1. RNN
2. LSTM
3. GRU

# 1. RNN ( recurrent NN )

Great  for modeling sequential data.

Eg of sequential data - speech, audio, text etc

Use cases - speech recognition, language translation, stock prediction etc

# Text can also be sequence data

T e x t

can also be sequence data

# Example of a sequential memory

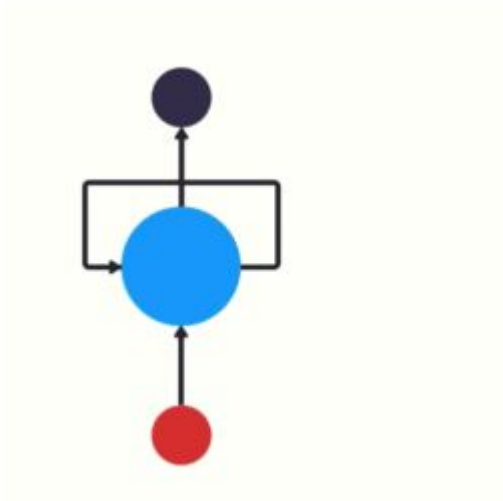
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

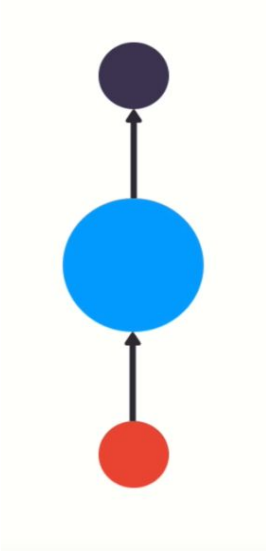
Z Y X W V U T S R Q P O N M L K J I H G F E D C B A

H I J K L M N O P Q R S T U V W X Y Z

THAT MEANS SEQ. DATA IS EASIER TO REMEMBER!

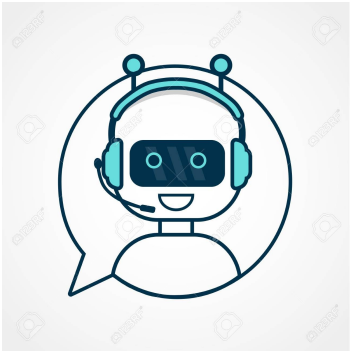
& RNNs ARE ABSTRACT CONCEPT OF SEQ. MEMORY





# CHATBOT EXAMPLE

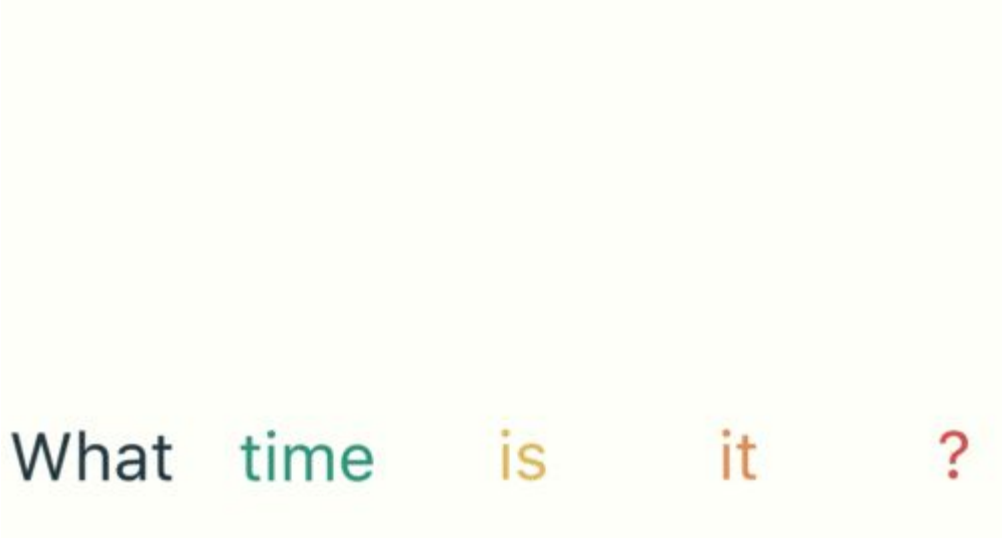
YOU: WHAT TIME IS IT ?

