Introduction to Recurrent Neural Networks

Vanilla RNN / Long-Short Term Memory /
Gated Recurrent Unit

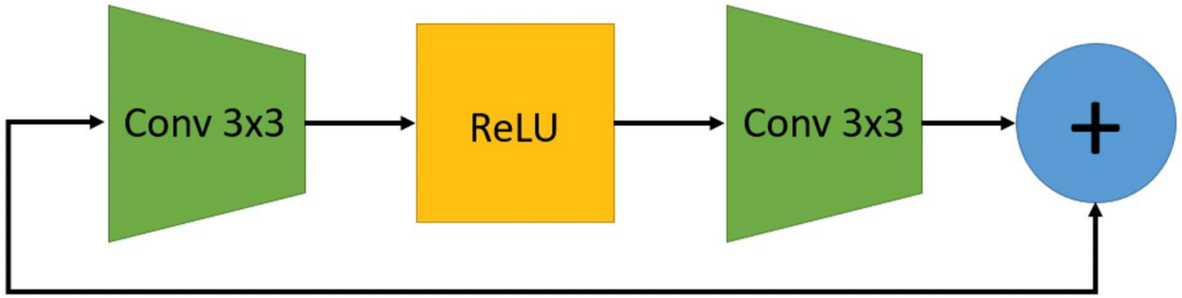
Dr. Risman Adnan Mattororang
Telkom University

Outline

- Why not feed-forward neural networks
- What is recurrent neural networks
  - Issue with recurrent neural networks
  - Vanishing and exploding gradient
- Introduction to long-short term memory
- Gated Recurrent Unit

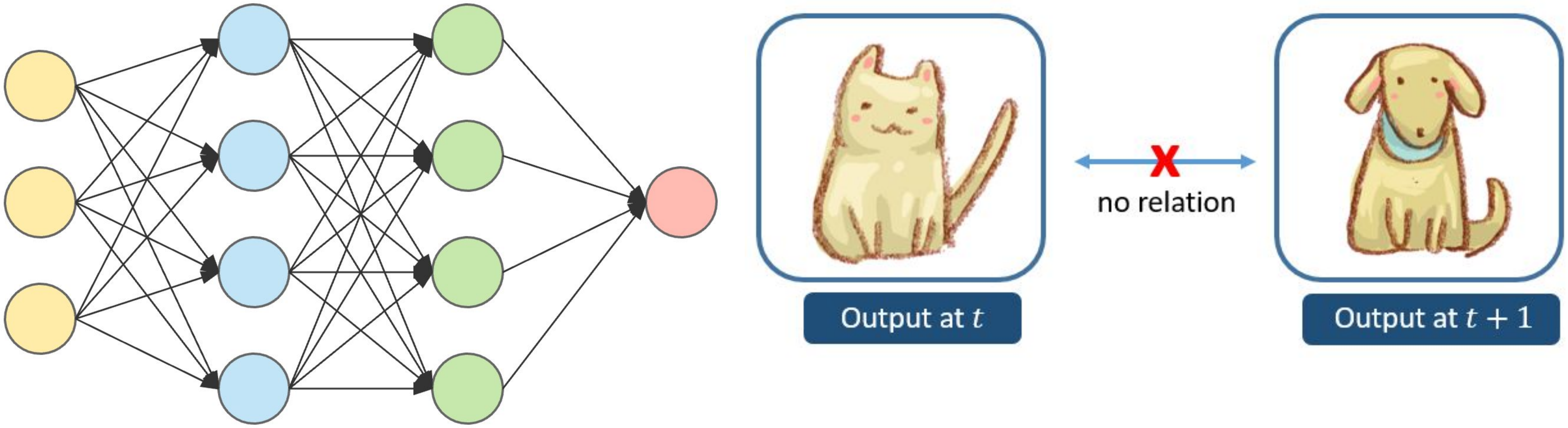
Feed-forward Structure

Standard Neural Networks are DAGs (Directed Acyclic Graphs). That means they have a topological ordering.



“They process one input instance at a time.”

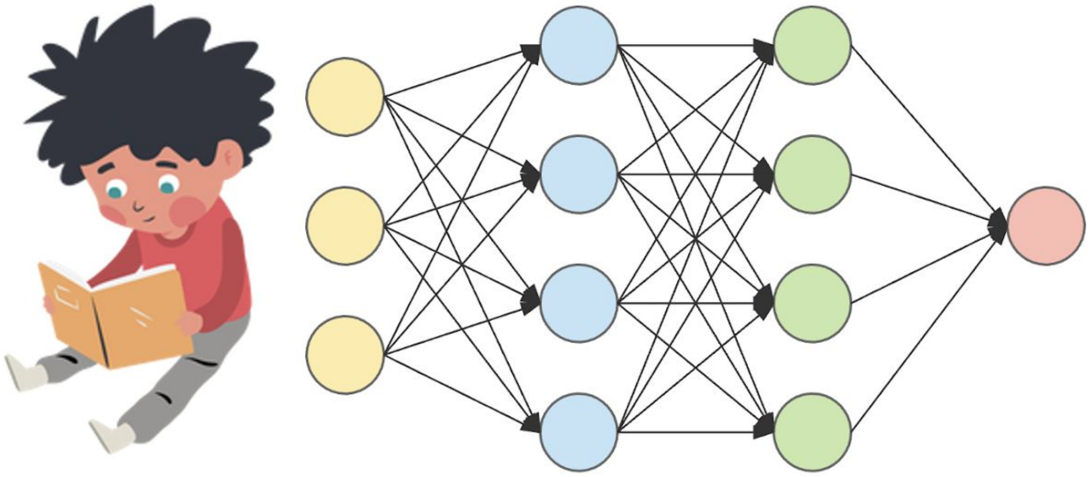
Why not feed-forward neural networks

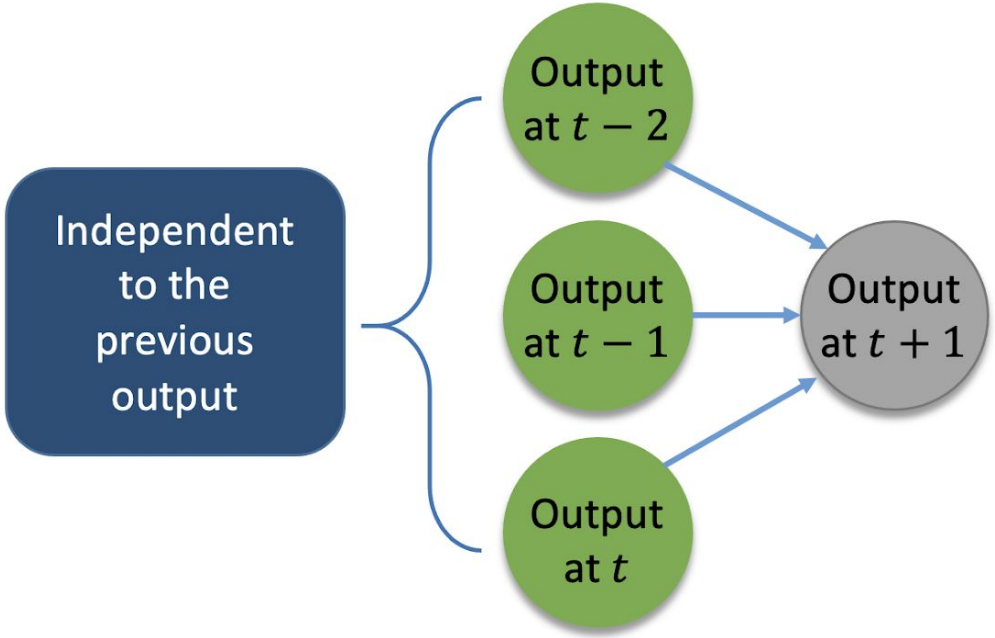


A Trained feed-forward can be access to any random collection of data (images) and the image at t exposed is not necessarily alter how the image at $t+1$ is predicted

