Deep Learning - 2

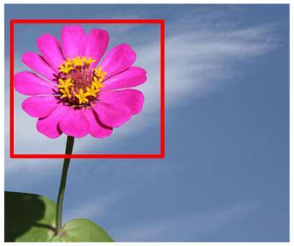
Presentation Title

H M Sabbir Ahmad

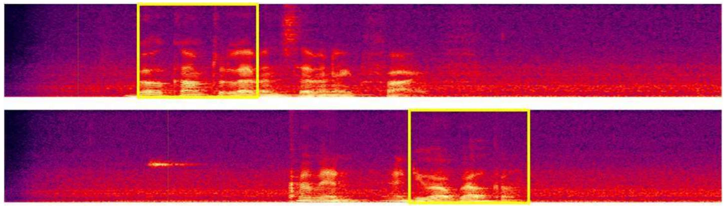
Phd, Systems Engineering.

09/12/2024

The need for *shift invariance*



=



- In many problems the *location* of a pattern is not important
  - Only the presence of the pattern
- Conventional MLPs are sensitive to the location of the pattern
  - Moving it by one component results in an entirely different input that the MLP won't recognize
- Requirement: Network must be *shift invariant*

Slide credit: Bhiksha Raj

Modelling Series

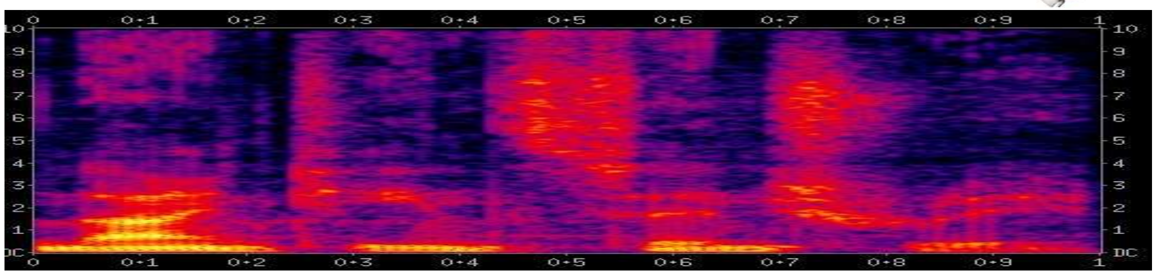
- In many situations one must consider a *series* of inputs to produce an output
  - Outputs too may be a series

- Examples: ..

Slide credit: Bhiksha Raj

What did I say?

“To be” or not “to be”??



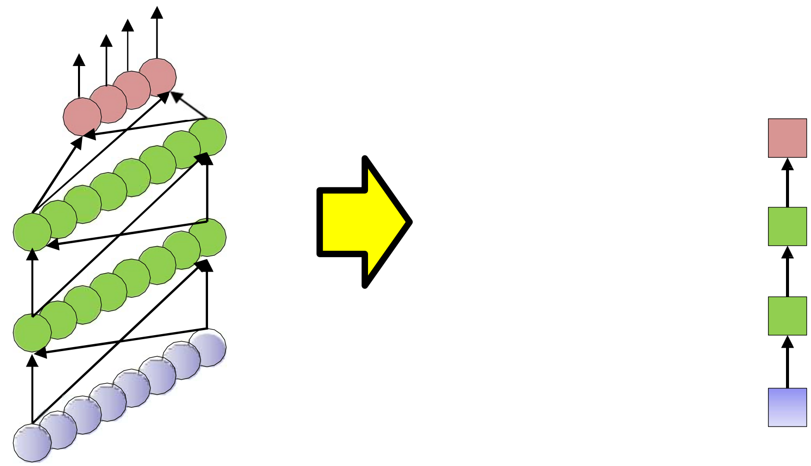
- Speech Recognition
  - Analyze a series of spectral vectors, determine what was said
- Note: Inputs are sequences of vectors. Output is a classification result

Slide credit: Bhiksha Raj

These are classification and prediction problems

- Consider a sequence of inputs
  - Input vectors
- Produce one or more outputs
- This can be done with neural networks
  - Obviously

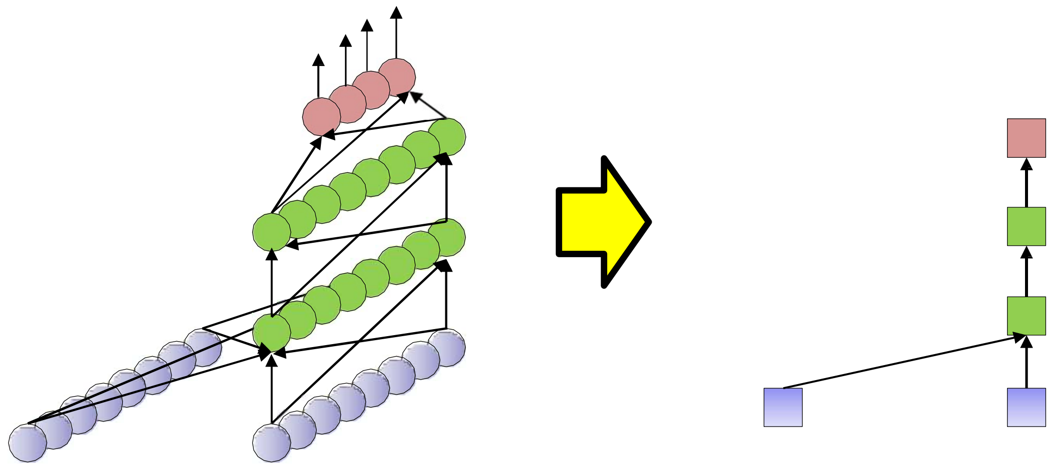
Representational shortcut



- Input at each time is a *vector*
- Each layer has many neurons
  - Output layer too may have many neurons
- But will represent everything by simple boxes
  - Each box actually represents an entire *layer with many units*

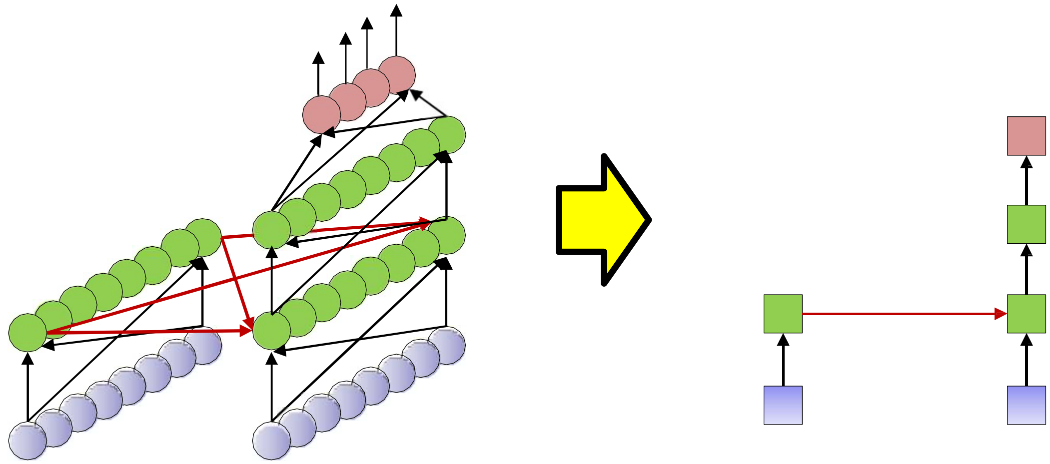
Slide credit: Bhiksha Raj

Representational shortcut



- Input at each time is a *vector*
- Each layer has many neurons
  - Output layer too may have many neurons
- But will represent everything by simple boxes
  - Each box actually represents an entire *layer with many units*

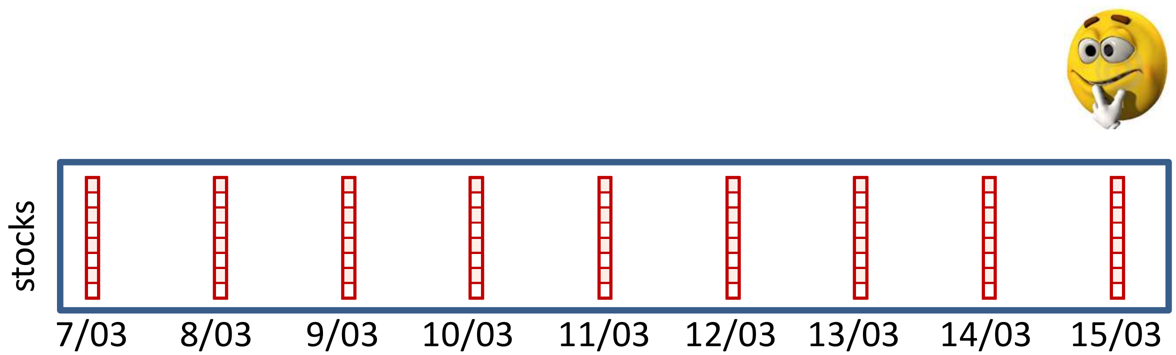
Representational shortcut



- Input at each time is a *vector*
- Each layer has many neurons
  - Output layer too may have many neurons
- But will represent everything as simple boxes
  - Each box actually represents an entire *layer with many units*