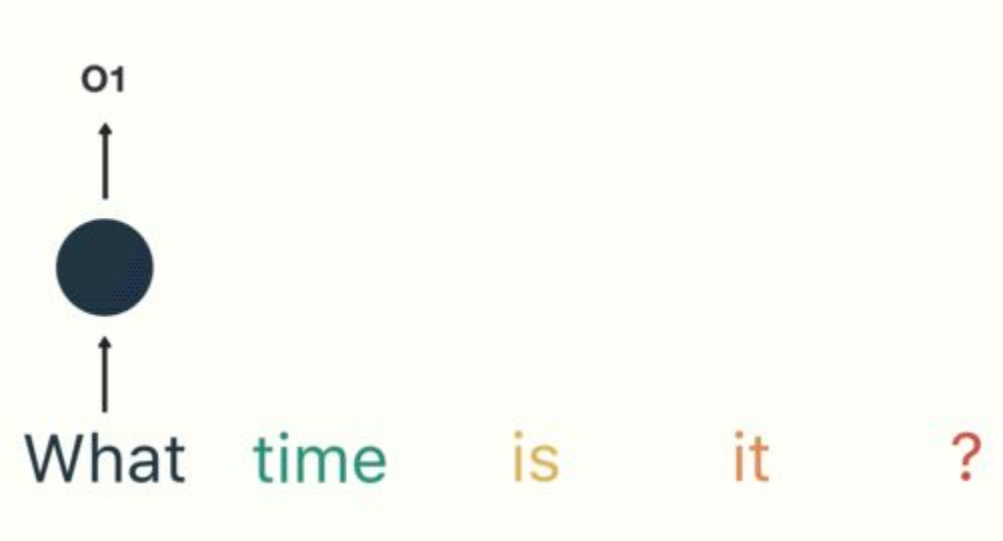


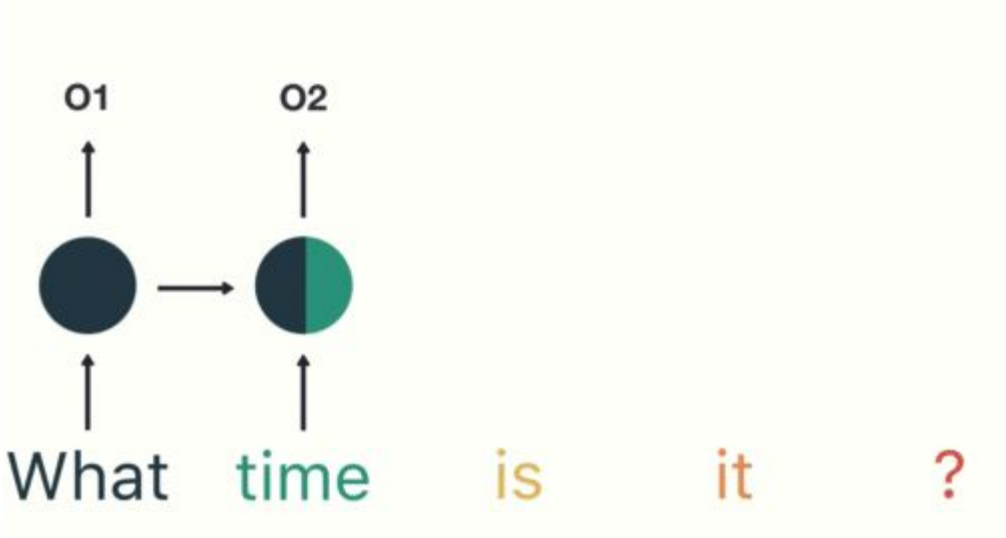
ASKING FOR TIME ….

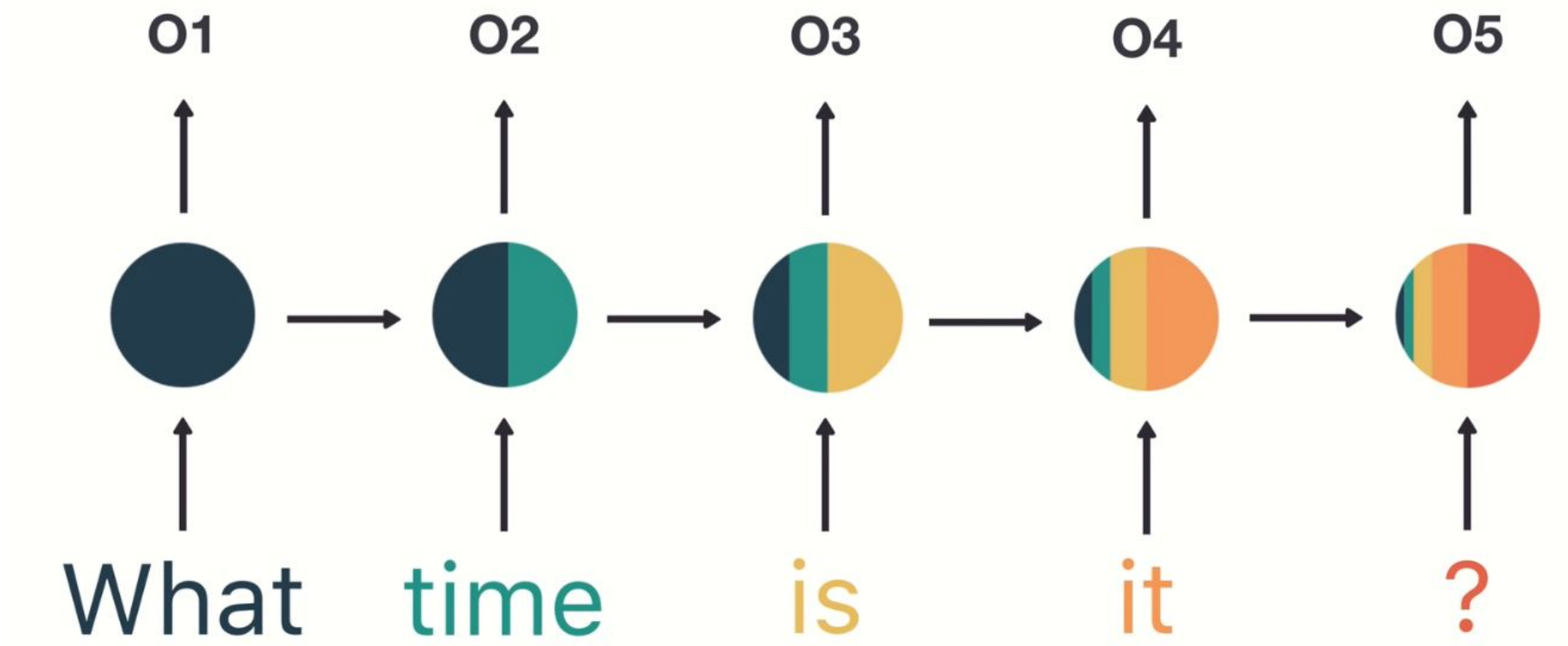
What time is it?

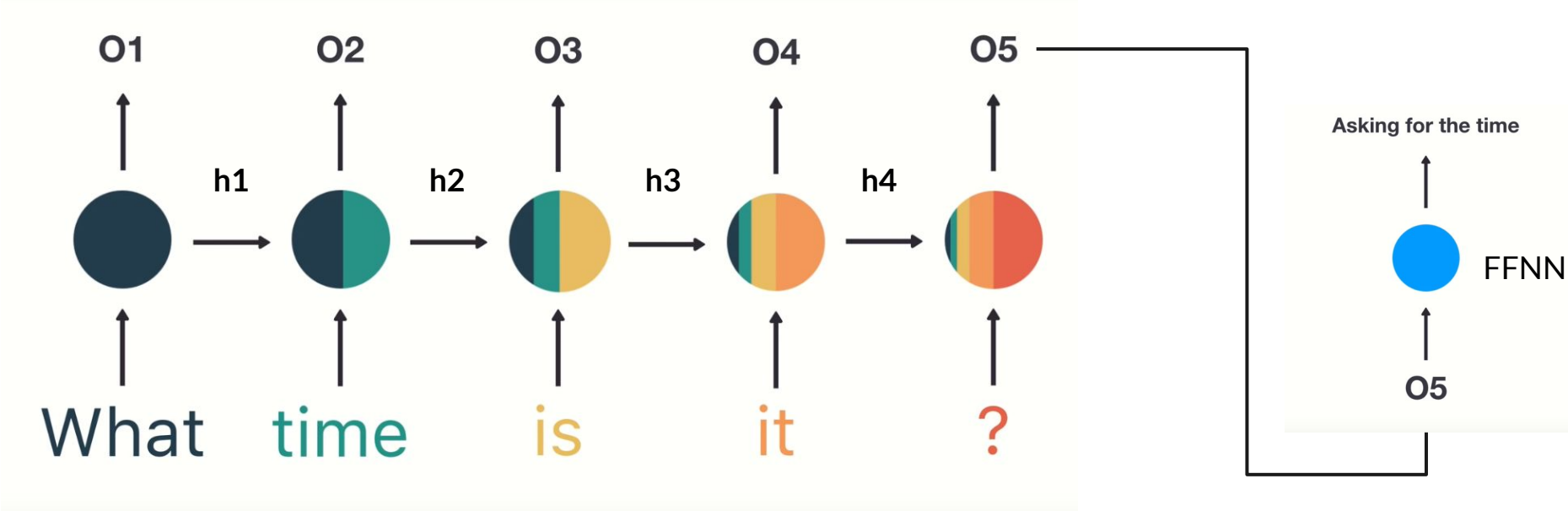
What time is it ?


