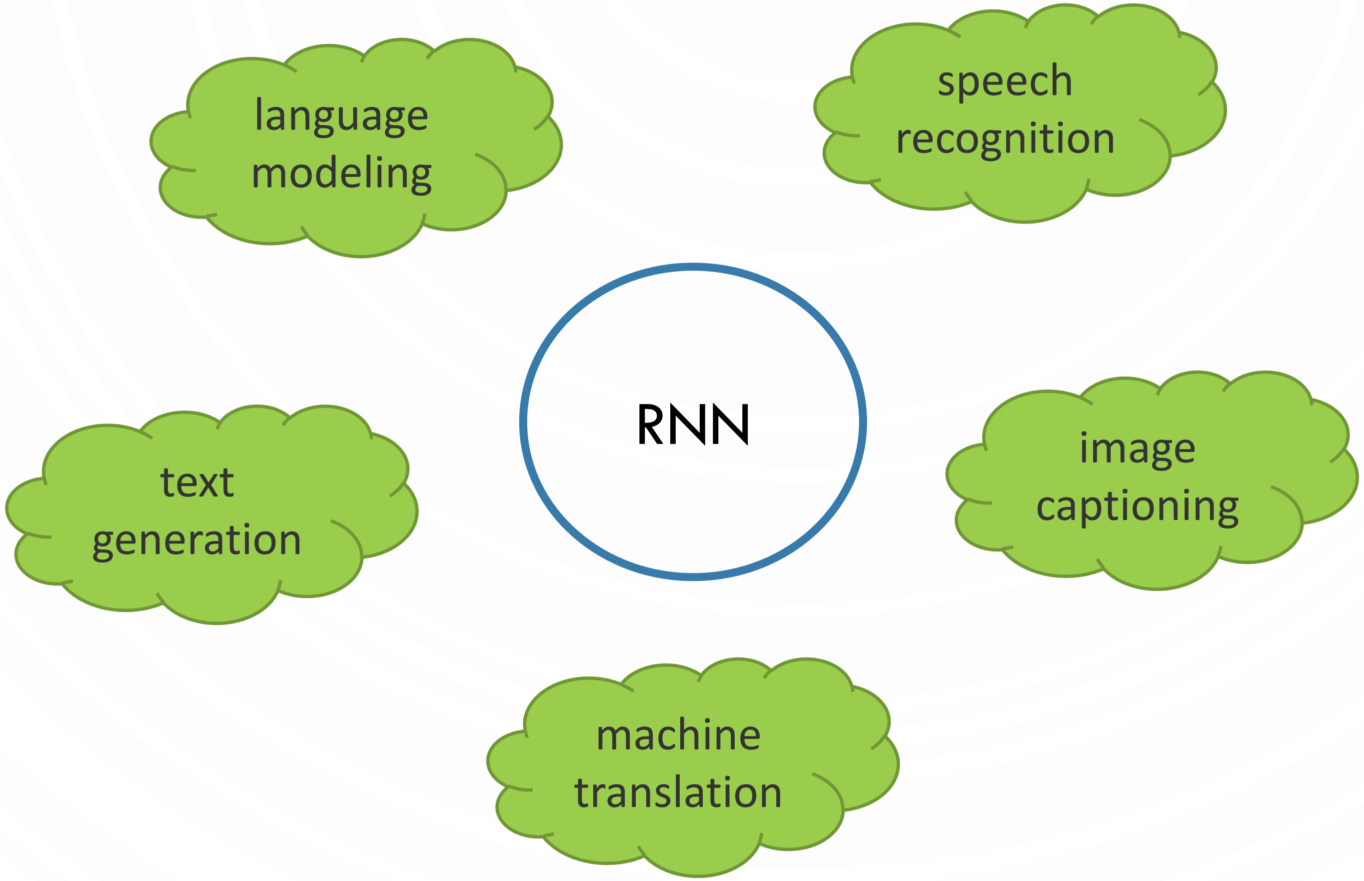
Recurrent Neural Network

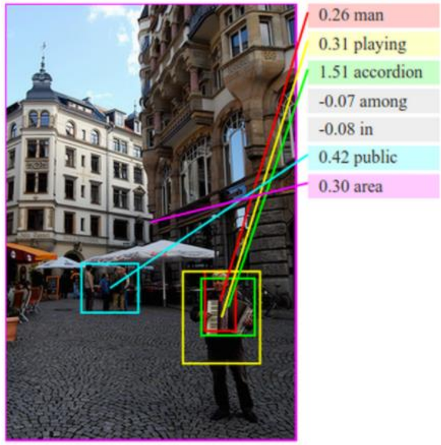
- Recurrent: perform the same task for each element in sequence
- Neuron: multiple copies of the same network
- Network: self-connected
- Memory: hidden nodes
- Arbitrary long? Just in theory

WHAT IS A RNN ?