Why not feed-forward neural networks

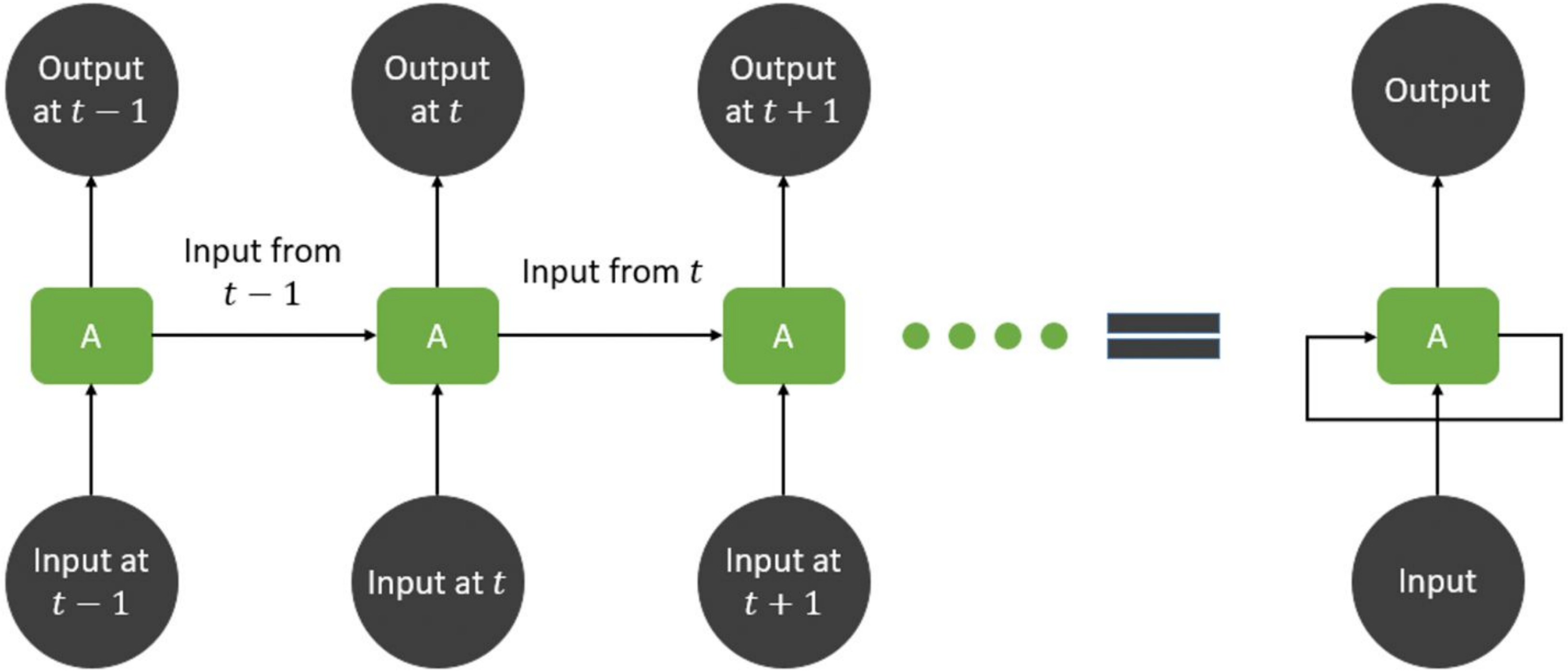
When we read book, we understand the content based on our understanding from previous content / words





Why not feed-forward neural networks

How to overcome the problem ?

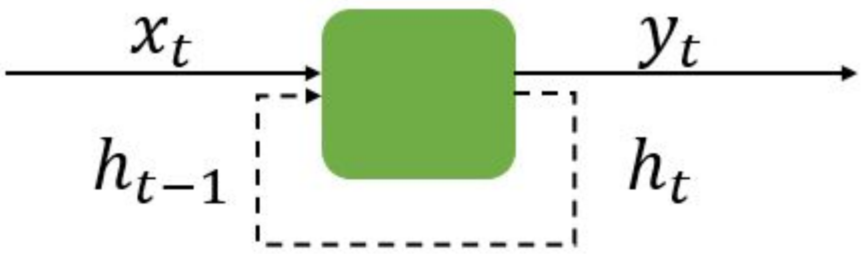


Outline

- Why not feed-forward neural networks
- **What is recurrent neural networks**
  - Issue with recurrent neural networks
  - Vanishing and exploding gradient
- Introduction to long-short term memory
- Gated Recurrent Unit

What is recurrent neural networks ?

- produce sequences of outputs $y_1, y_2, \dots, y_n$
- Examples of sequence data in real-world, genomes, handwriting, the spoken word, numerical time series data (such as sensors), etc.



What is Recurrent Neural Network

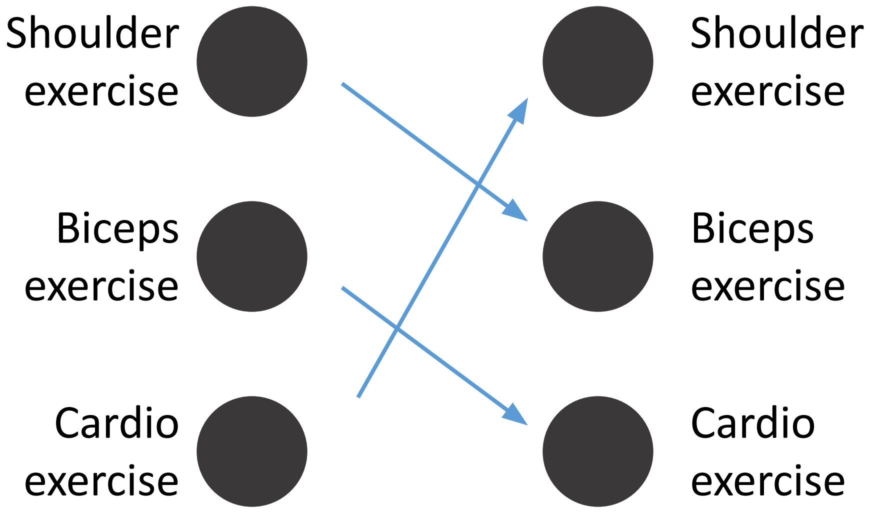
patient

Suppose:
A doctor physiotherapy made a schedule to patient and the exercise schedule are repeated every third day.

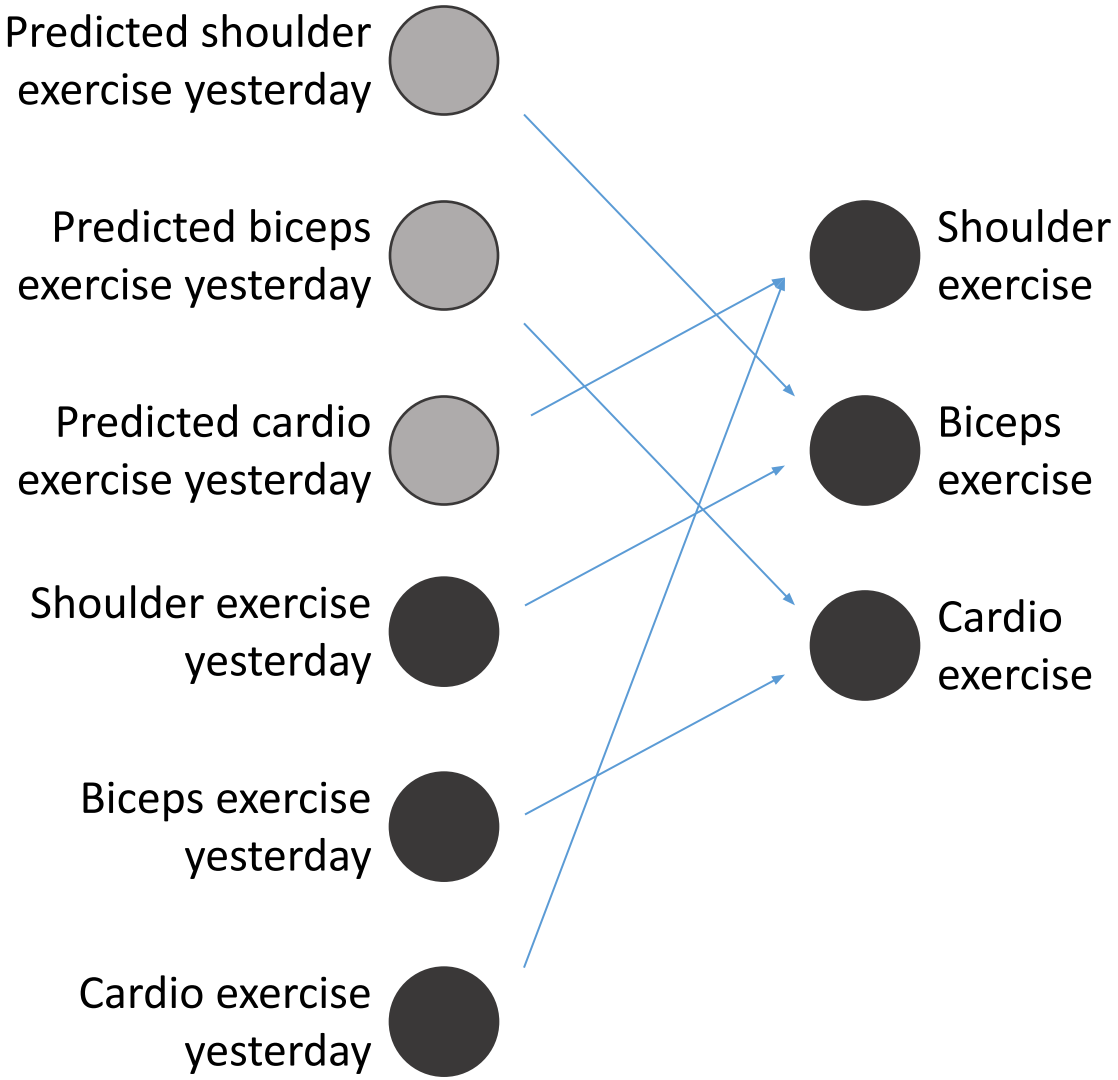
- First day, shoulder exercise
- Second day, biceps exercise
- Third day, cardio exercise

“An analogy and real-world use case”