What time is it ?

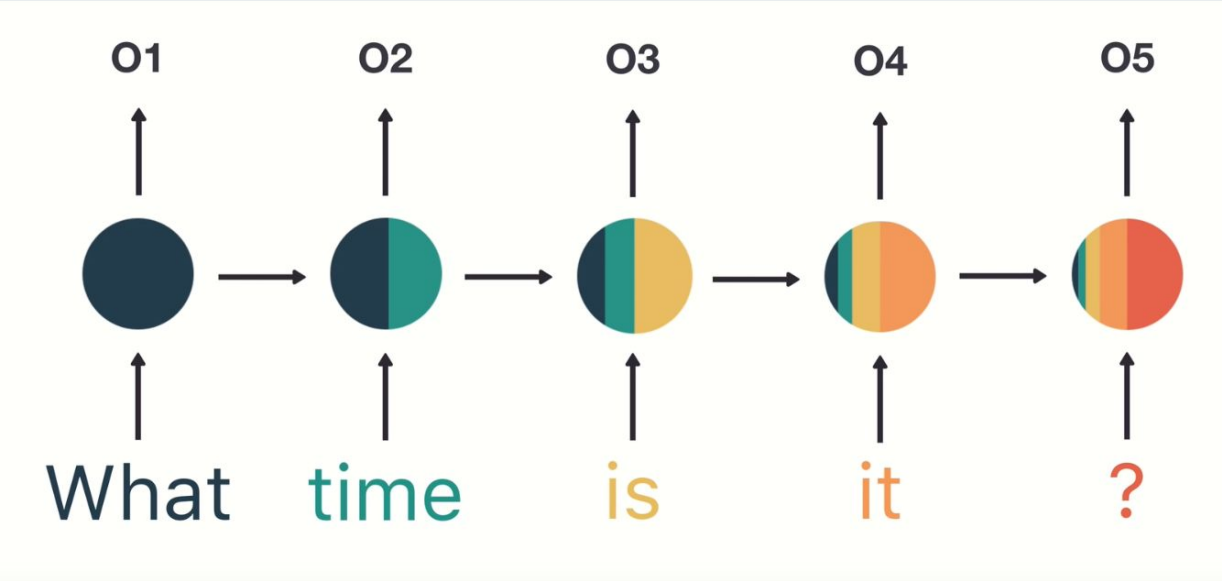

01
What time is it ?


O1
O2
What
time
is
it
?


O1
O2
O3
O4
O5
What
time
is
it
?


O1
O2
O3
O4
O5
h1
h2
h3
h4
What
time
is
it
?
Asking for the time
FFNN
O5

```
# PSEUDO CODE

rnn = RNN()
ff = FeedForwardNN()
hidden_state = [0.0, 0.0, 0.0, 0.0]

for word in input_:
    output, hidden_state = rnn(word, hidden_state)

prediction = ff(output)
```




RNNs suffer from vanishing gradient hence they have short term memory.

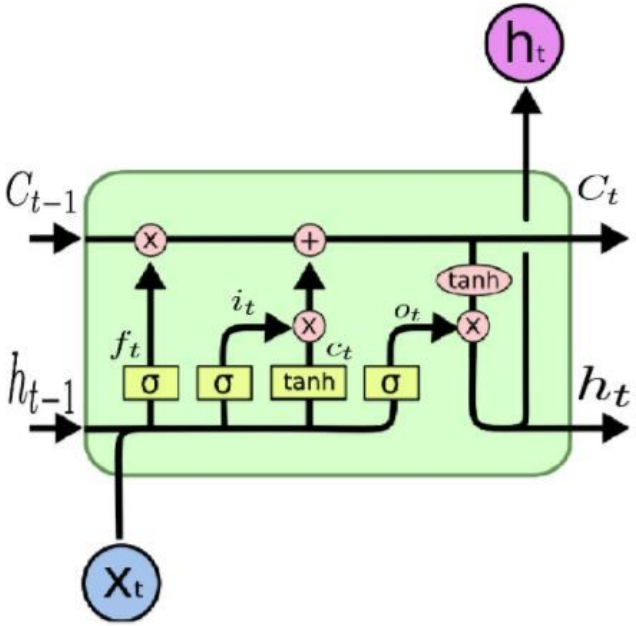
**new weight = weight - learning rate*gradient**