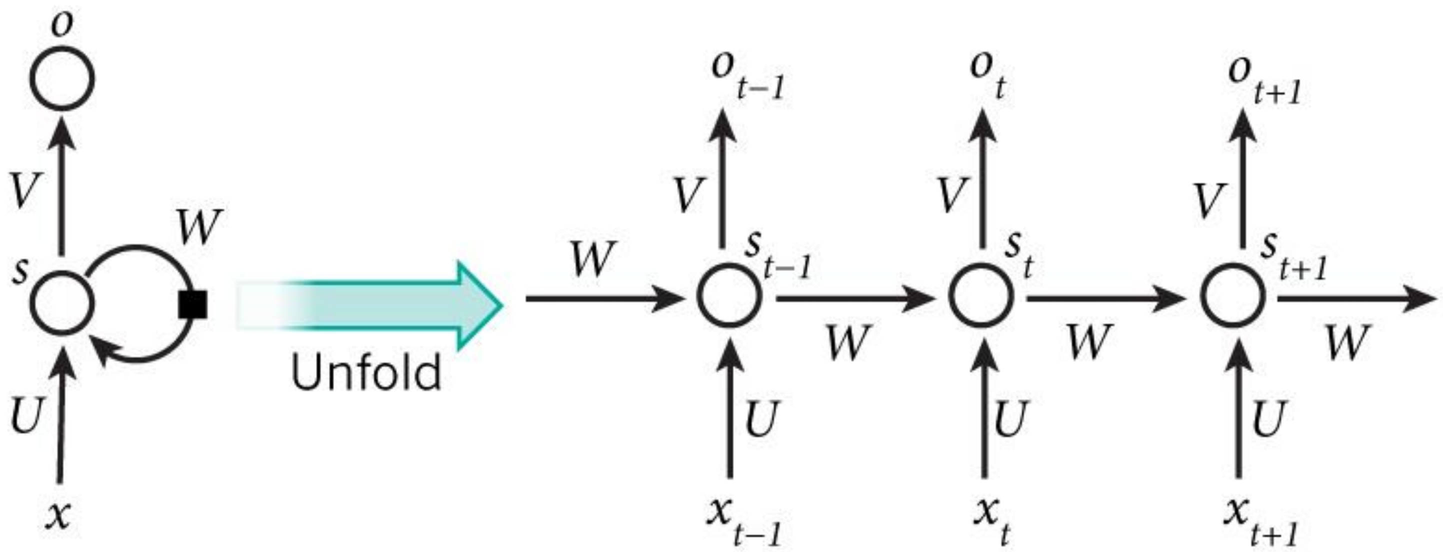


RNN APPLICATIONS





RNN MODEL

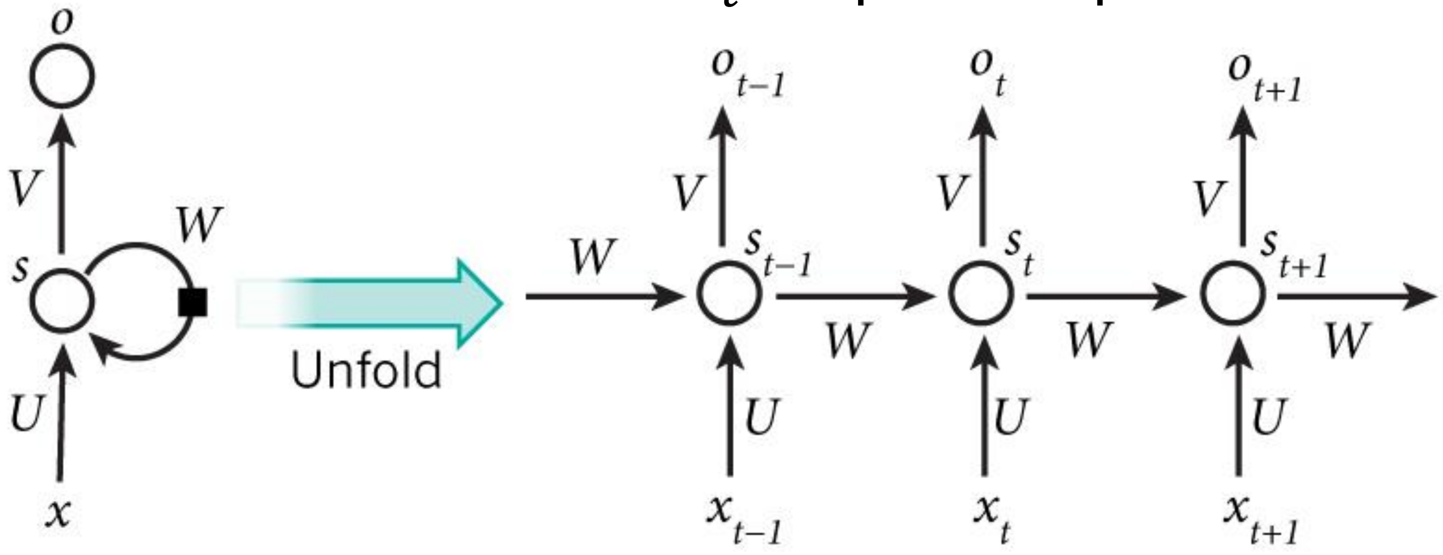


RNN MODEL

$O_t = softmax(Vs_t)$

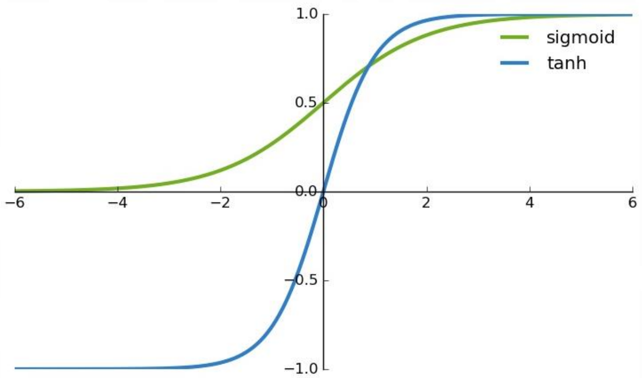
o_t: output at step t

U, V, W:
weight matrix



x_t: input at time step t (one-hot vector)