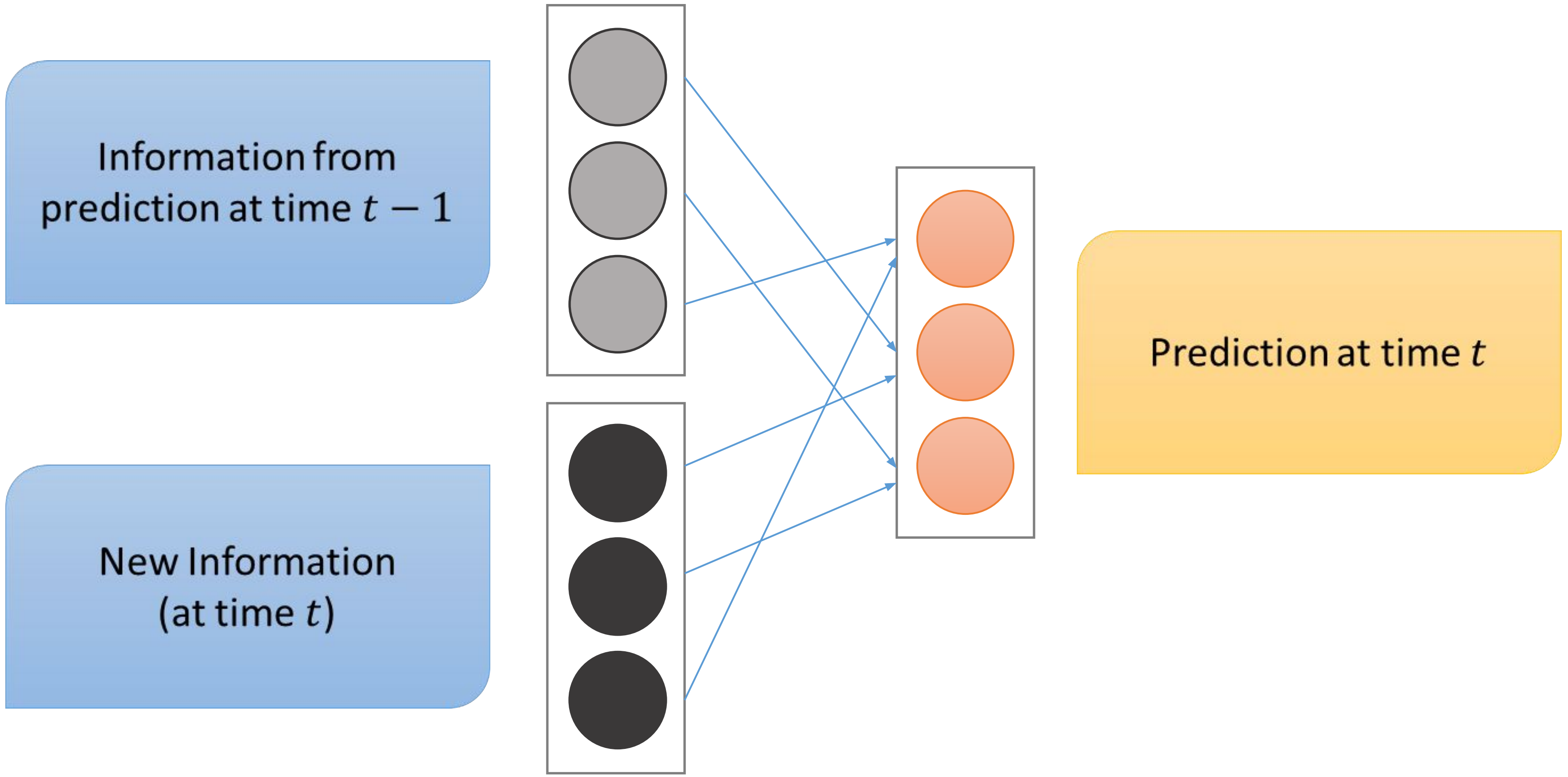
What is recurrent neural networks ?



Predicting type of exercise

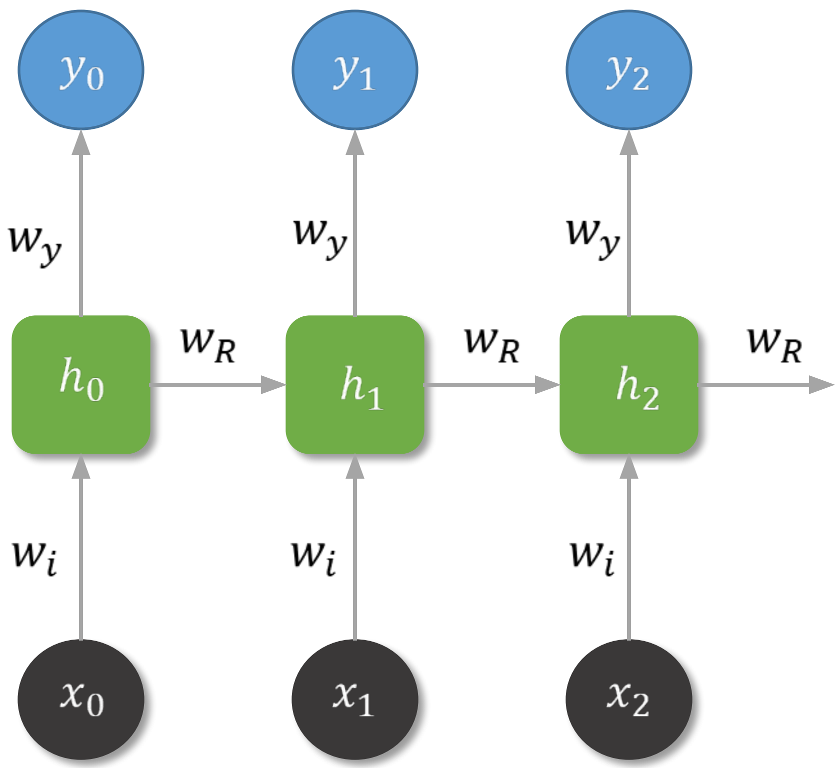


What is recurrent neural networks ?



Predicting type of exercise

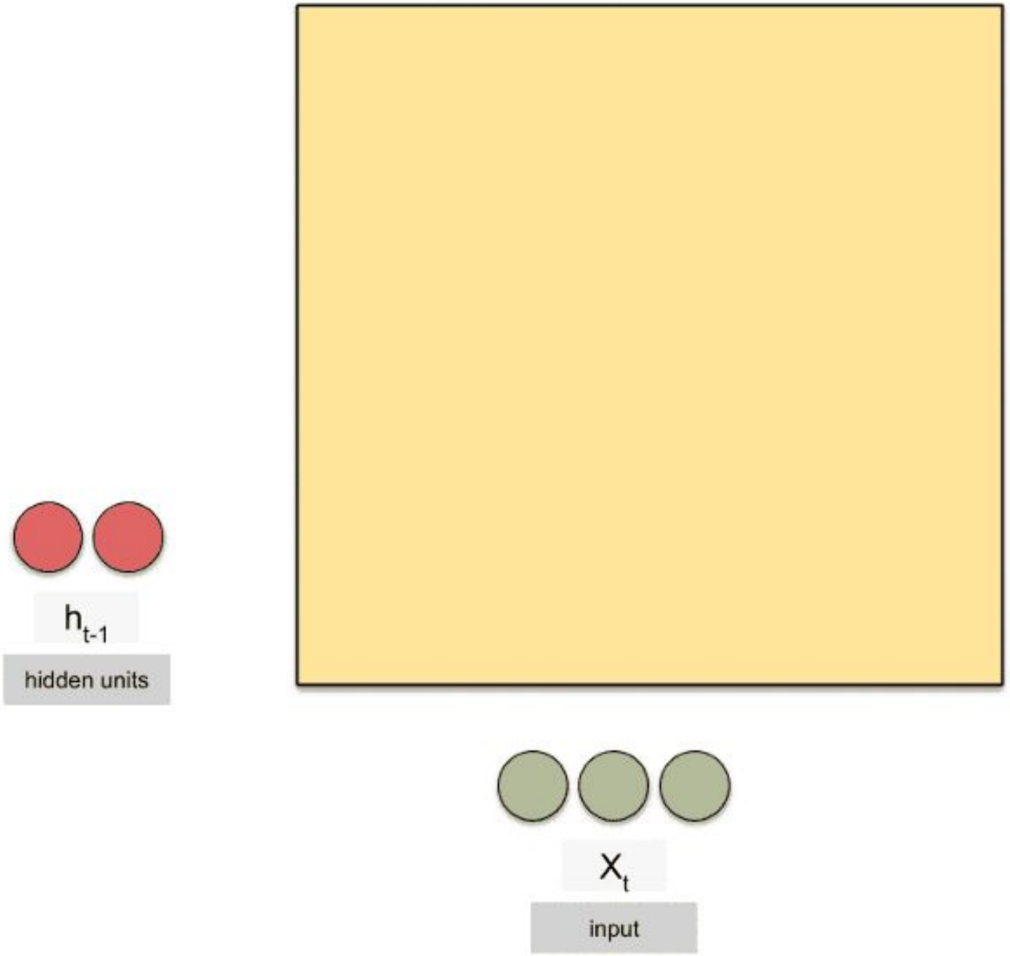
What is recurrent neural networks ?