Slide credit: Bhiksha Raj

The stock prediction problem…



- Stock market
  - Must consider the series of stock values in the past several days to decide if it is wise to invest today

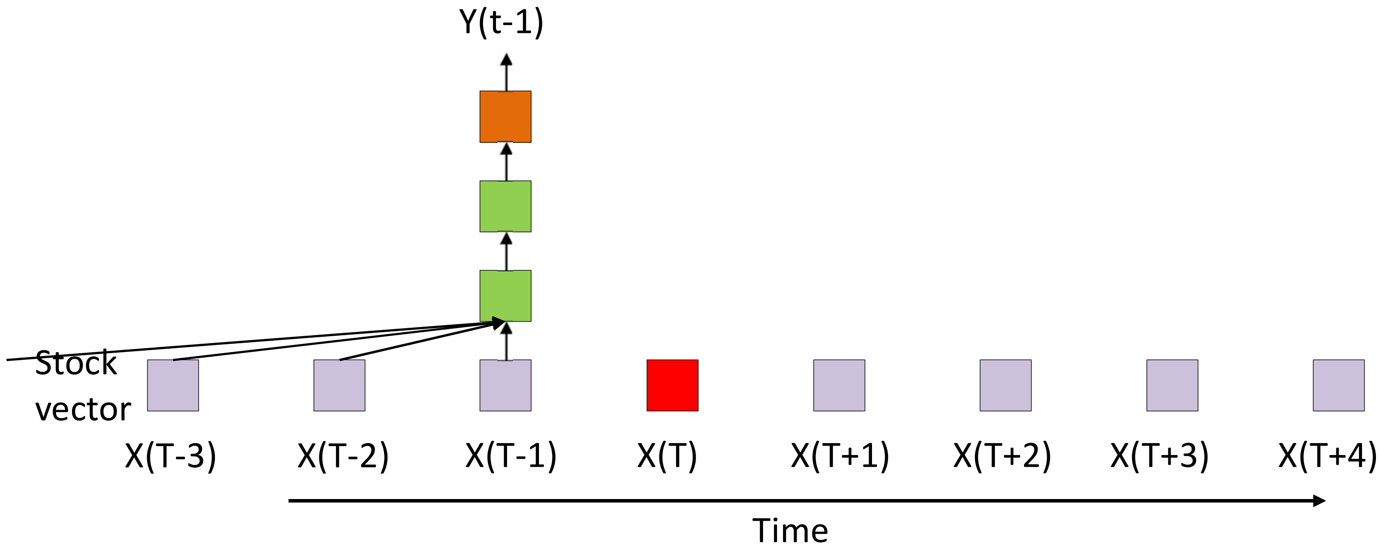
Slide credit: Bhiksha Raj

Finite-response model

- This is a *finite response* system
  - Something that happens *today* only affects the output of the system for N days into the future
    - N is the *width* of the system

$$Y_t = f(X_t, X_{t-1}, \dots, X_{t-N})$$

The stock predictor

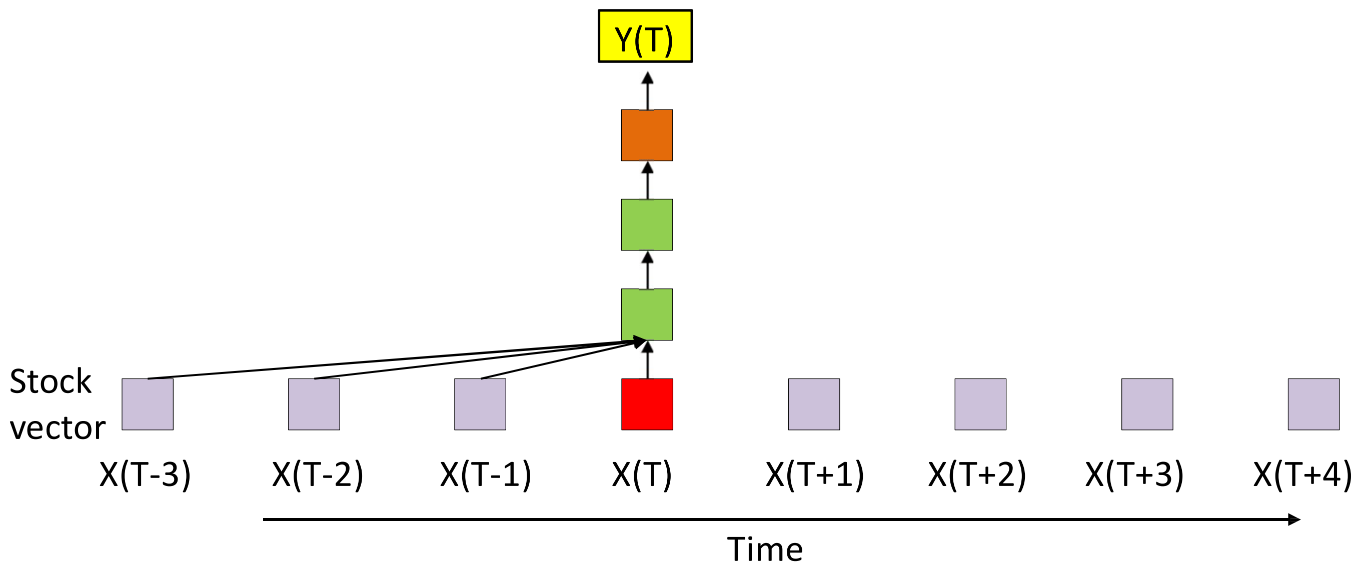


- This is a *finite response* system
  - Something that happens *today* only affects the output of the system for N days into the future
    - N is the *width* of the system

$$Y_t = f(X_t, X_{t-1}, \dots, X_{t-N})$$

Slide credit: Bhiksha Raj

The stock predictor



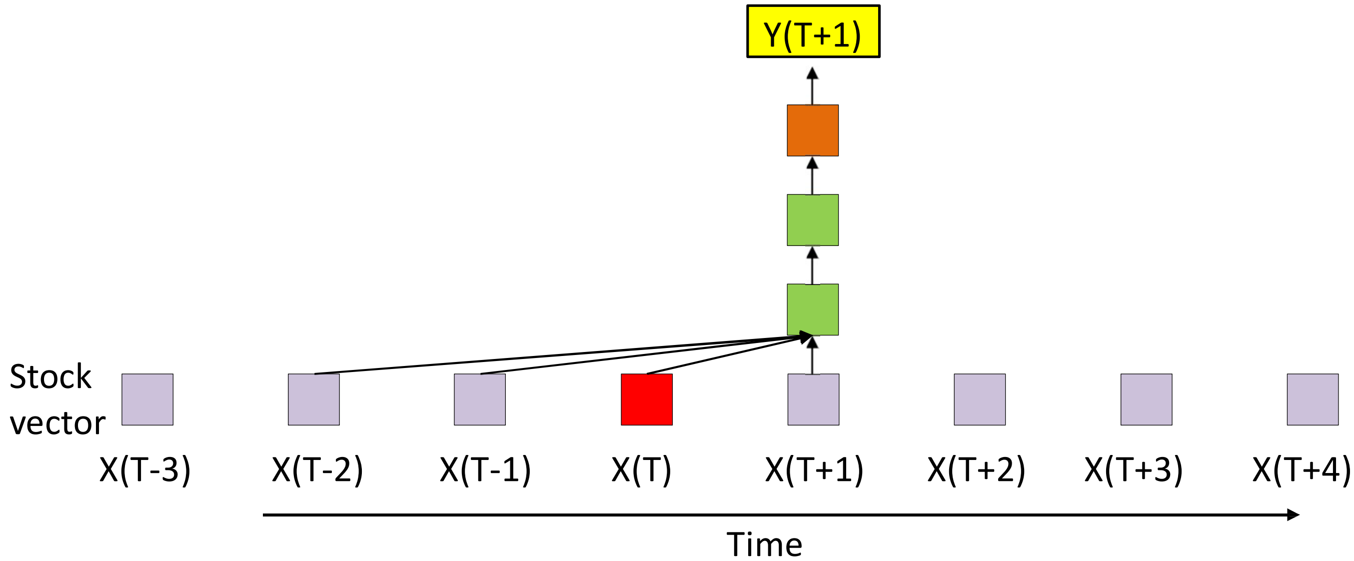
- This is a *finite response* system
  - Something that happens *today* only affects the output of the system for N days into the future
    - N is the *width* of the system