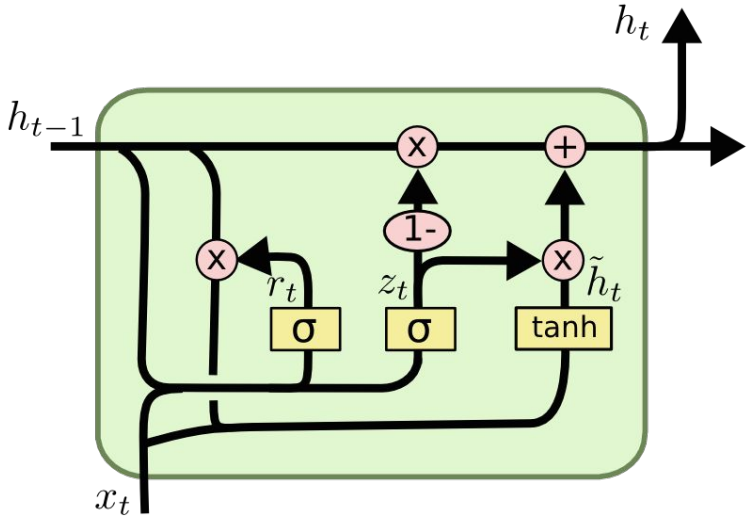
**2.0999 = 2.1 - 0.001**

Not much of a difference

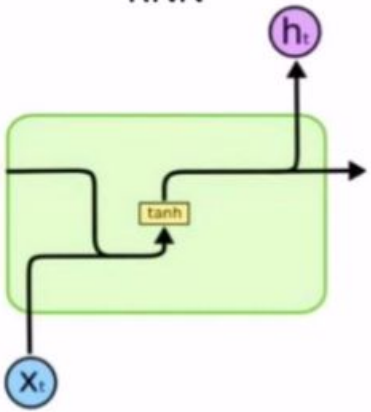
update value

## LSTM and GRU ( comes to rescue to tackle short term memory issue )

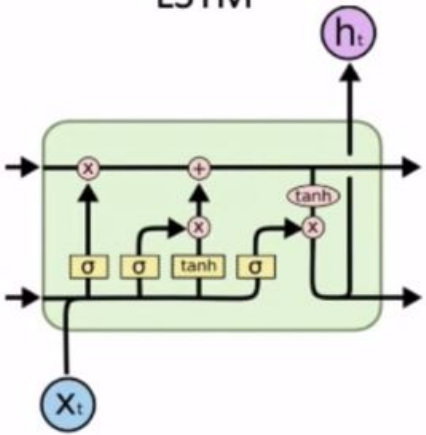


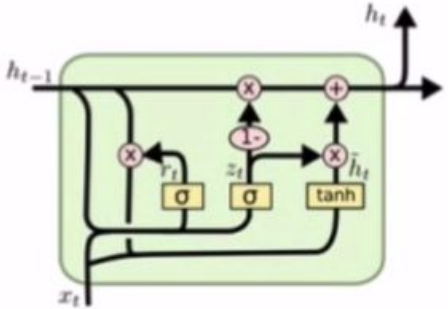
RNN




LSTM

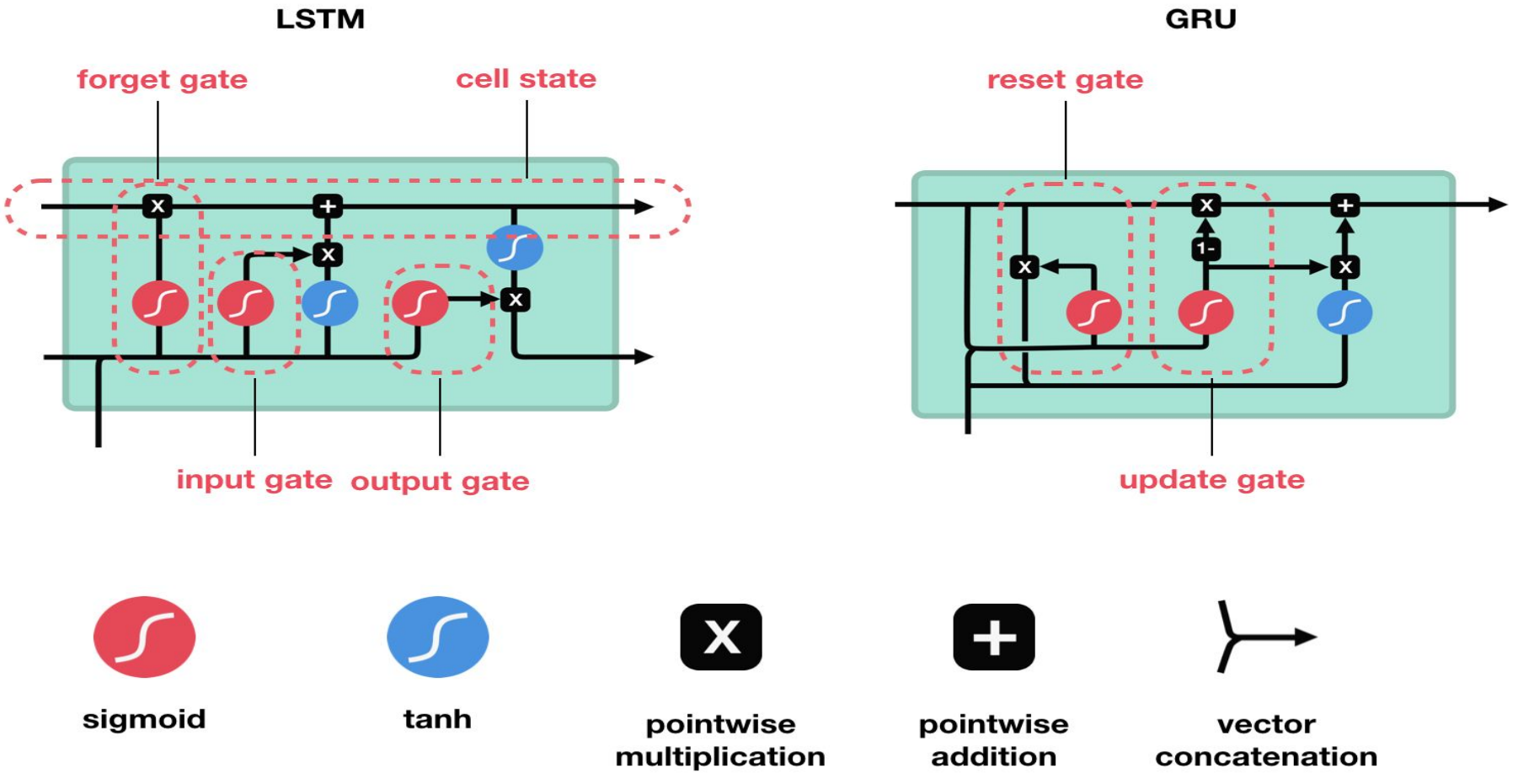



GRU




1. Simple arch.
2. Good for speed as it has less tensor ops.
3. Suffers short term memory.

1. Quite complex.
2. Good for modeling longer sequences which has long term dependencies.
3. Conquers short term memory.


LSTM
forget gate
cell state
input gate
output gate
GRU
reset gate
update gate
sigmoid
tanh
pointwise multiplication
pointwise addition
vector concatenation