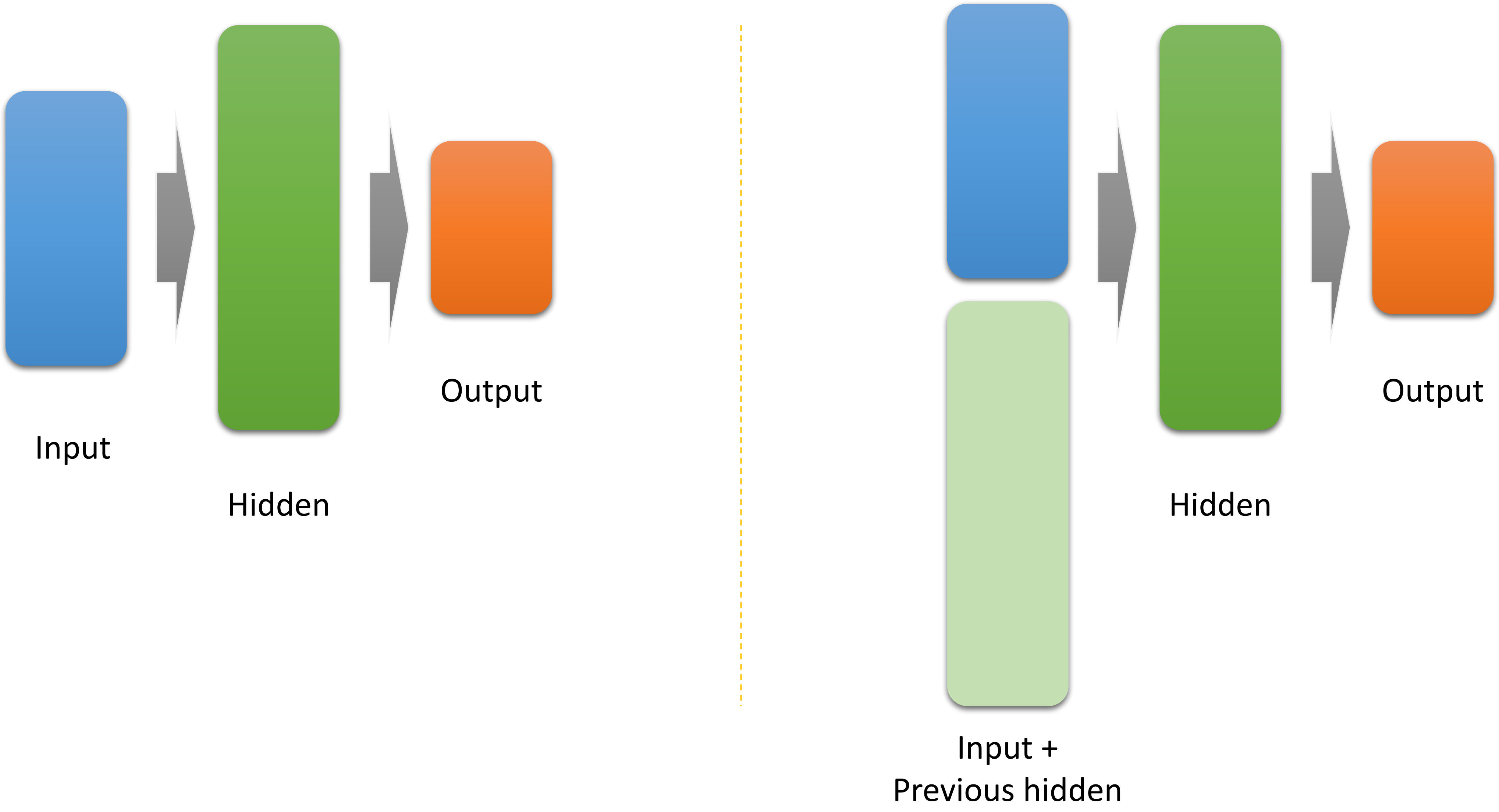
$$h_{(t)} = g_h(w_i x_t + w_R h_{(t-1)} + b_h)$$

$$y_{(t)} = g_y(w_y h_{(t)} + b_y)$$

Animated RNN



(Recall) Feedforward NN Vs Recurrent NN



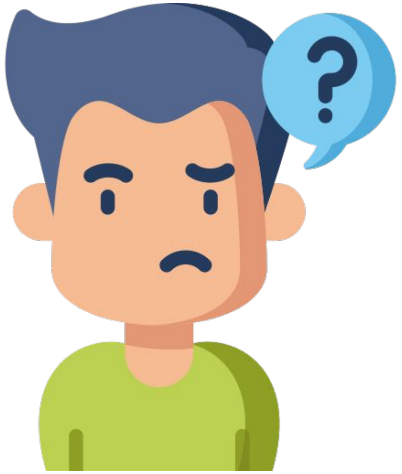
Recurrent Neural Networks

Why ?

$$[input + previous\ hidden] \rightarrow Hidden \rightarrow Output$$

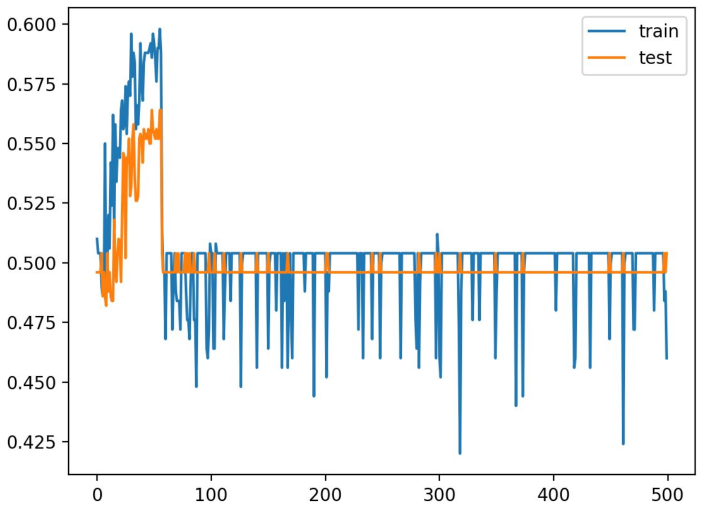
And why not

$$[input + previous\ input] \rightarrow Hidden \rightarrow Output$$



Training a recurrent neural networks

Recurrent neural networks uses backpropagation algorithm, however this algorithm applied for every timestamp. It is known as backpropagation through time (BTT)