$$Y_t = f(X_t, X_{t-1}, \dots, X_{t-N})$$

BOSTON UNIVERSITY

Slide credit: Bhiksha Raj

The stock predictor

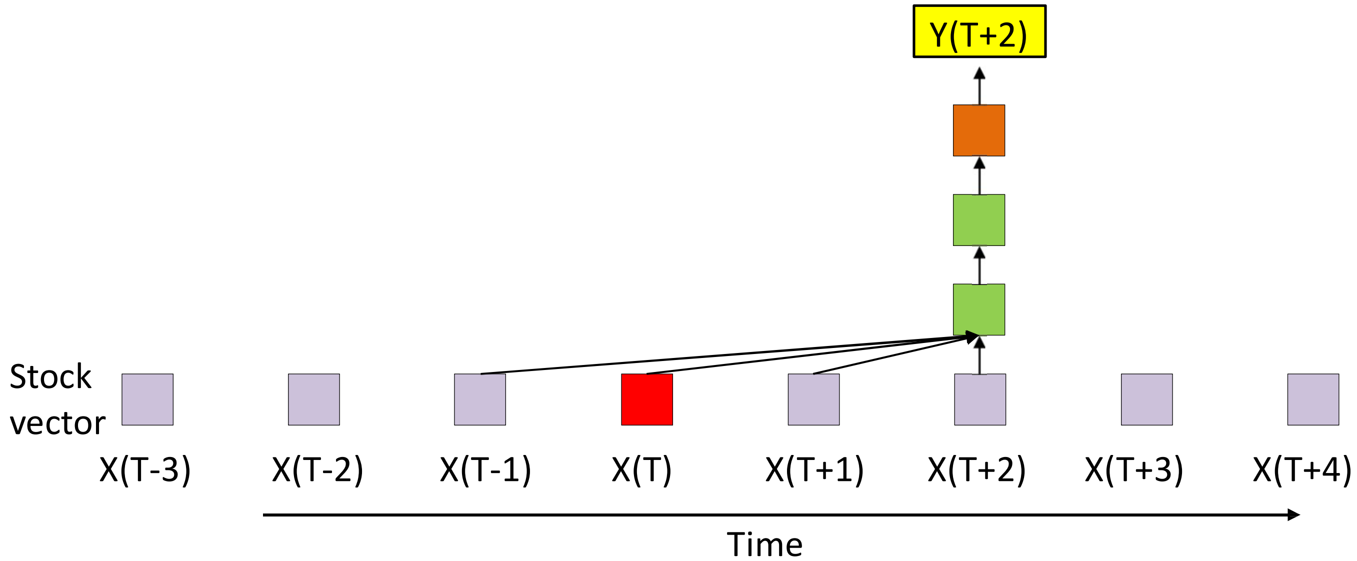


- This is a *finite response* system
  - Something that happens *today* only affects the output of the system for N days into the future
    - N is the *width* of the system

$$Y_t = f(X_t, X_{t-1}, \dots, X_{t-N})$$

Slide credit: Bhiksha Raj

The stock predictor

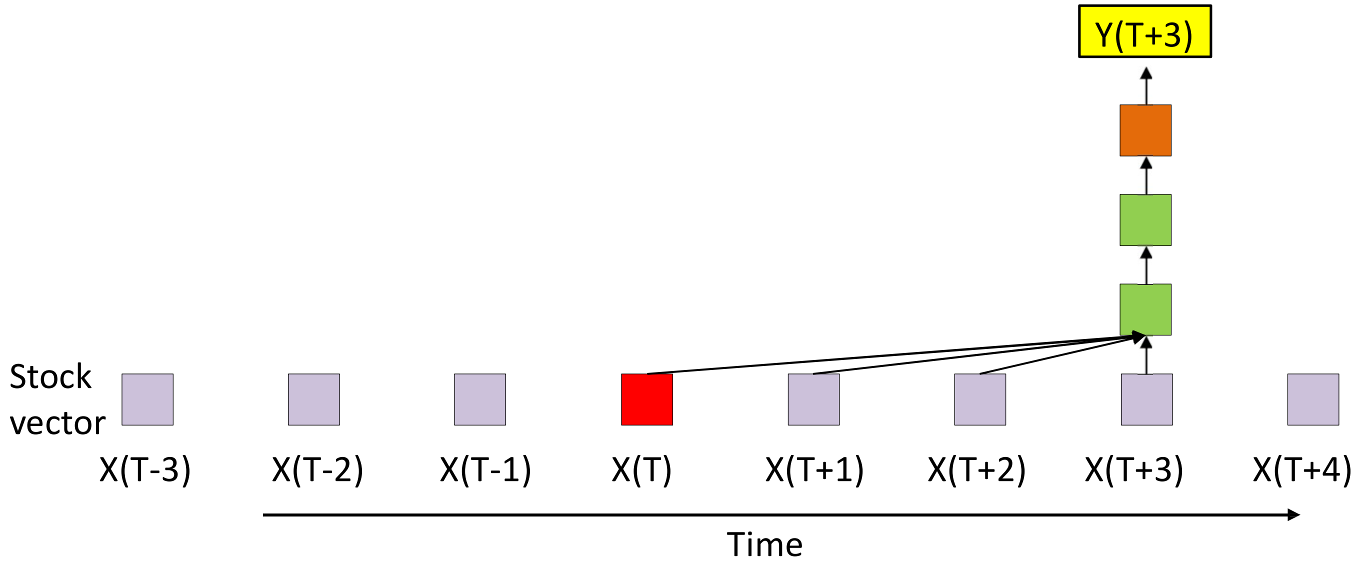


- This is a *finite response* system
  - Something that happens *today* only affects the output of the system for N days into the future
    - N is the *width* of the system

$$Y_t = f(X_t, X_{t-1}, \ldots, X_{t-N})$$

Slide credit: Bhiksha Raj

The stock predictor

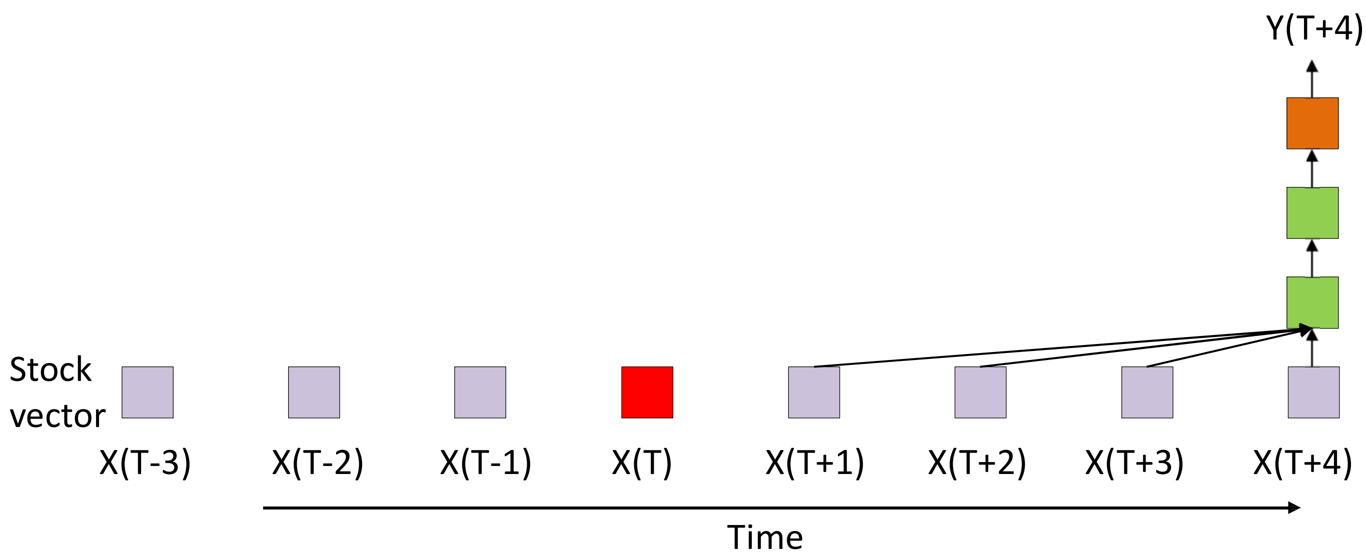


- This is a *finite response* system
  - Something that happens *today* only affects the output of the system for N days into the future
    - N is the *width* of the system

$$Y_t = f(X_t, X_{t-1}, \dots, X_{t-N})$$

Slide credit: Bhiksha Raj

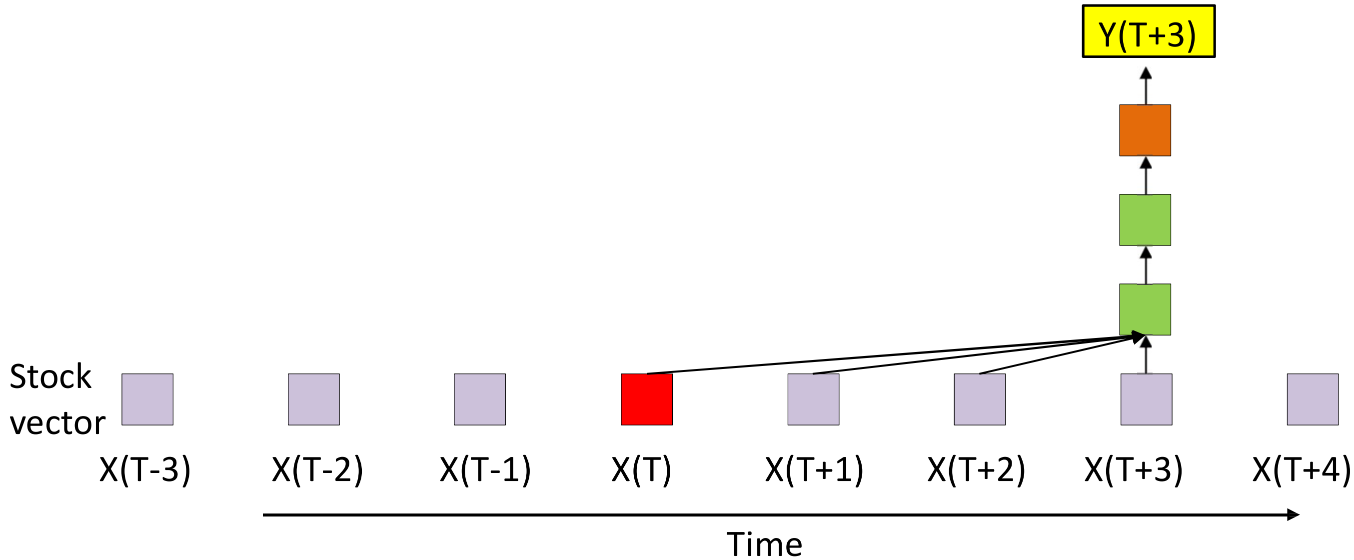
The stock predictor



- This is a *finite response* system
  - Something that happens *today* only affects the output of the system for N days into the future
    - N is the *width* of the system