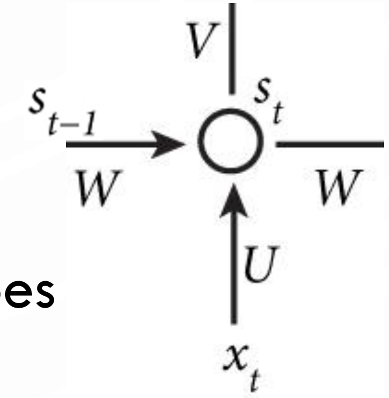
RNN MODEL



s_t: hidden state at time step t

$$s_t = tanh(Ux_t + Ws_{t-1})$$

s_{-1} is typically initialised to all zeroes



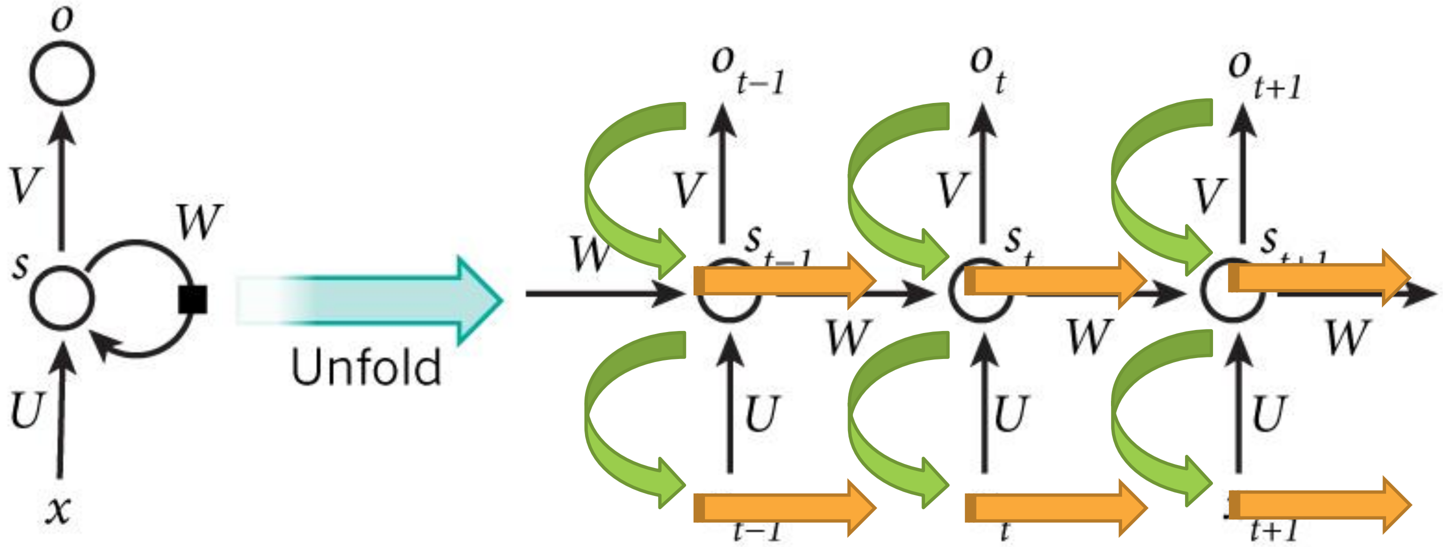
Memory of network

Limitation:

Vanishing Gradient

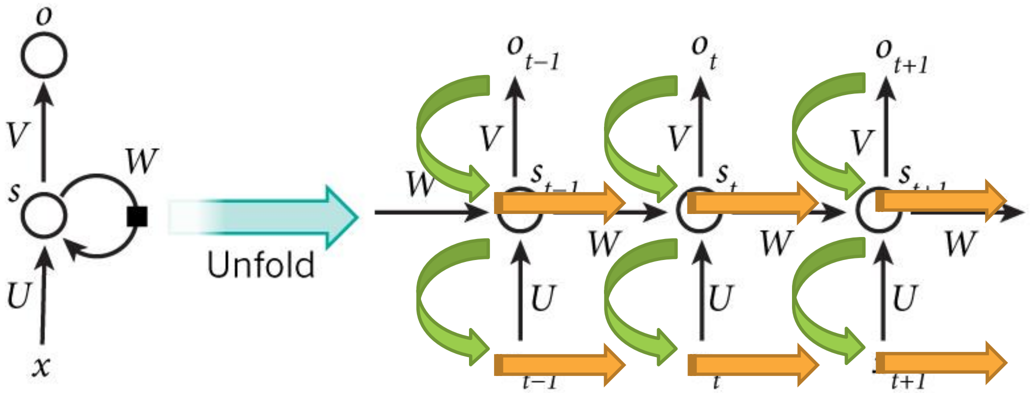
RNN TRAINING - BPTT

Backpropagation + Gradient Descent



Backpropagation Through Time

RNN TRAINING - BPTT



$$L_t = g(O_t)$$

$$L = \sum_{t=0}^{T} L_t$$

Loss Function g: cross-entropy Loss

$$s_t = tanh(Ux_t + Ws_{t-1})$$

$$O_t = softmax(Vs_t)$$