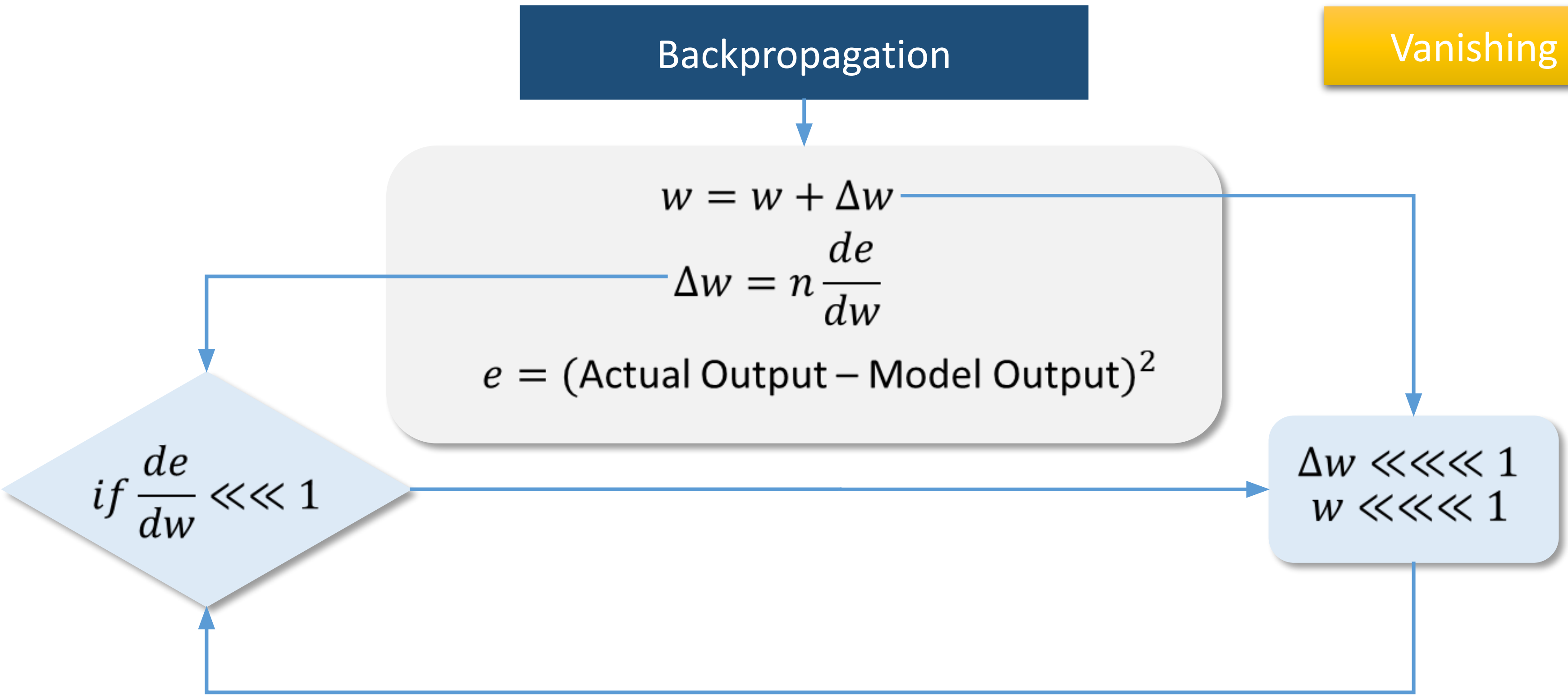


Vanishing gradient problem

Exploding gradient problem

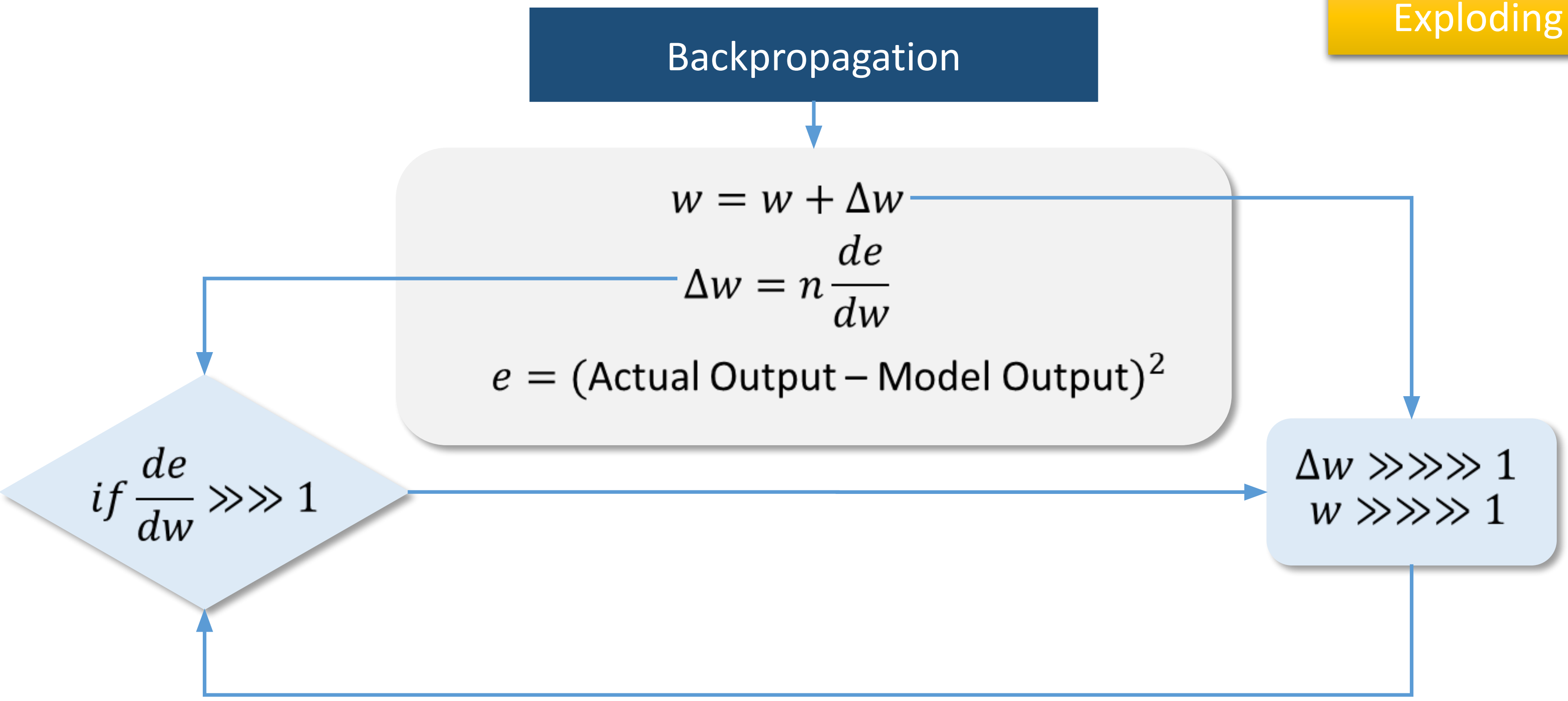
Problem with RNN

Vanishing gradient problem



Problem with RNN

Exploding gradient problem



How to overcome RNN Challenges ?

Vanishing Gradient Problem

- **ReLU activation function**
  we can use activation like ReLU, which gives us output 1 while calculating gradient
- **Clipping**
  Clip the gradient value when it goes lower than threshold
- **LSTM / GRUs**
  Different network architecture that has been designed to overcome this problem

Exploding Gradient Problem

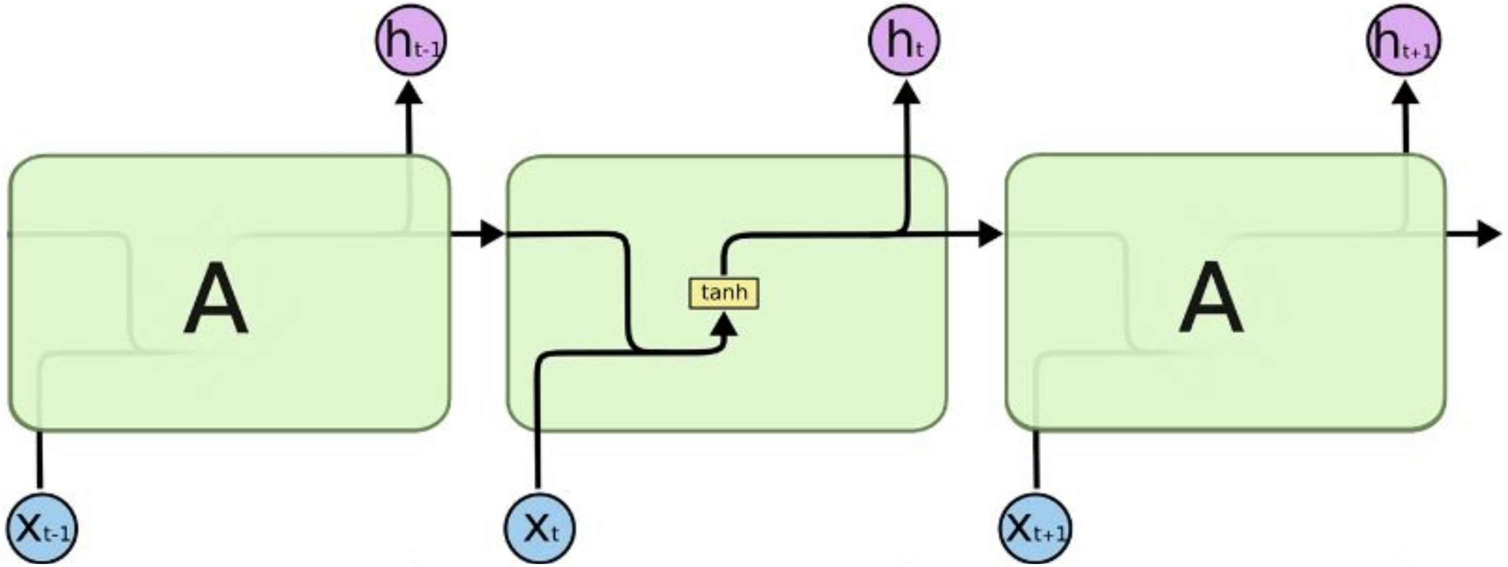
- **Truncated BTT**
  Instead of we start backpropagation at the last timestamp, we choose smaller timestamp such as 10 (we will lose temporal context after 10 timestamp)
- **Clipping**
  Clip the gradient value when it goes bigger than threshold
- **RSMProp**
  Using RMSProp to adjust learning rate

Outline

- Why not feed-forward neural networks
- What is recurrent neural networks
  - Issue with recurrent neural networks
  - Vanishing and exploding gradient
- **Introduction to long-short term memory**
- Gated Recurrent Unit

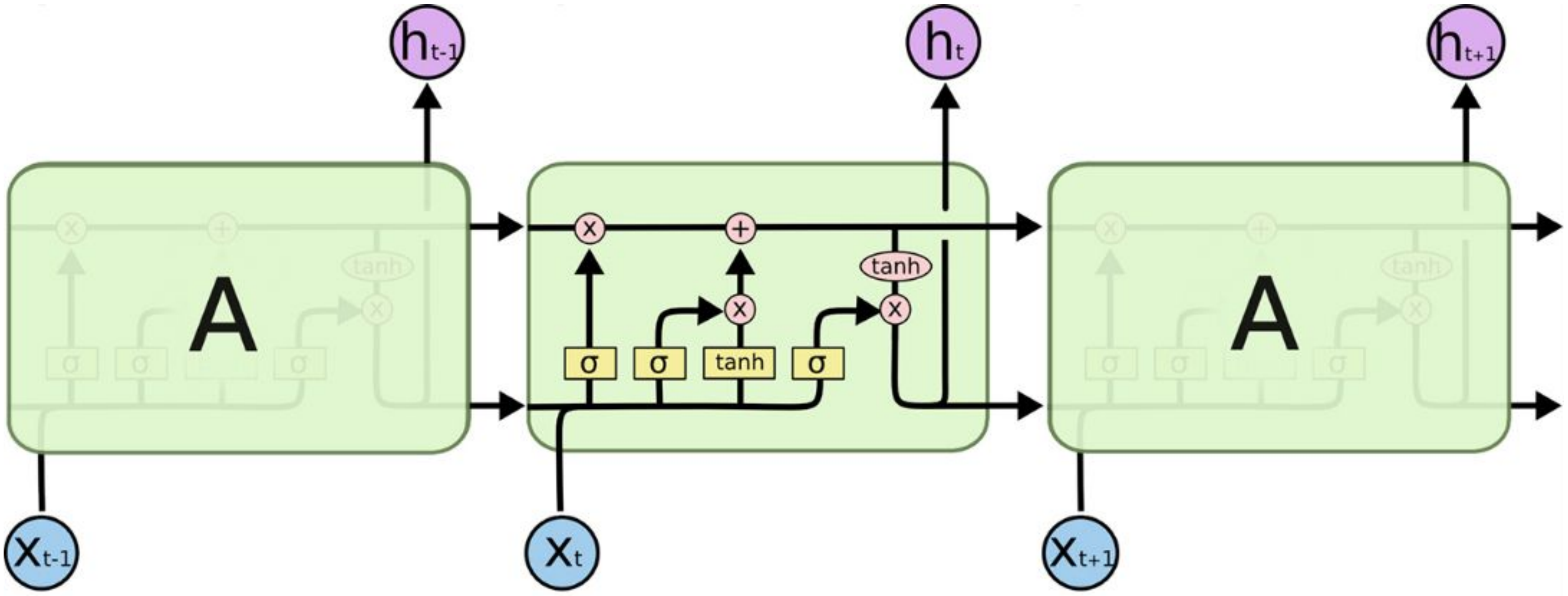
Long Short Term Memory Networks

- Long short term memory – usually called LSTM, are special kind of RNN
- Capable of long-term dependencies