## USE cases -

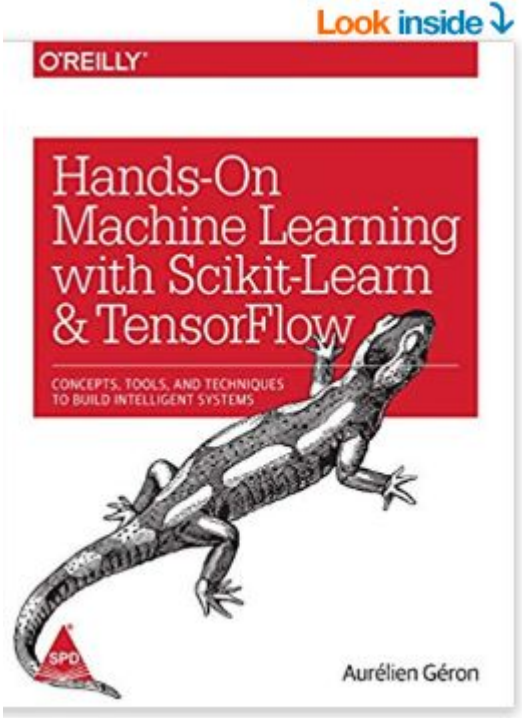
1. Speech recognition
2. Generating caption for a pic
3. Text generation

# Customer review example-

***5.0 out of 5 stars***

**A perfect book for ML Scikit and Tensorflow**

**This is one of the best books you can get for someone who is just starting out in ML, in its libraries such as Tensorflow, *It covers the basics very good.* As a book, it is 5/5**

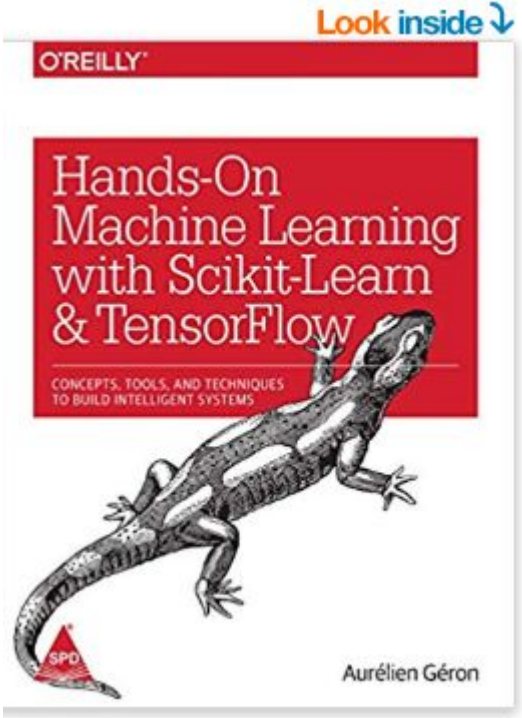


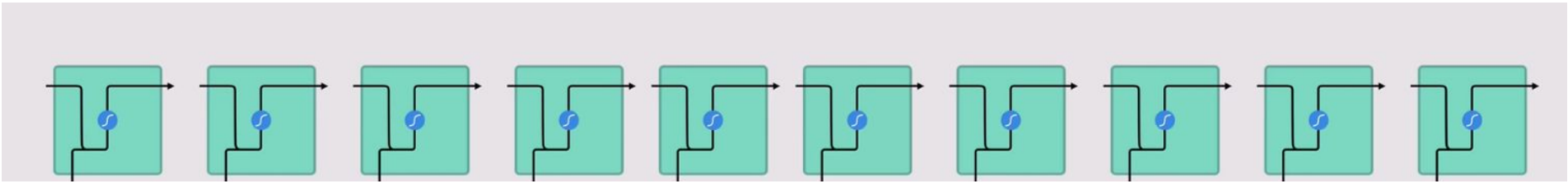
# Customer review example-

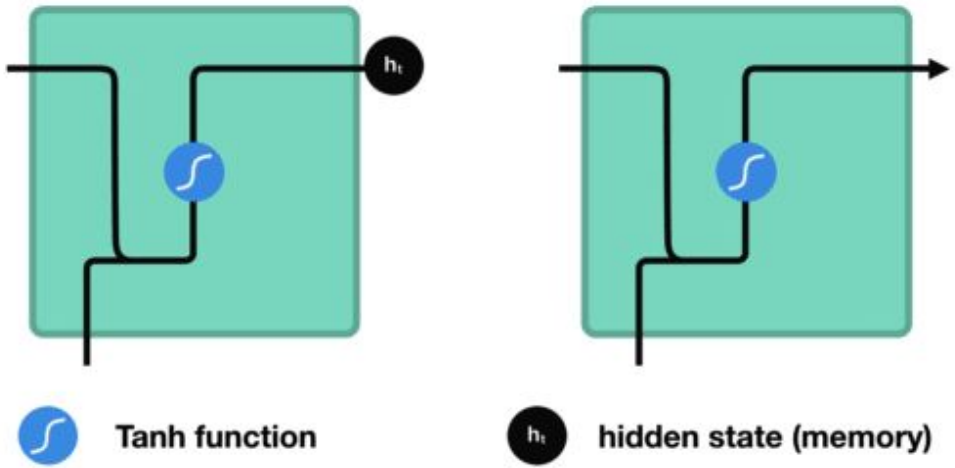
***5.0 out of 5 stars***

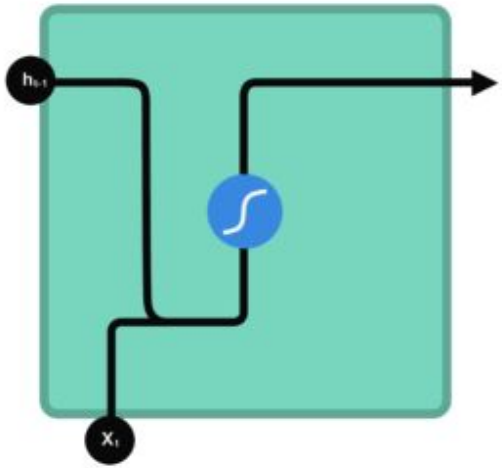
**A perfect book** for ML Scikit and Tensorflow

This is one of the **best books** you can get for someone who is just starting out in ML, in its libraries such as Tensorflow, ***It covers the basics very good.*** **As a book, it is 5/5**




perfect
book
for
ML
Scikit
and
Tensorflow
This
is
one


$h_t$
Tanh function
$h_t$
hidden state (memory)