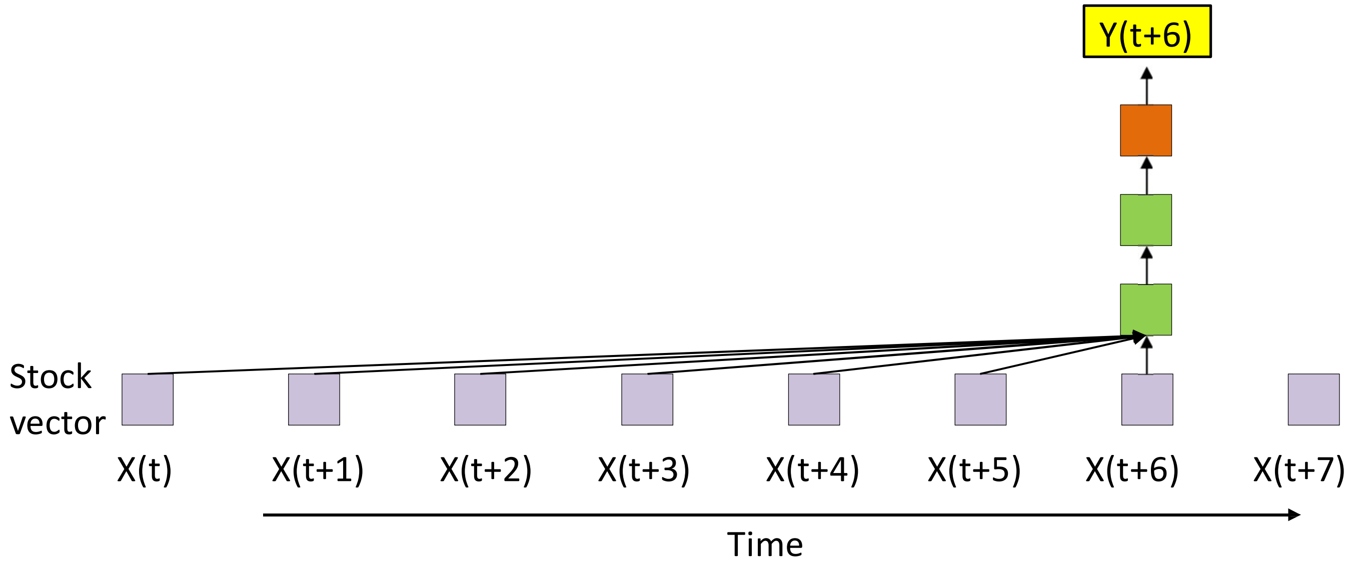
$$Y_t = f(X_t, X_{t-1}, \ldots, X_{t-N})$$

Finite-response model



- Something that happens *today* only affects the output of the system for N days into the future
  - **Predictions consider *N* days of history**
- To consider more of the past to make predictions, you must increase the "history" considered by the system

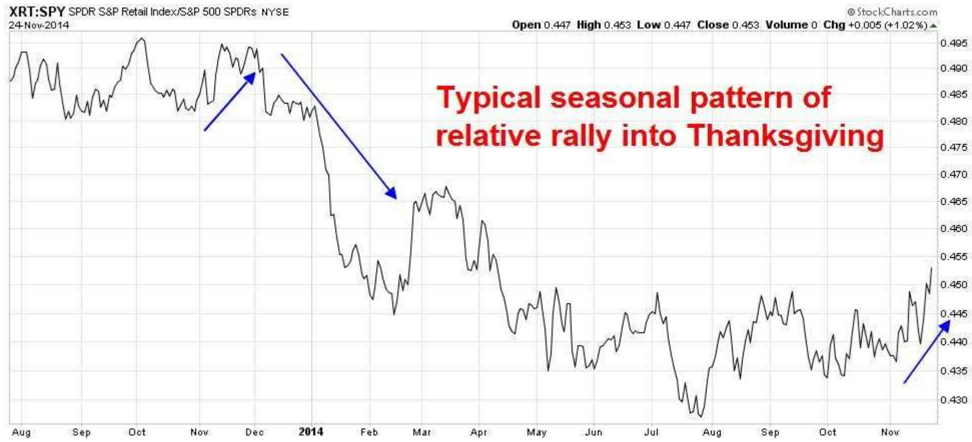
Finite-response



- Problem: Increasing the "history" makes the network more complex
  - No worries, we have the CPU and memory
    - Or do we?

Slide credit: Bhiksha Raj

Systems often have long-term dependencies



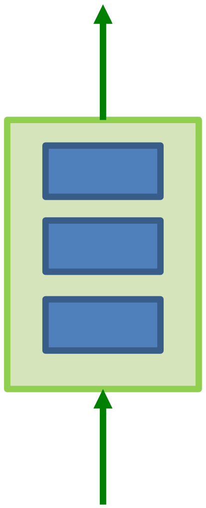
- Longer-term trends –
  - Weekly trends in the market
  - Monthly trends in the market
  - Annual trends
  - Though longer historic tends to affect us less than more recent events..

Slide credit: Bhiksha Raj

An alternate model for infinite response systems:

the state-space model

$$h_t = f(x_t, h_{t-1})$$
$$y_t = g(h_t)$$



- h_t is the *state* of the network
- Need to define initial state h_{-1}

- The state an be arbitrarily complex

Slide credit: Bhiksha Raj

An alternate model for infinite response systems:

the state-space model

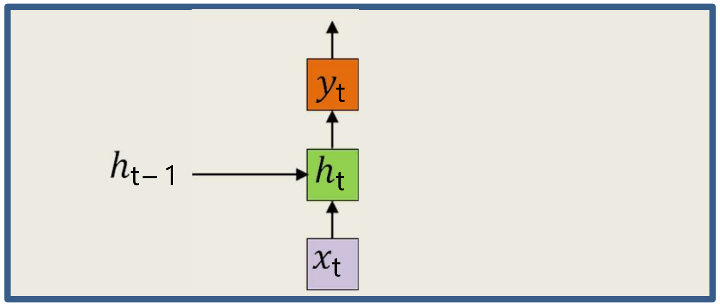
$$h_t = f(x_t, h_{t-1})$$
$$y_t = g(h_t)$$

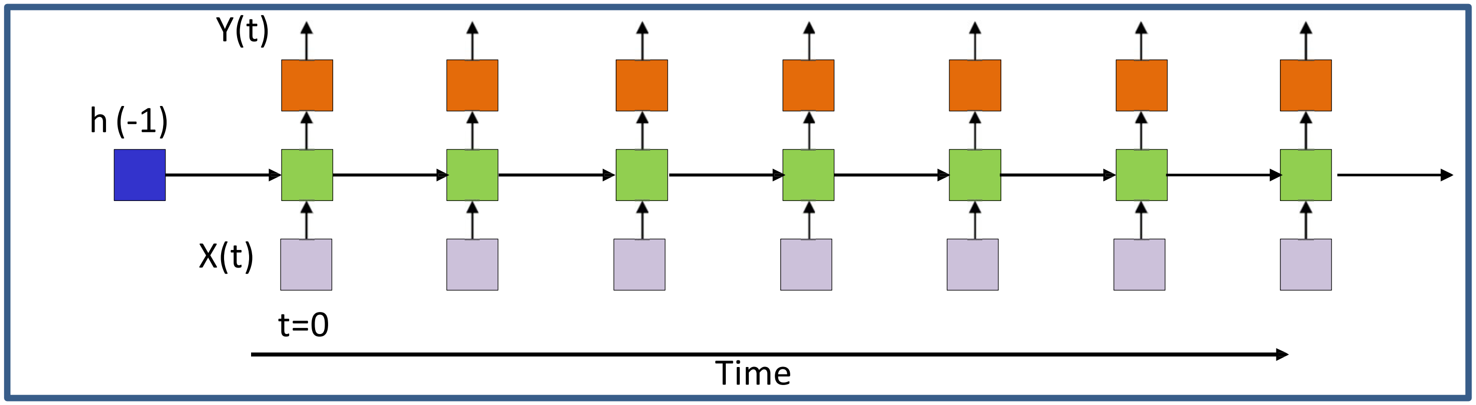
- h_t is the *state* of the network
  - *State* summarizes information about the entire past
    - Model directly embeds the memory in the state
- Need to define initial state h_{-1}

- This is a *fully recurrent* neural network
  - Or simply a *recurrent neural network*