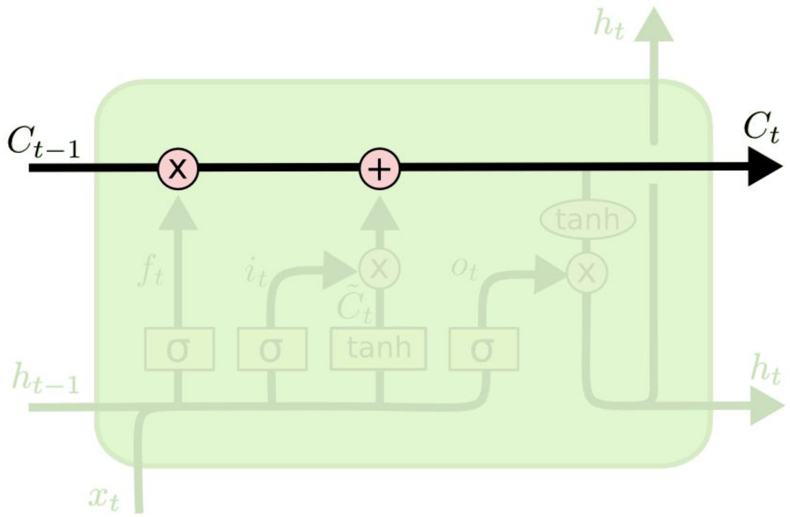
A repeating module in standard RNN

Long Short Term Memory Networks



Long Short Term Memory Networks

- Cell state is the key of LSTM
- It runs straight down the entire chain, with only some minor linear interactions
- Information can be added or deleted from this state vector via the forget and input gates.



LSTM Cell State

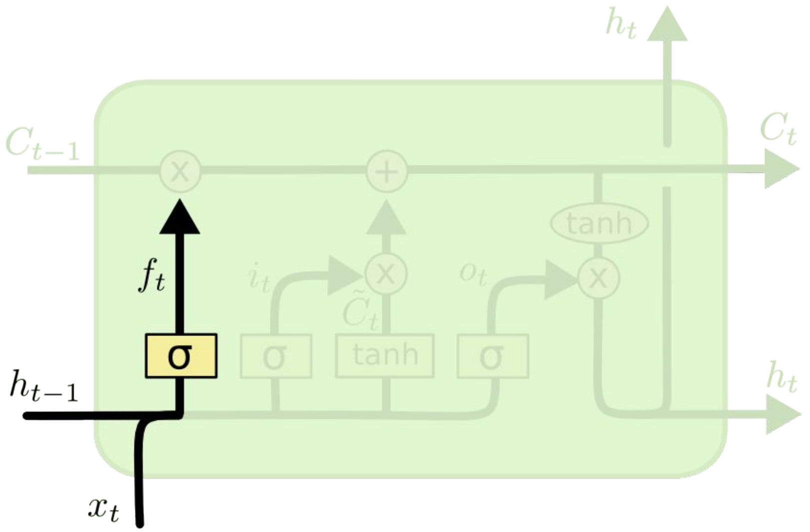
Illustration

- Want to remember person & phone number
- Forget gate will remove existing information of a prior subject when a new one is encountered.
- Input gate "adds" in the information for the new subject.

Long Short Term Memory Networks

Step – 1

Identify the information that and it will be thrown from the cell state. This decision is made by sigmoid layer called as **forget gate** layer



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right)$$

W_f = Weight

h_{t-1} = Output from the previous timestamp

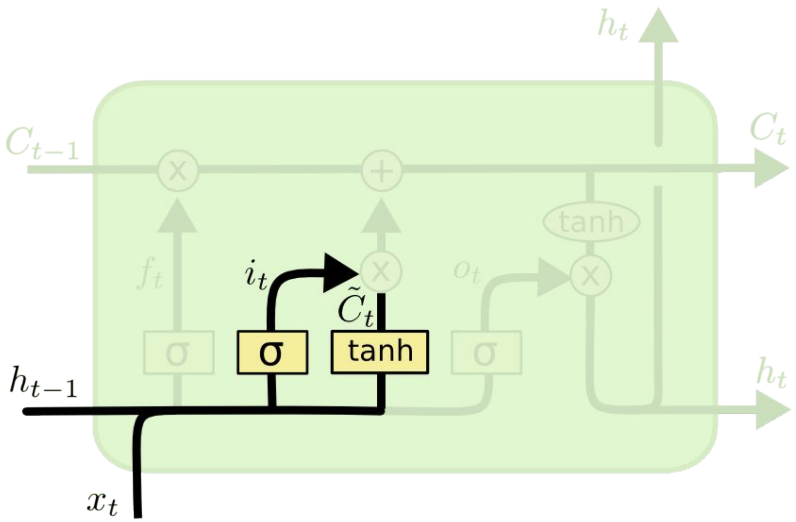
x_t = New input

b_f = Bias

Long Short Term Memory Networks

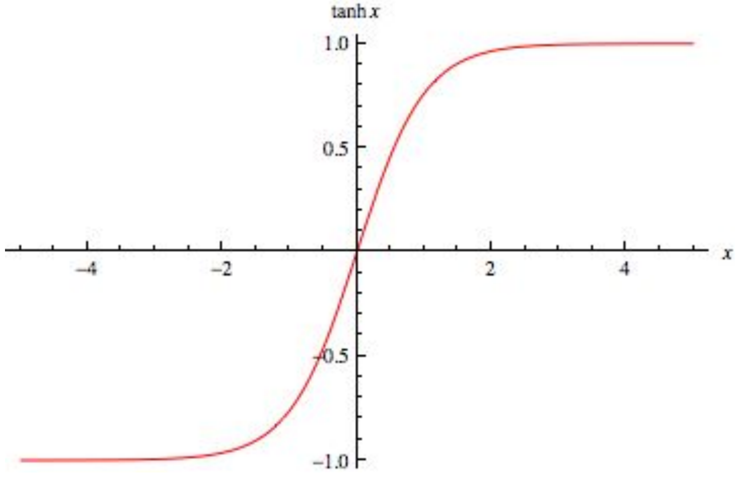
Step – 2

Decide what is new information we are going to store in cell state. This whole process comprises of the following steps. A **sigmoid layer** called the **input gate** layer, decide which values will be updated. Next a **tanh layer** create a vector of a new candidate values, that could be added to the state



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

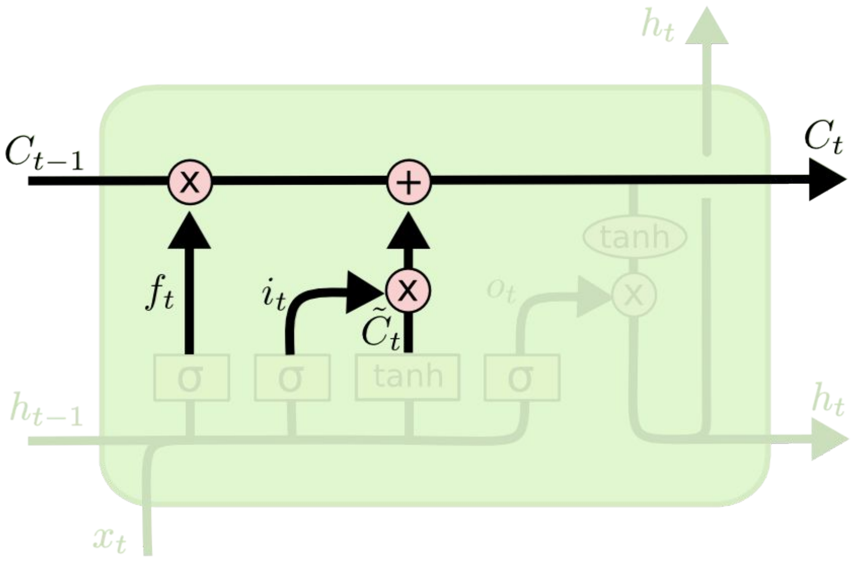
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$



Long Short Term Memory Networks

Step – 3

Update the old state C_{t-1}, into the new cell state C_t. First, we multiply the old state (c_{t-1}) by f_t, forgetting the things we forget earlier. Then we add $i_t * \tilde{C}_t$. This is a new candidate values, scaled by how much we decide to update each state value.