Tanh function

$h_t$ new hidden state

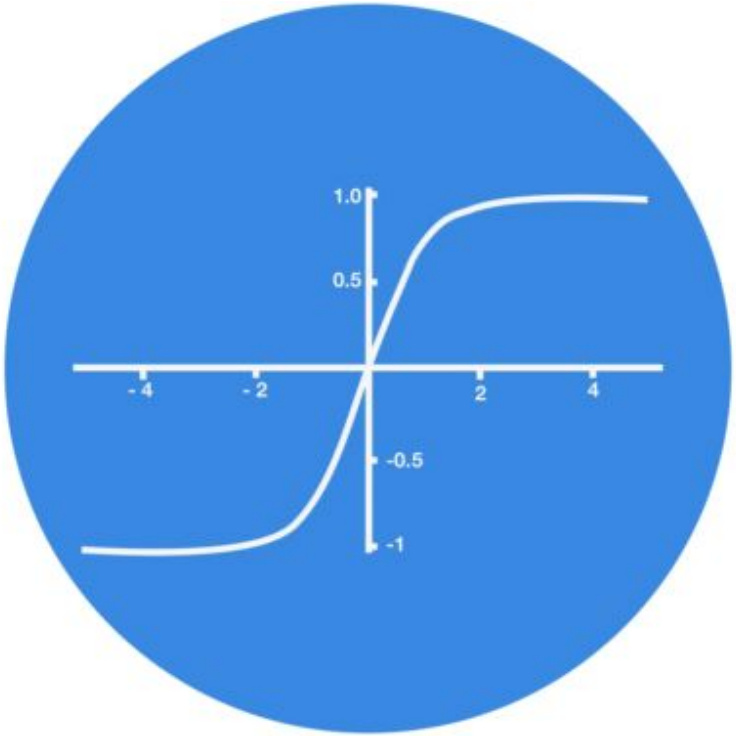
$h_{t-1}$ previous hidden state

$x_t$ input

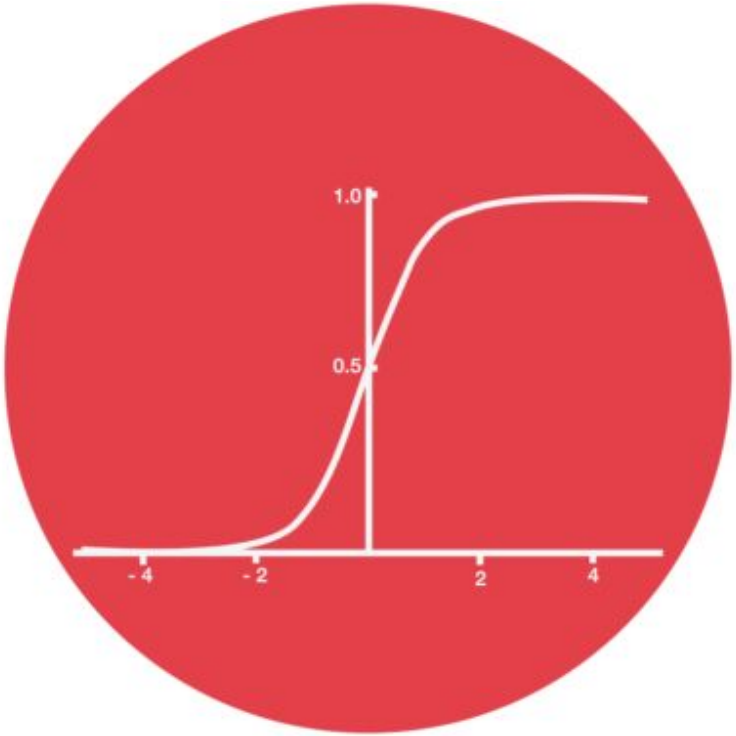
concatenation

# Tanh function and Sigmoid function

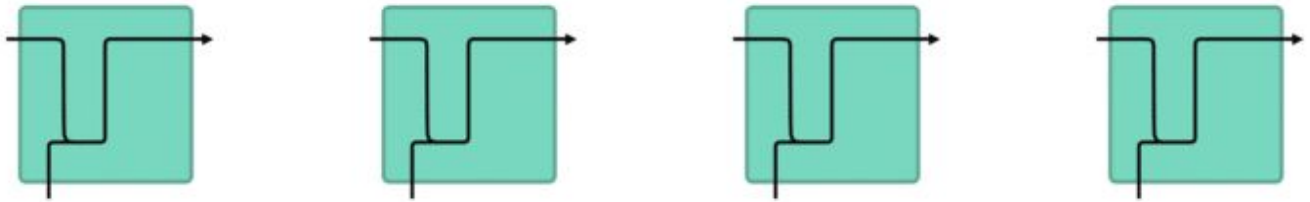
| |
|---|
| 5 |
| 0.1 |
| -0.5 |



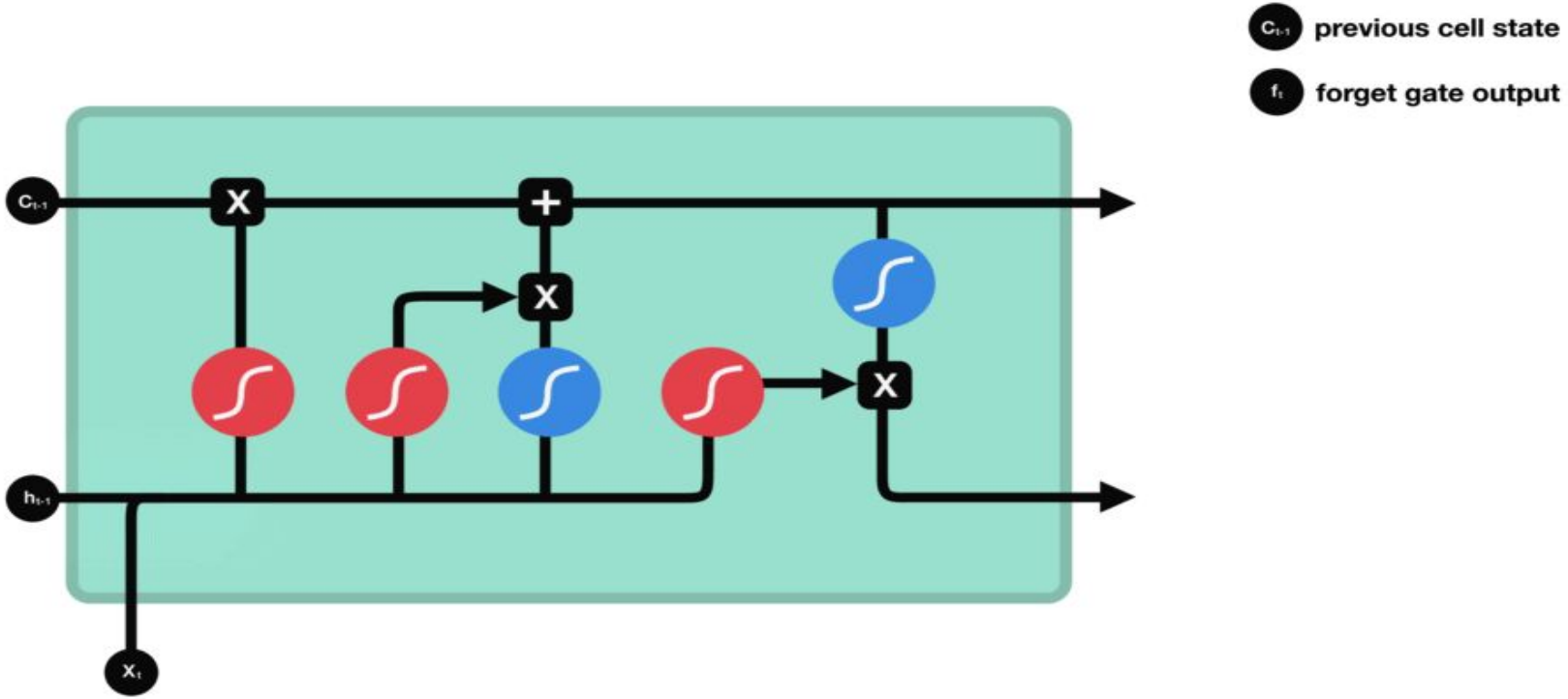
| |
|---|
| 5 |
| 0.1 |
| -0.5 |



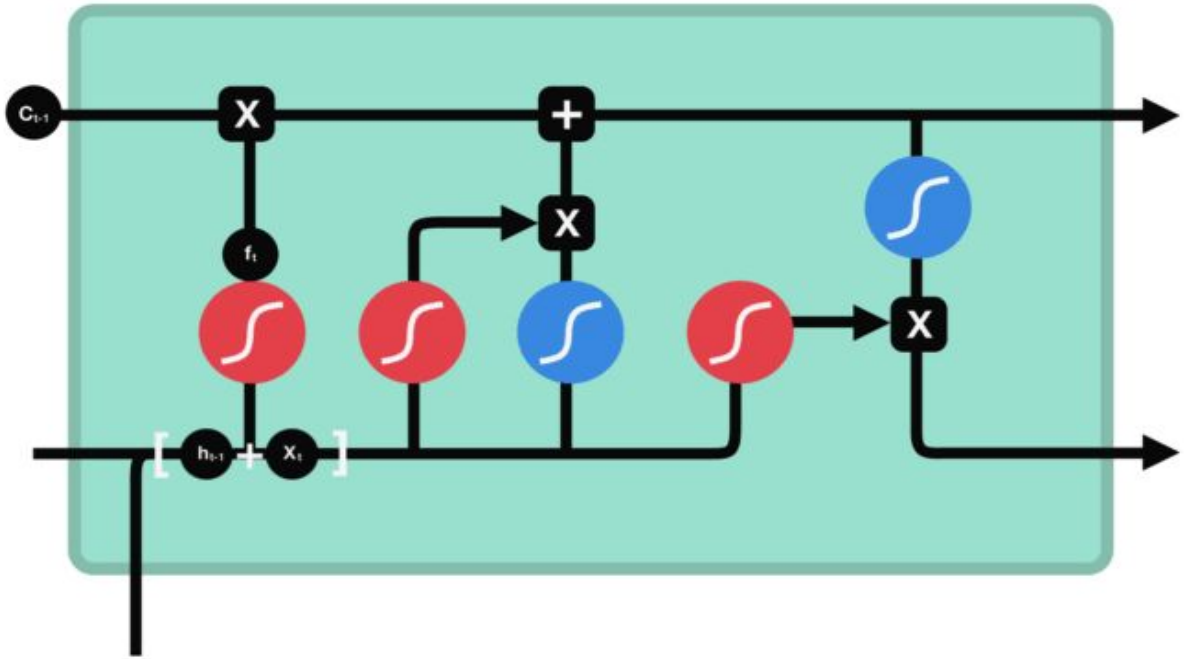
| |
|---|
| 5 |
| 0.01 |
| -0.5 |



# Forget gate



# Input Gate



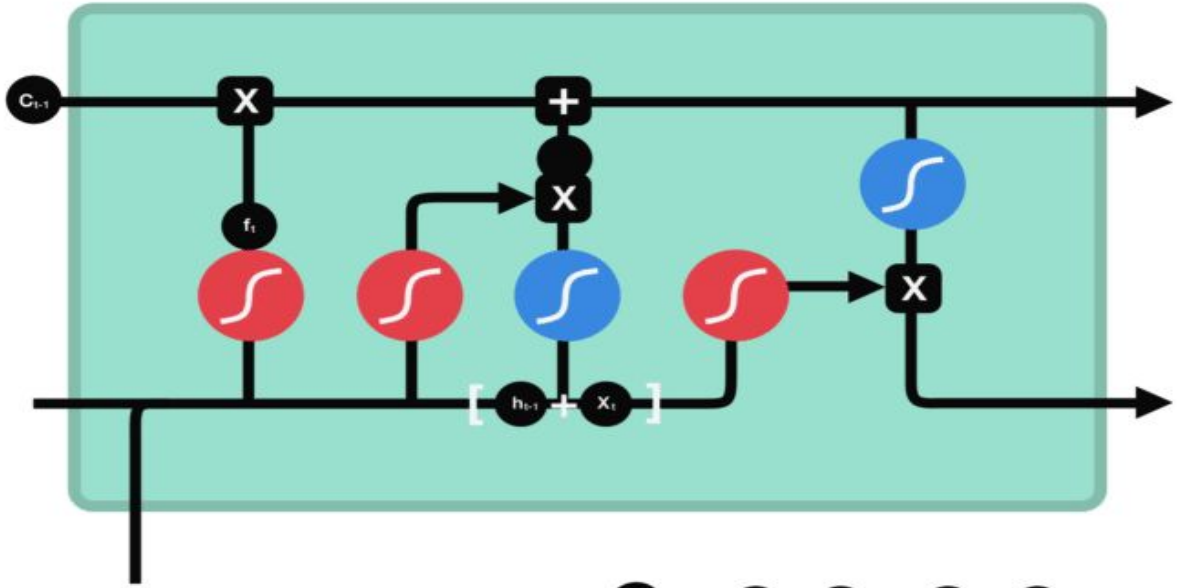

$C_{t-1}$ previous cell state

$f_t$ forget gate output

$i_t$ input gate output

$\tilde{C}_t$ candidate

# Cell State




$C_{t-1}$ previous cell state