Take derivative on V, W, U respectively

$$\frac{\partial L}{\partial V} = \sum_{t=0}^{T} \frac{\partial L_t}{\partial V}$$

$$\frac{\partial L}{\partial W} = \sum_{t=0}^{T} \sum_{k=0}^{t} \frac{\partial L_t}{\partial s_t} \left(\prod_{j=k+1}^{t} \frac{\partial s_j}{\partial s_{j-1}} \right) \frac{\partial s_k}{\partial W}$$

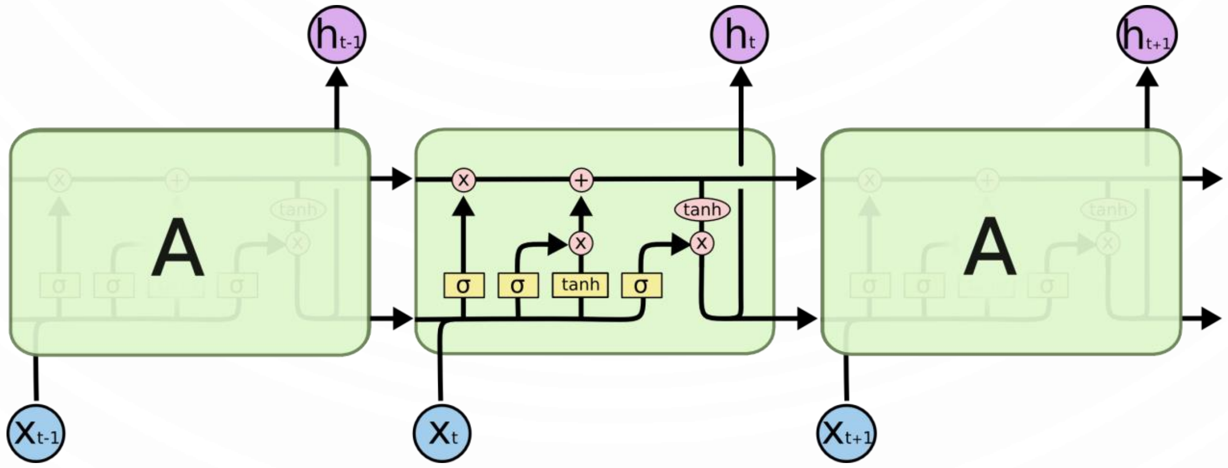
$$\frac{\partial L}{\partial U} = \sum_{t=0}^{T} \sum_{k=0}^{t} \frac{\partial L_t}{\partial s_t} \left(\prod_{j=k+1}^{t} \frac{\partial s_j}{\partial s_{j-1}} \right) \frac{\partial s_k}{\partial U}$$