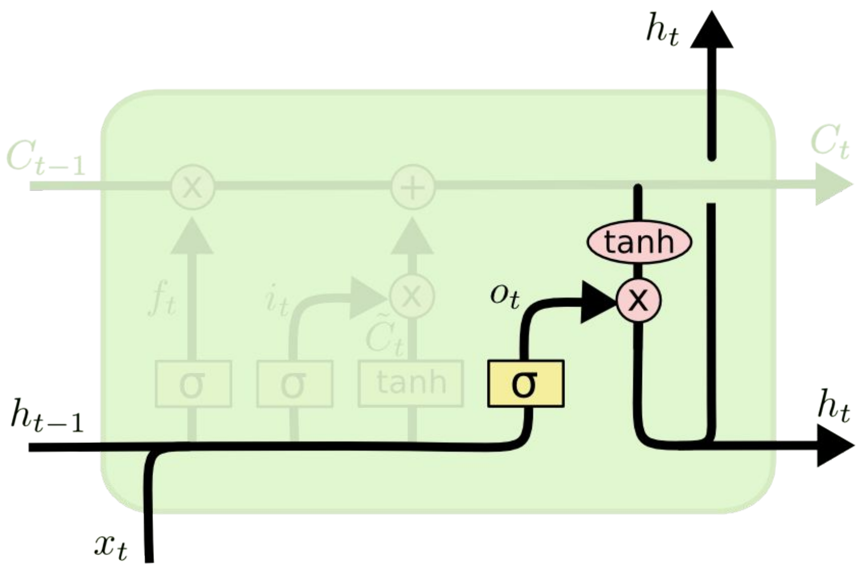


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Long Short Term Memory Networks

Step – 4

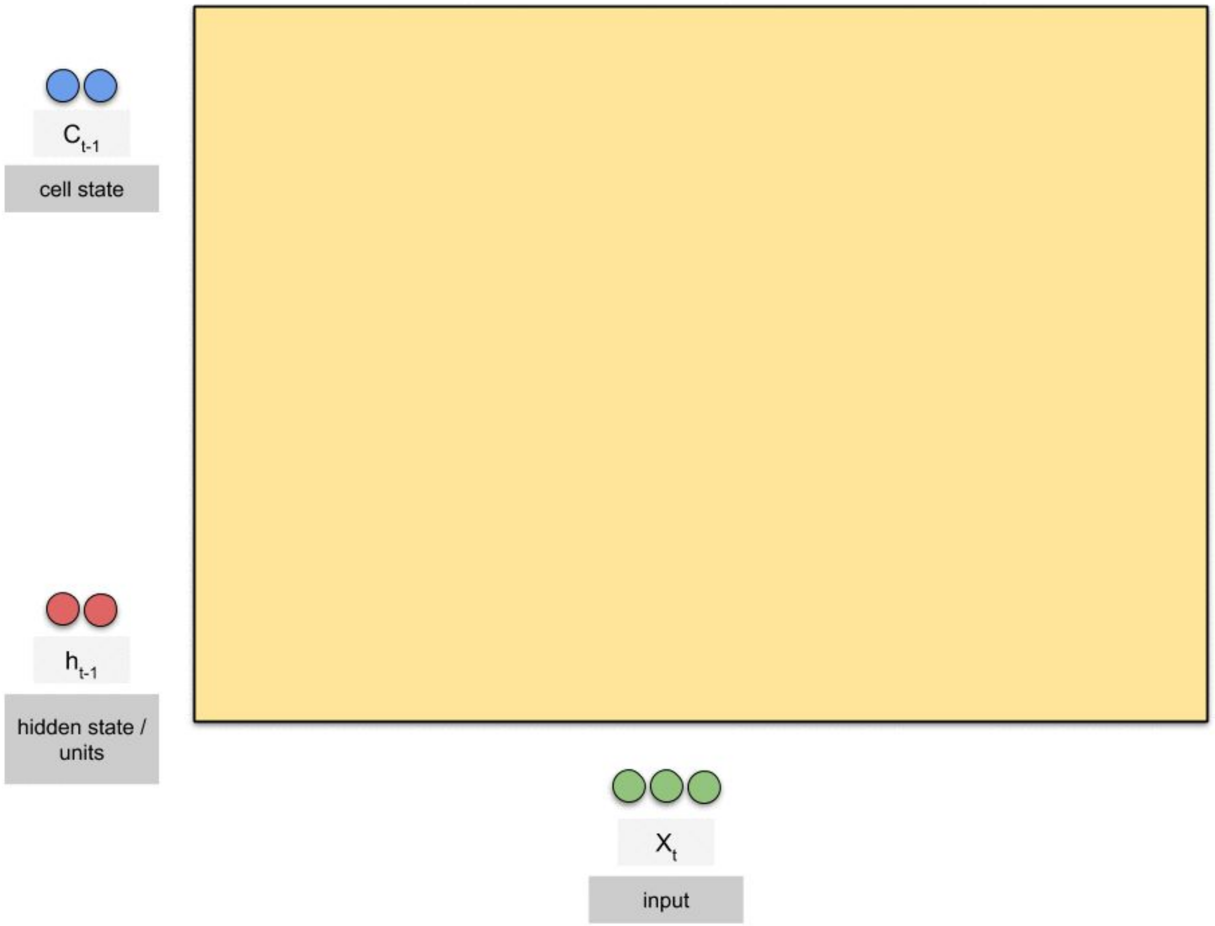
Run sigmoid layer which decide what part of the cell state we are going to output. Then we put the cell state through tanh (push value between -1 and 1) and then multiply it by the output of the sigmoid gate, so that we only output the part we decide.



$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Animated LSTM



Summary of LSTM Application Arch.

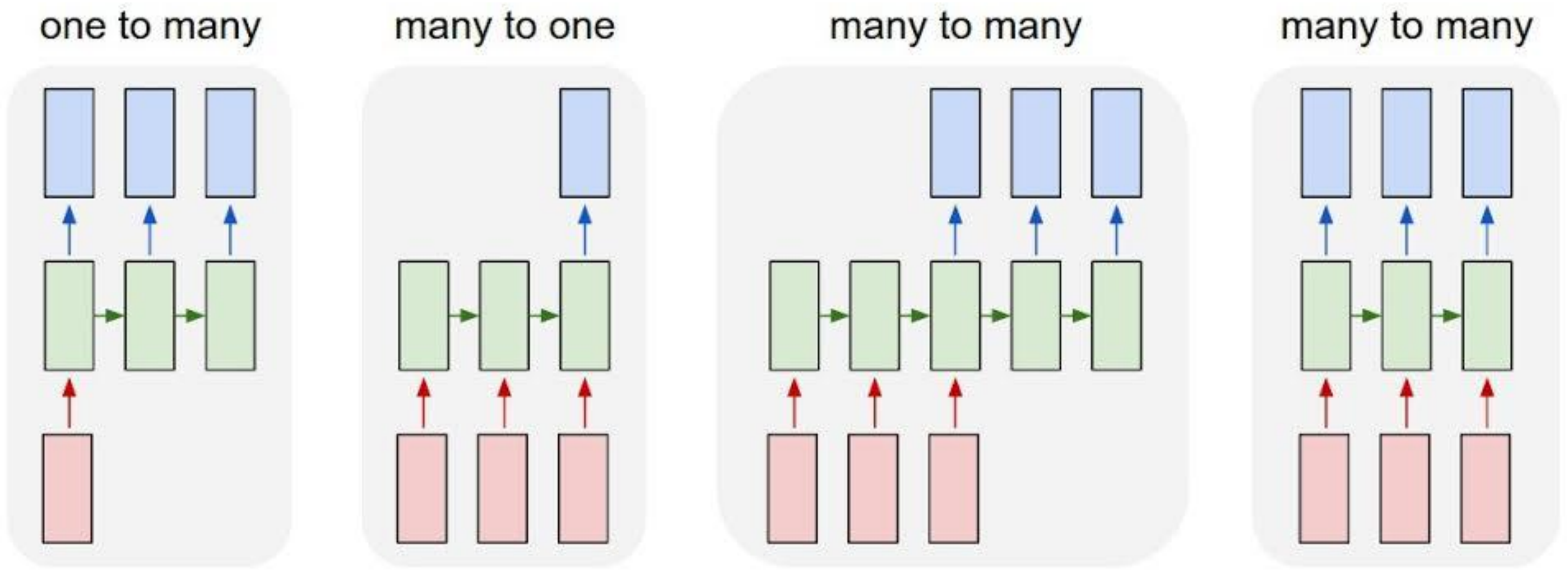


Image Captioning

Video Activity Recognize Text Classification

Video Captioning Machine Translation

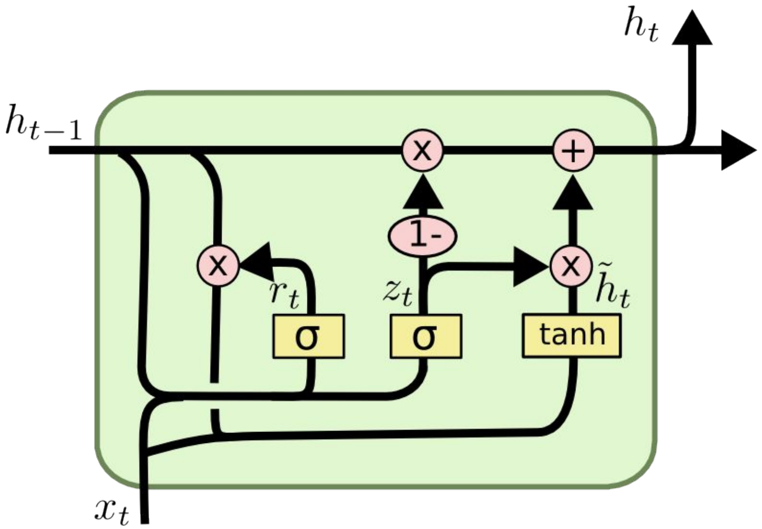
POS Tagging Language Modeling

Outline

- Why not feed-forward neural networks
- What is recurrent neural networks
  - Issue with recurrent neural networks
  - Vanishing and exploding gradient
- Introduction to long-short term memory
- **Gated Recurrent Unit**

Gated Recurrent Unit (GRU)

- A very simplified version of the LSTM
  - Merge forget & input gate into single "update" gate
  - Merge cell and hidden state
- Has fewer parameters than LSTM & has been shown to outperform LSTM on some tasks.