Slide credit: Bhiksha Raj

The simple state-space model

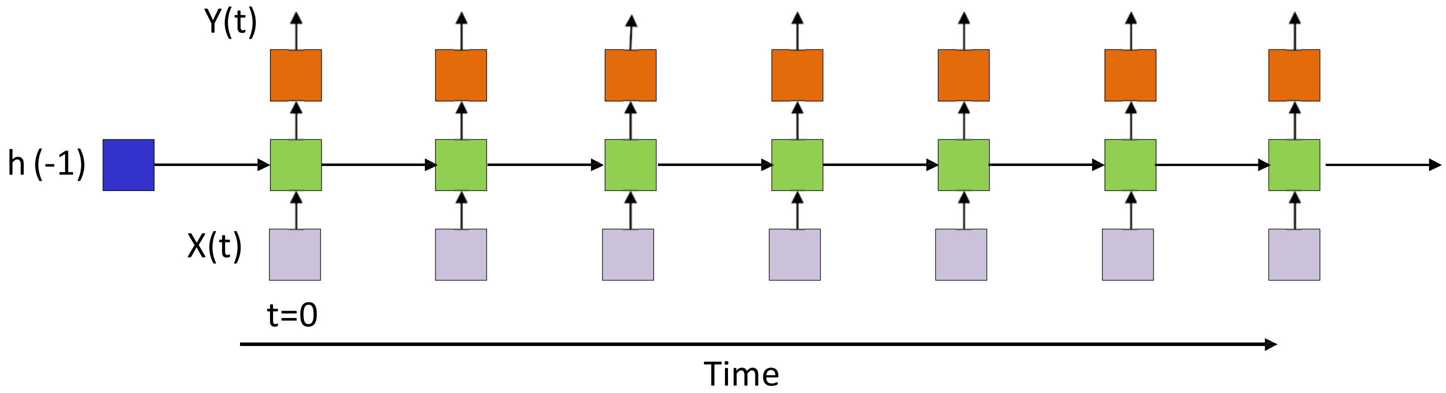




- The state (green) at any time is determined by the input at that time, and the state at the previous time
- *An input at t=0 affects outputs forever*
- Also known as a recurrent neural net

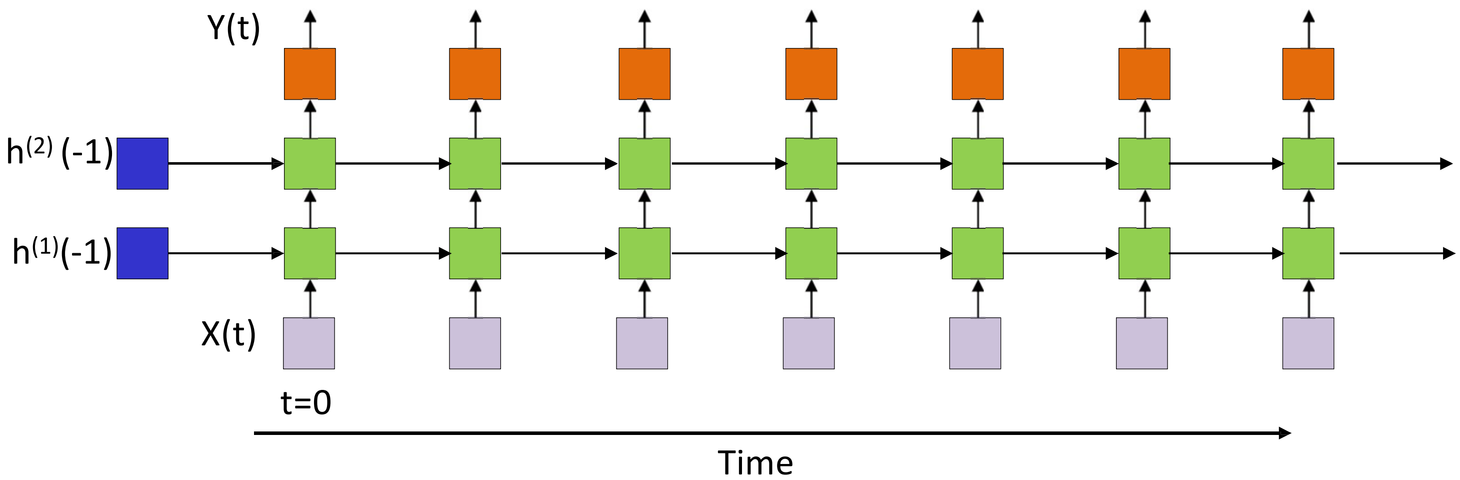
Slide credit: Bhiksha Raj

Single hidden layer RNN



- Recurrent neural network
- All columns are identical
- *An input at t=0 affects outputs forever*

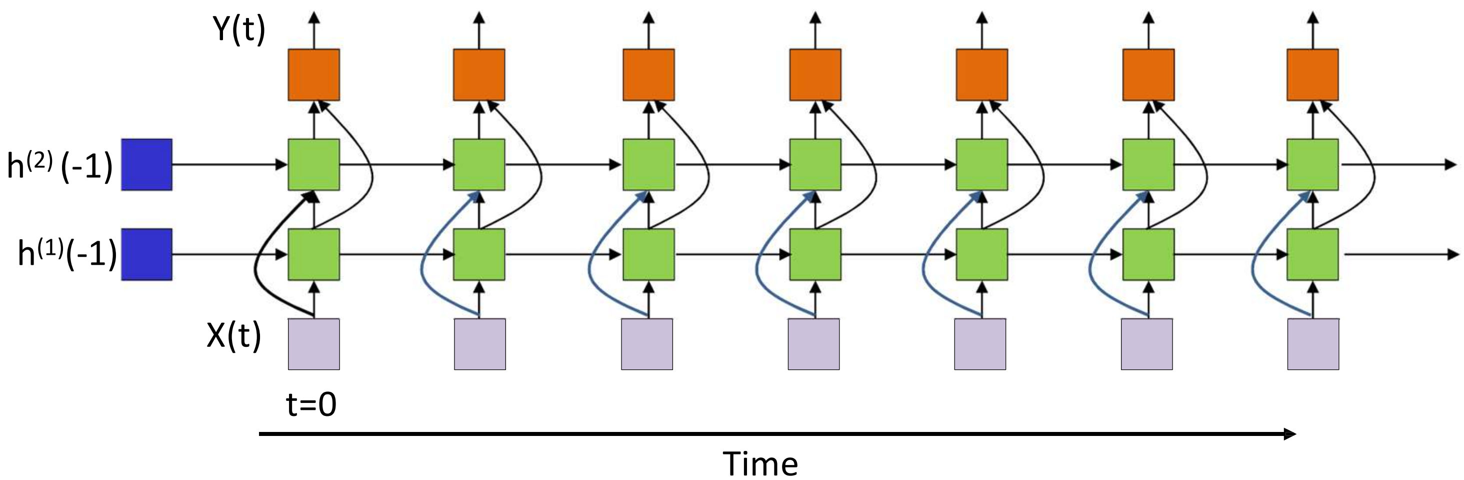
Multiple recurrent layer RNN



- Recurrent neural network
- All columns are identical
- *An input at t=0 affects outputs forever*

Slide credit: Bhiksha Raj

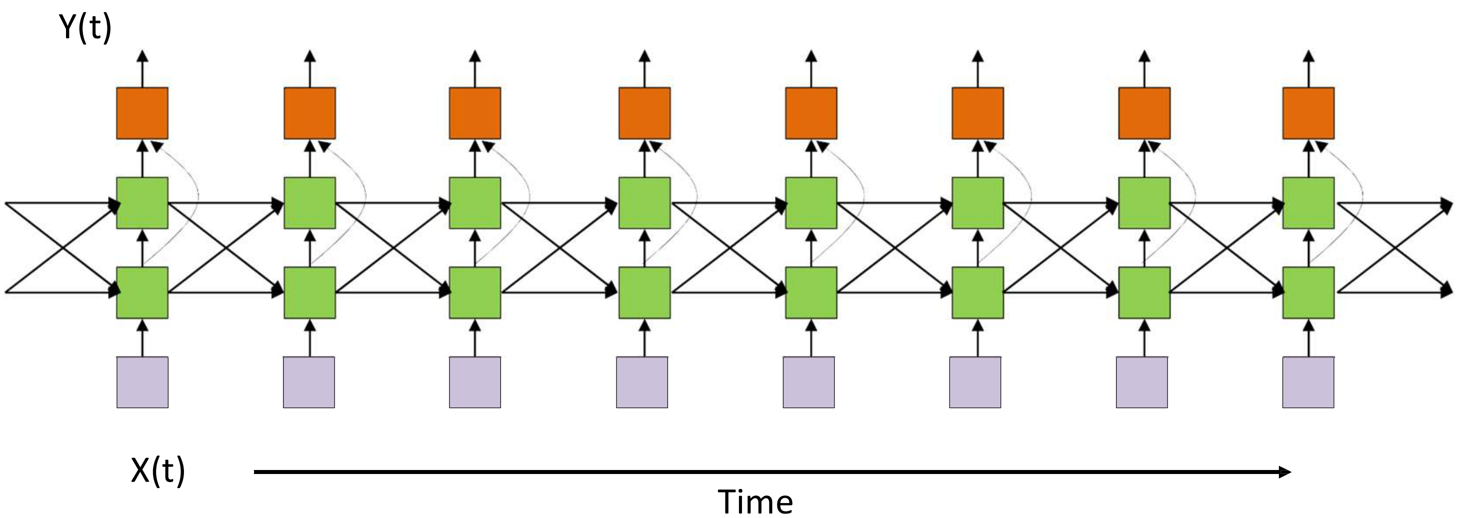
Multiple recurrent layer RNN



- We can also have skips..