$f_t$ forget gate output

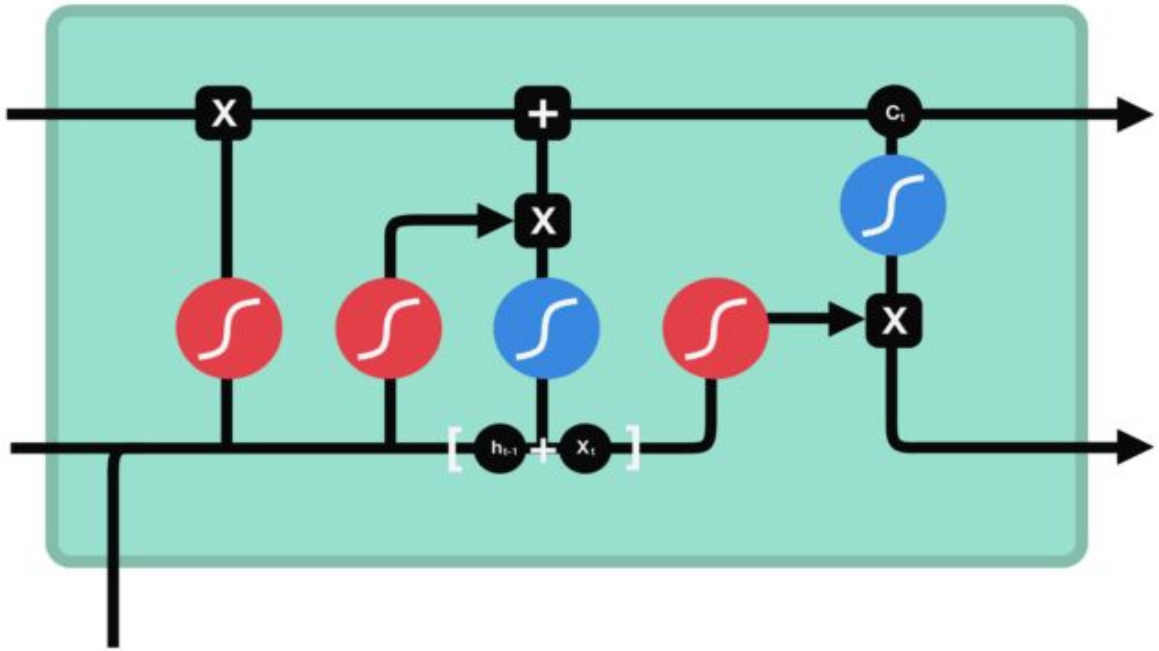
$i_t$ input gate output

$\tilde{C}_t$ candidate

$C_t$ new cell state

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

# Output Gate




$c_{t-1}$ previous cell state

$f_t$ forget gate output

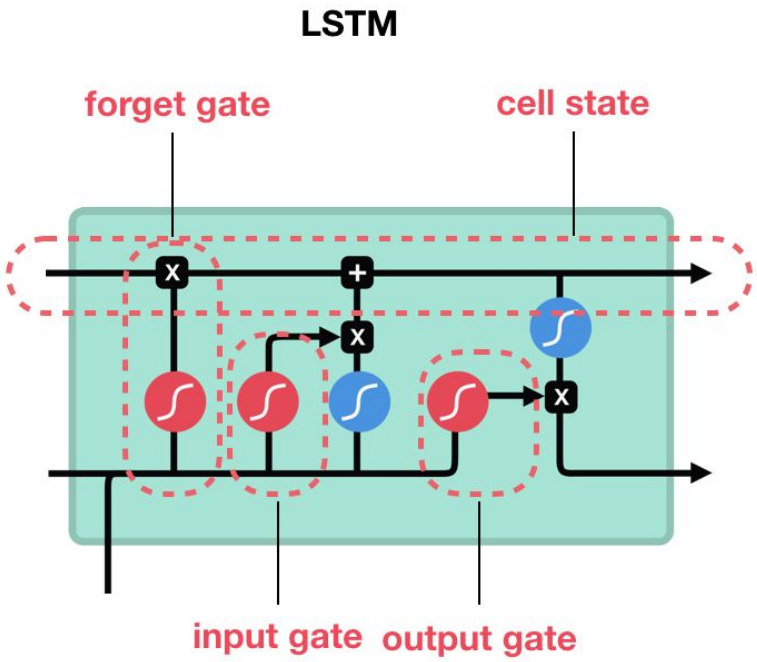
$i_t$ input gate output

$\tilde{c}_t$ candidate

$c_t$ new cell state

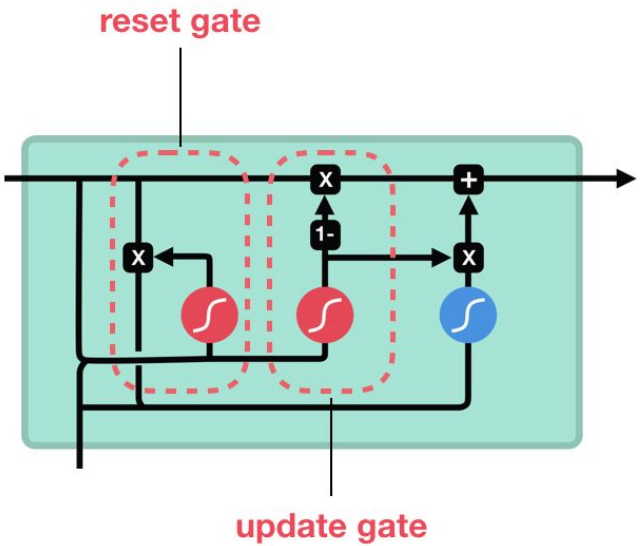
$o_t$ output gate output

$h_t$ hidden state




```python
# PSEUDO CODE

def LSTMCELL(prev_ct, prev_ht. input_):
    combine = prev_ht + input_
    # passing through sigmoid of forget gate
    ft = forget_layer(combine)
    # passing through sigmoid of input gate
    it = input_layer(combine)
    # passing through tanh between input and output gate
    candidate = candidate_layer(combine)
    # passing through sigmoid of output gate
    ot = output_layer(combine)
    ct = prev_ct * ft + combine * it
    ht = ot*tanh(ct)
    return ht, ct

ct = [0, 0, 0]
ht = [0, 0, 0]

for input_ in inputs:
    ct, ht = LSTMCELL(ct, ht, input_)

```

## GRU



# References -


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

https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

https://towardsdatascience.com/illustrated-guide-to-recurrent-neural-networks-79e5eb8049c9