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

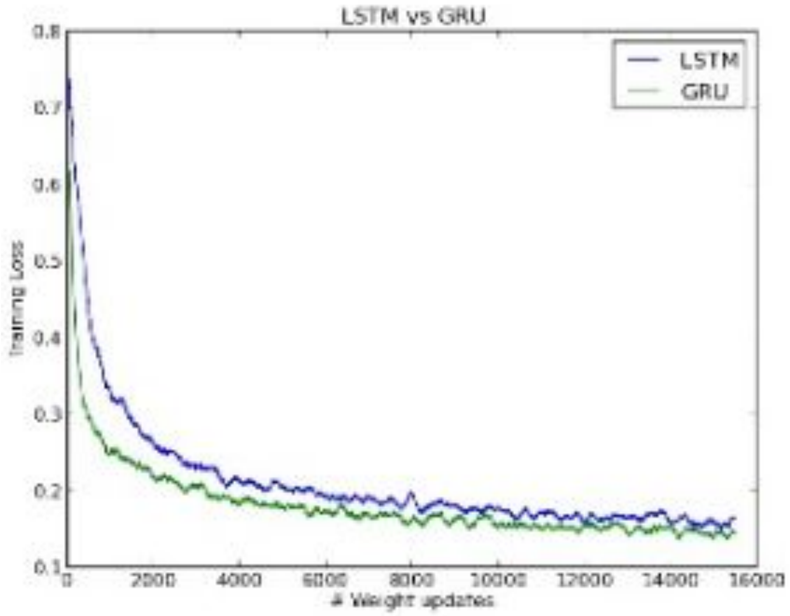
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

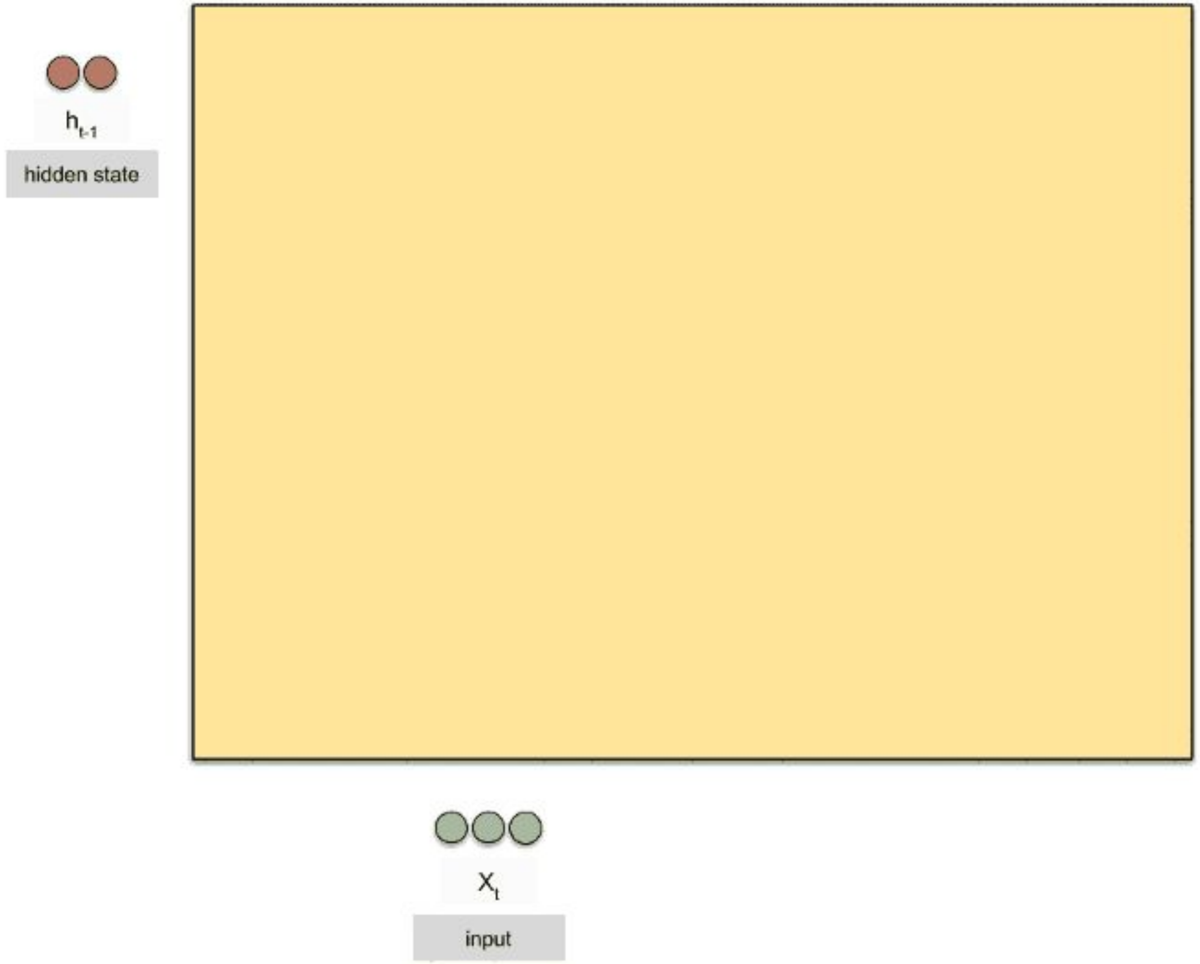
[Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, Cho *et al.*, 2014]

GRU vs LSTM

- GRU has significantly fewer parameters and trains faster.
- Experimental results comparing the two are still inconclusive, many problems they perform the same, some better than the other on some tasks.



Animated GRU



Attention Layer (Advanced Topic)

- For many applications, it helps to add "attention" to RNNs.
- The Attention mechanism in Deep Learning is based off this concept of directing your focus, and it pays greater attention to certain factors when processing the data
- Allows network to learn to attend to different parts of the input at different time steps, shifting its attention to focus on different aspects during its processing.

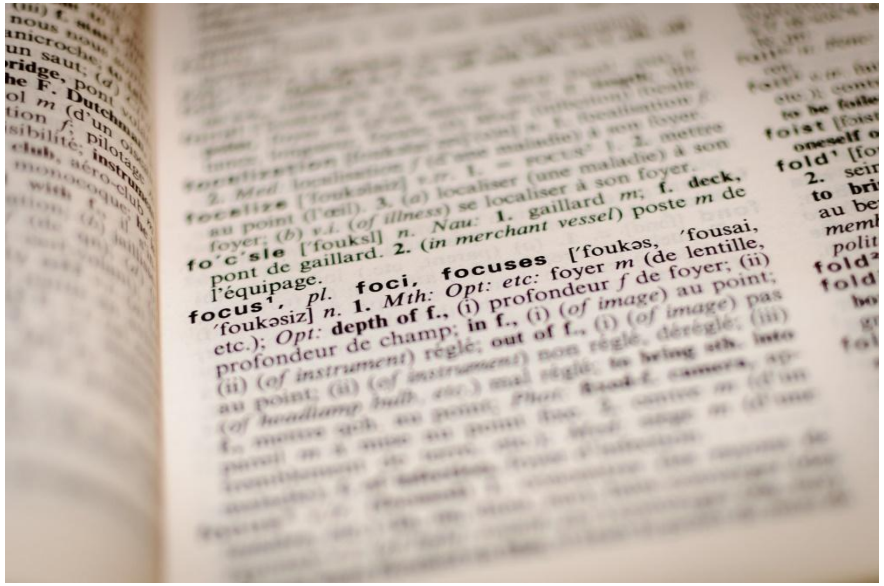
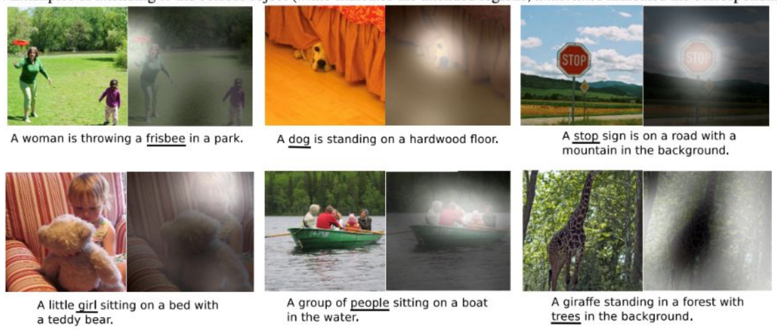


Figure 4. Examples of attending to the correct object (*white* indicates the attended regions, *underlines* indicated the corresponding word)



Thank You