Slide credit: Bhiksha Raj

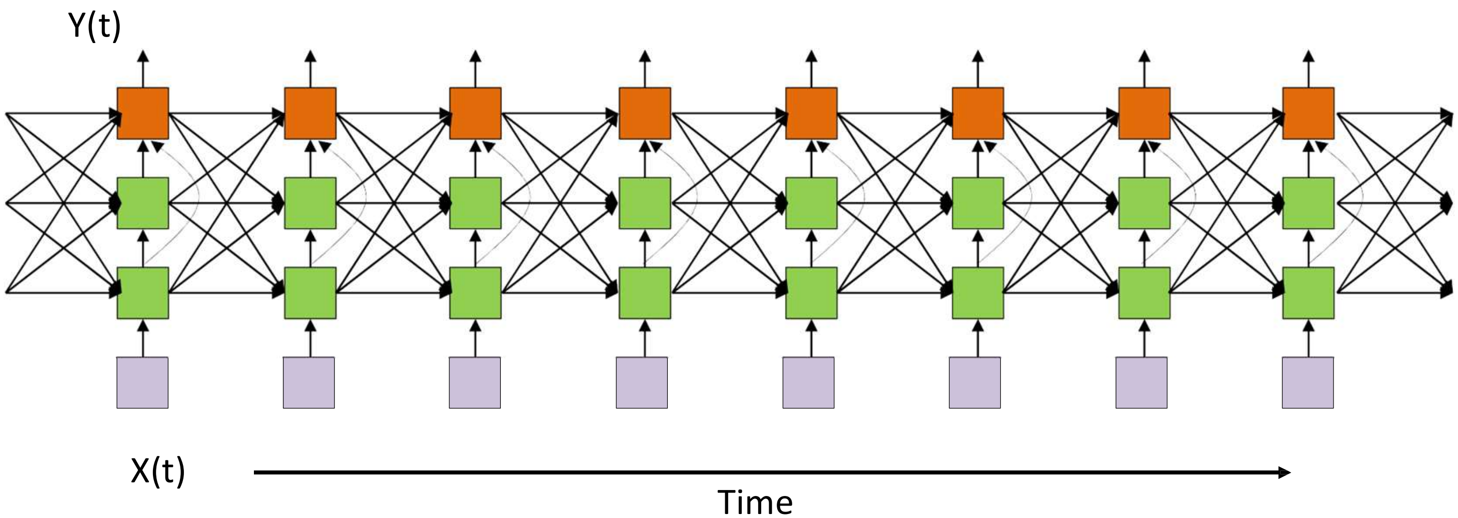
A more complex state



- All columns are identical
- *An input at t=0 affects outputs forever*

Slide credit: Bhiksha Raj

Or the network may be even more complicated

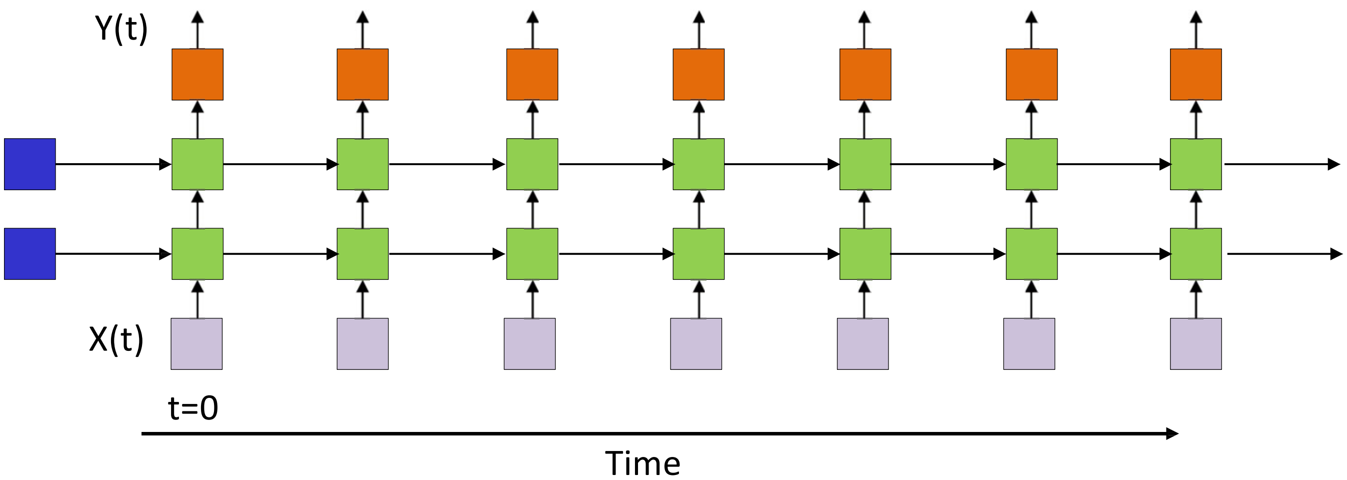


- Shades of NARX
- All columns are identical
- *An input at t=0 affects outputs forever*

BOSTON UNIVERSITY

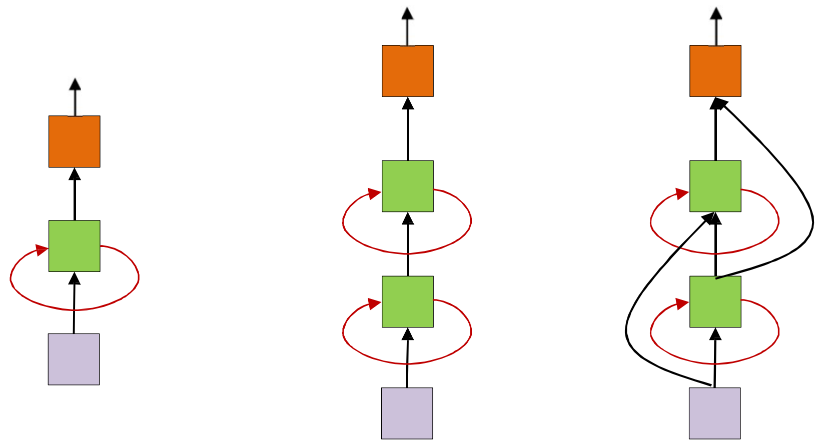
Slide credit: Bhiksha Raj

The simplest structures are most popular



- Recurrent neural network
- All columns are identical
- *An input at t=0 affects outputs forever*