BPTT　　Vanishing Gradient Problem

LSTM - LONG SHORT TERM MEMORY NETWORK

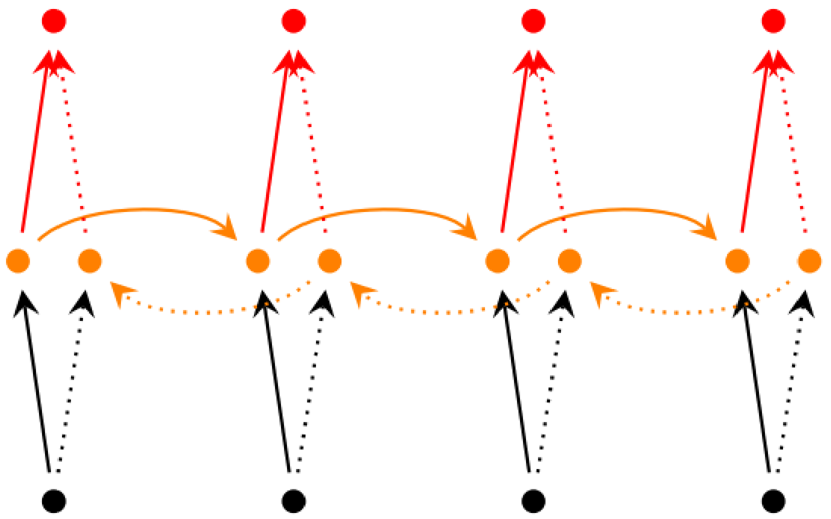
In theory, RNNs are absolutely capable of handling "long-term dependencies."
Sadly, in practice, RNNs don't seem to be able to learn them.

** [Hochreiter (1991) [German]](#) and [Bengio, et al. (1994)](#)*

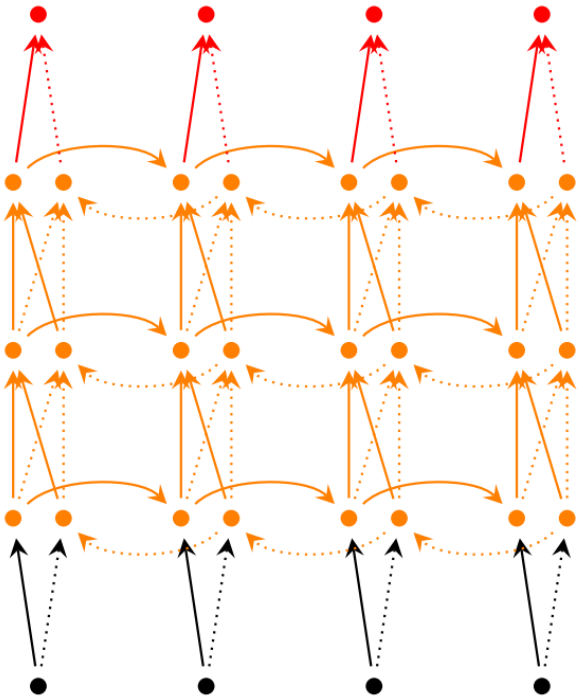


http://colah.github.io/posts/2015-08-Understanding-LSTMs/

SHORTCOMINGS & ADAPTATIONS



Bidirectional RNN



Deep Bidirectional RNN

CODE EXAMPLE

MUDSS
Data Science Society

THANK YOU!

References:

- Recurrent Neural Networks Tutorial, Part 1 – Introduction to RNNs – WildML
- Understanding LSTM Networks -- colah's blog
- NLP From Scratch: Classifying Names with a Character-Level RNN — PyTorch Tutorials
- RNN/LSTM BPTT detailed derivation and gradient vanishing problem analysis - ZhiHu