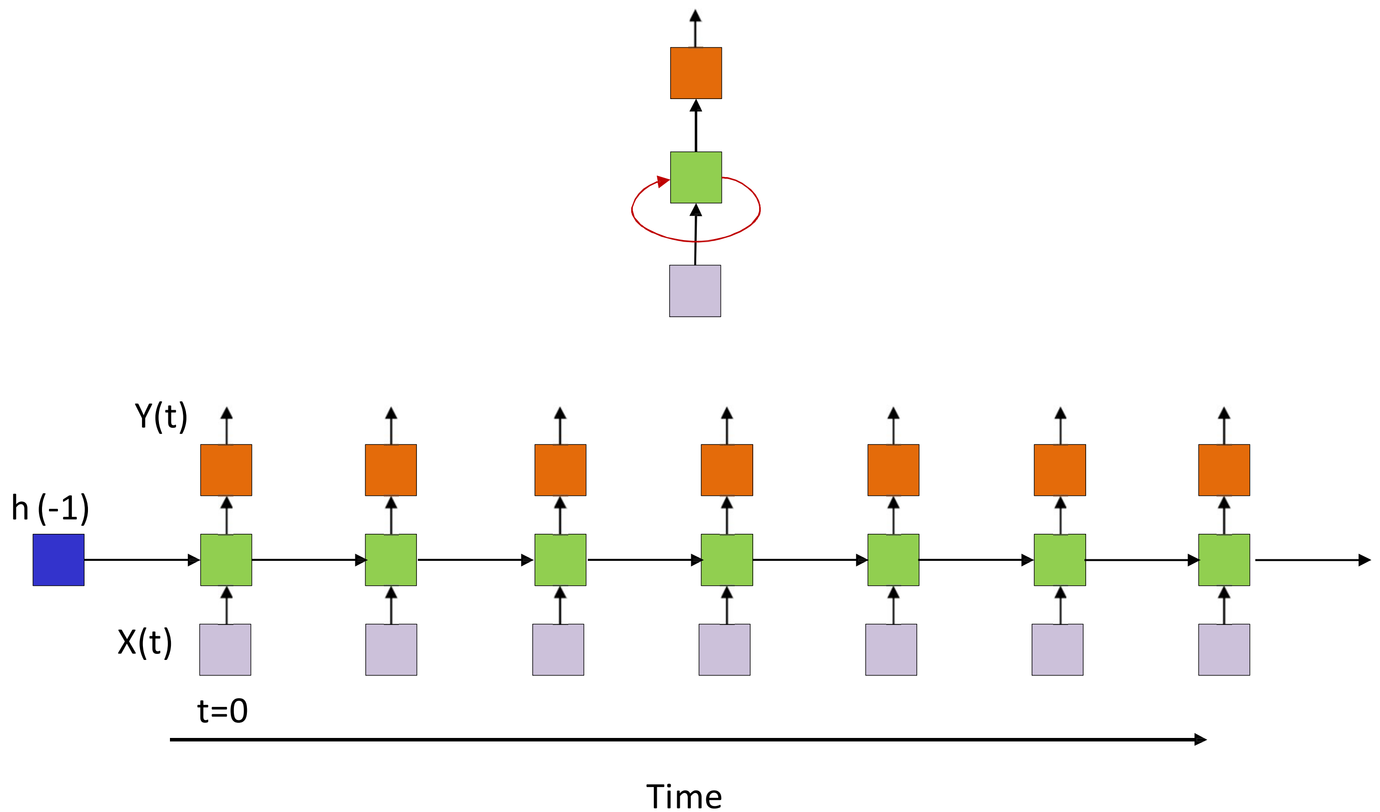
A Recurrent Neural Network



- Simplified models often drawn
- The loops imply recurrence

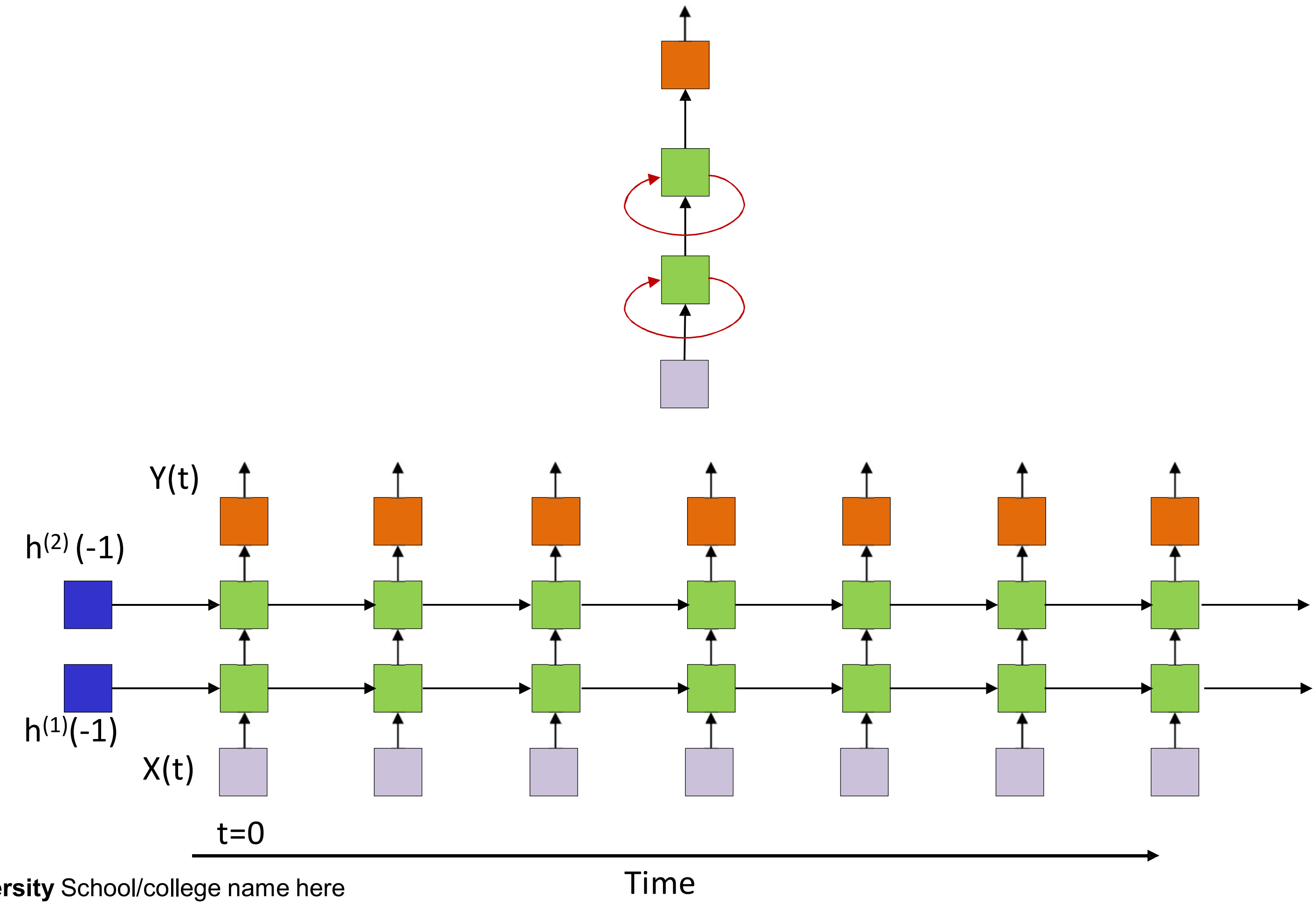
Slide credit: Bhiksha Raj

The detailed version of the simplified representation



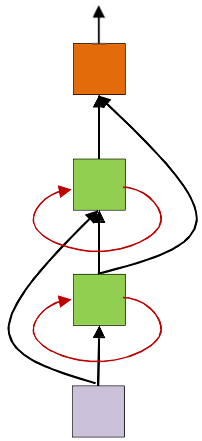
Slide credit: Bhiksha Raj

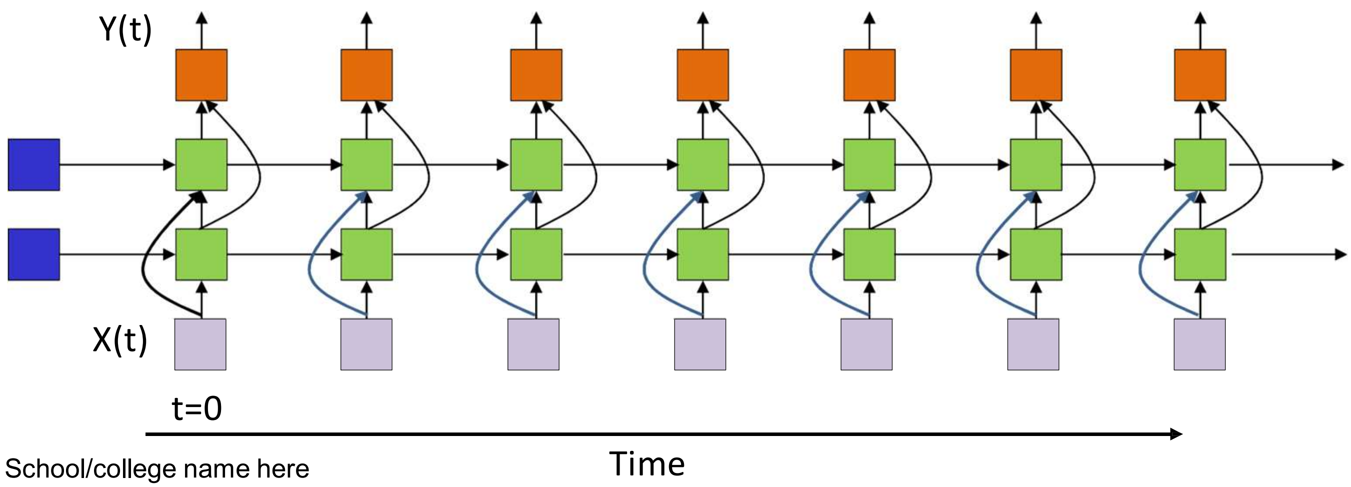
Multiple recurrent layer RNN



Slide credit: Bhiksha Raj

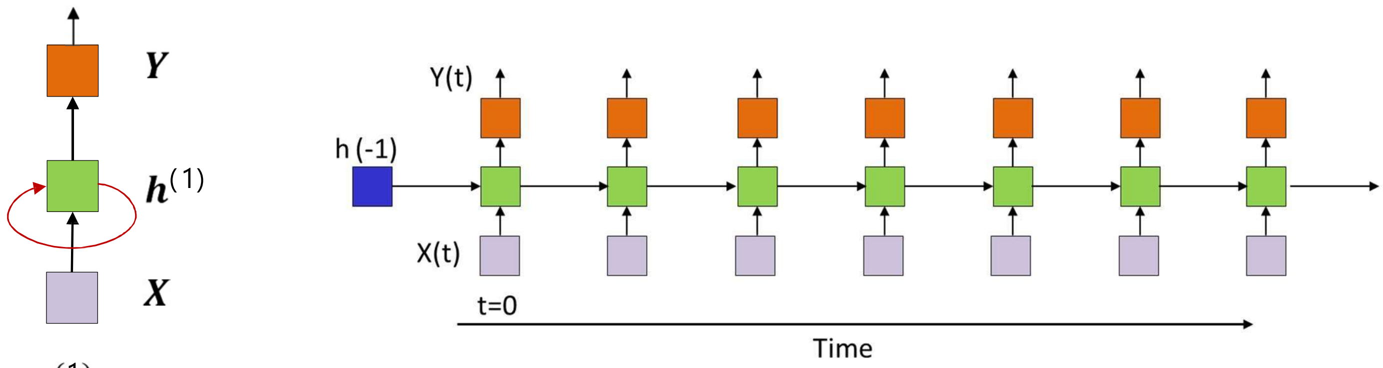
Multiple recurrent layer RNN





Slide credit: Bhiksha Raj

Equations



$h^{(1)}(-1) = part\ of\ network\ parameters$

- Computation:

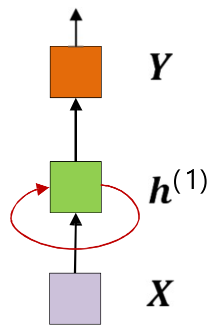
$$h^{(1)}(t) = f_1\left(W^{(1)}X(t) + W^{(11)}h^{(1)}(t-1) + b^{(1)}\right)$$

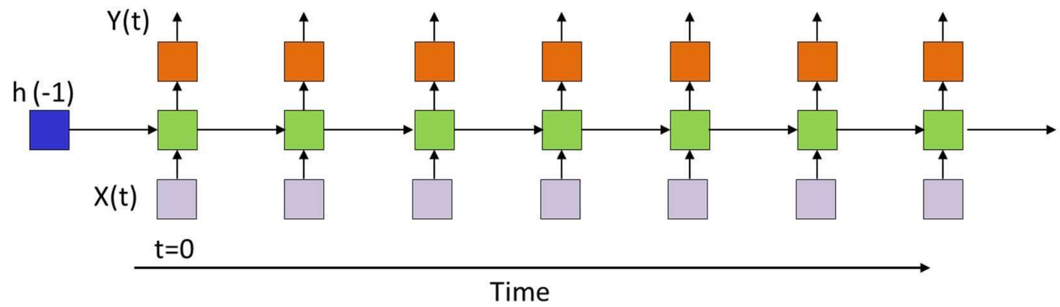
$$Y(t) = f_2\left(W^{(2)}h^{(1)}(t) + b^{(2)}\right)$$

- The recurrent state activation $f_1()$ is typically $\tanh()$

Slide credit: Bhiksha Raj

Equations





Current weights = *part of network parameters*

Recurrent weights

- Computation:

$$\boldsymbol{h}^{(1)}(t) = f_1\left(\boldsymbol{W}^{(1)}\boldsymbol{X}(t) + \boldsymbol{W}^{(11)}\boldsymbol{h}^{(1)}(t-1) + \boldsymbol{b}^{(1)}\right)$$

$$\boldsymbol{Y}(t) = f_2\left(\boldsymbol{W}^{(2)}\boldsymbol{h}^{(1)}(t) + \boldsymbol{b}^{(2)}\right)$$

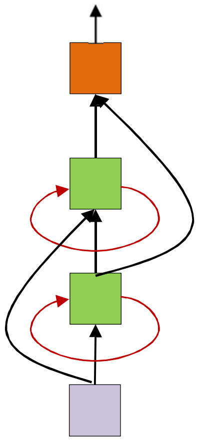
- The recurrent state activation $f_1()$ is typically $\tanh()$

Slide credit: Bhiksha Raj

Equations

$$\boldsymbol{h}^{(1)}(-1) = part\ of\ network\ parameters$$

$$\boldsymbol{h}^{(2)}(-1) = part\ of\ network\ parameters$$



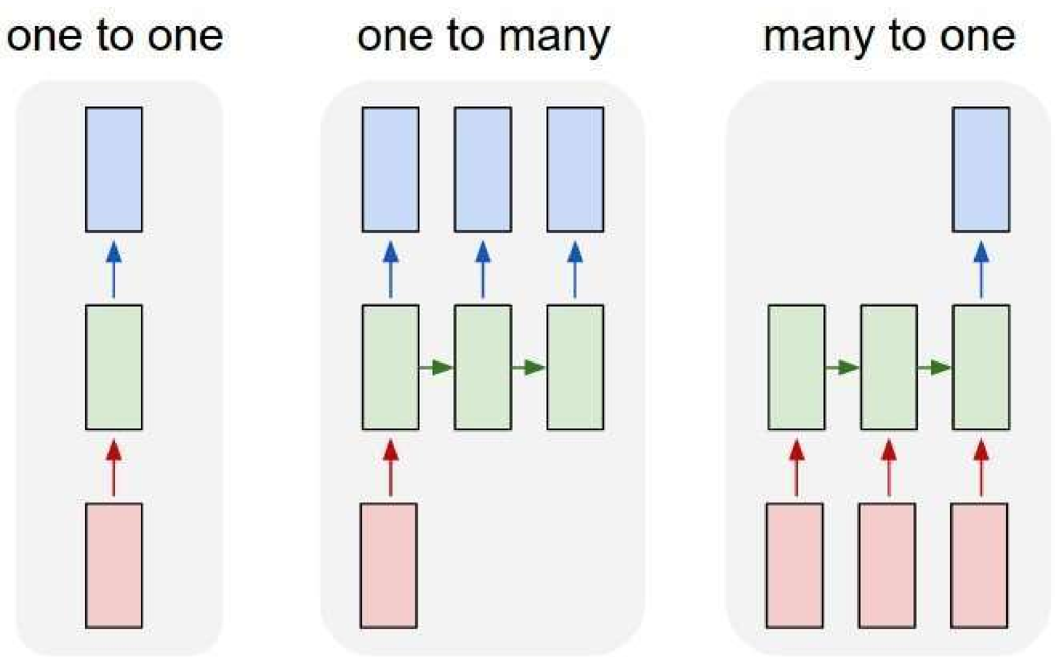
$$\boldsymbol{h}^{(1)}(t) = f_1\left(\boldsymbol{W}^{(01)}\boldsymbol{X}(t) + \boldsymbol{W}^{(11)}\boldsymbol{h}^{(1)}(t-1) + \boldsymbol{b}^{(1)}\right)$$

$$\boldsymbol{h}^{(2)}(t) = f_2\left(\boldsymbol{W}^{(12)}\boldsymbol{h}^{(1)}(t) + \boldsymbol{W}^{(02)}\boldsymbol{X}(t) + \boldsymbol{W}^{(22)}\boldsymbol{h}^{(2)}(t-1) + \boldsymbol{b}^{(2)}\right)$$

$$\boldsymbol{Y}(t) = f_3\left(\boldsymbol{W}^{(23)}\boldsymbol{h}^{(2)}(t) + \boldsymbol{W}^{(13)}\boldsymbol{h}^{(1)}(t) + \boldsymbol{b}^{(3)}\right)$$

Slide credit: Bhiksha Raj

Variants on recurrent nets

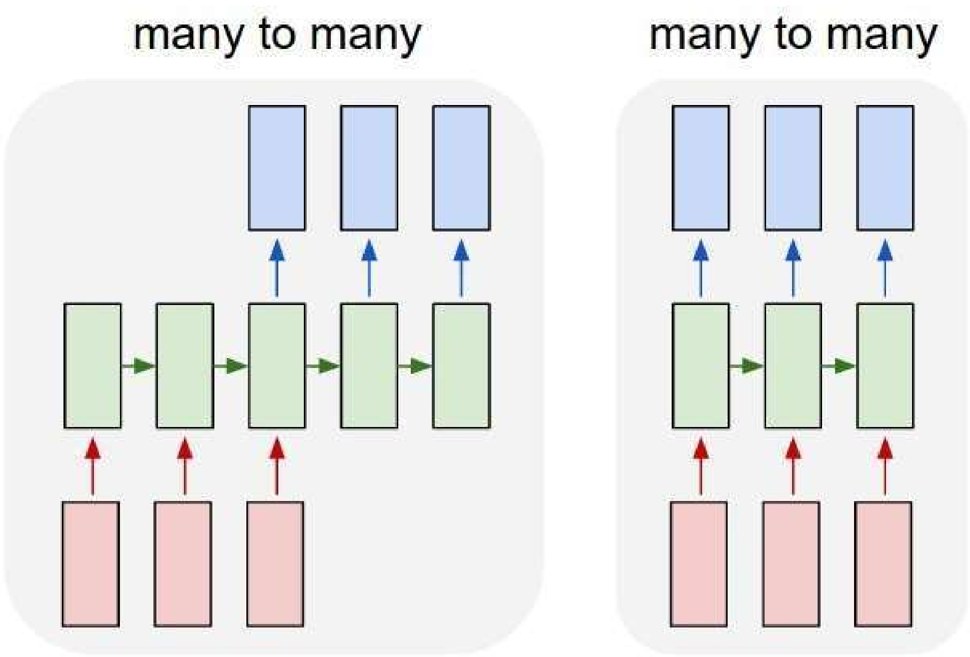


Images from Karpathy

- 1: Conventional MLP
- 2: Sequence *generation*, e.g. image to caption
- 3: Sequence based *prediction or classification,* e.g. Speech recognition, text classification

Slide credit: Bhiksha Raj

Variants



Images from Karpathy

- 1: *Delayed* sequence to sequence, e.g. machine translation
- 2: Sequence to sequence, e.g. stock problem, label prediction
- Etc…

Slide credit: Bhiksha Raj

The End

Thanks for your attention.

I would be glad if you have any question.

BOSTON UNIVERSITY