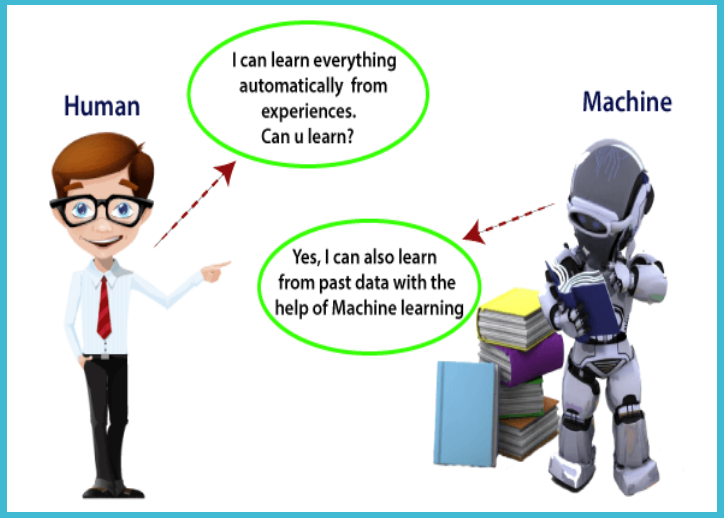
Recurrent Neural Network



Marwadi University

Department of Information and Communication Technology

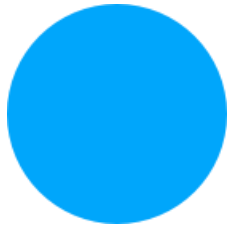
Unit 7:
Deep Learning

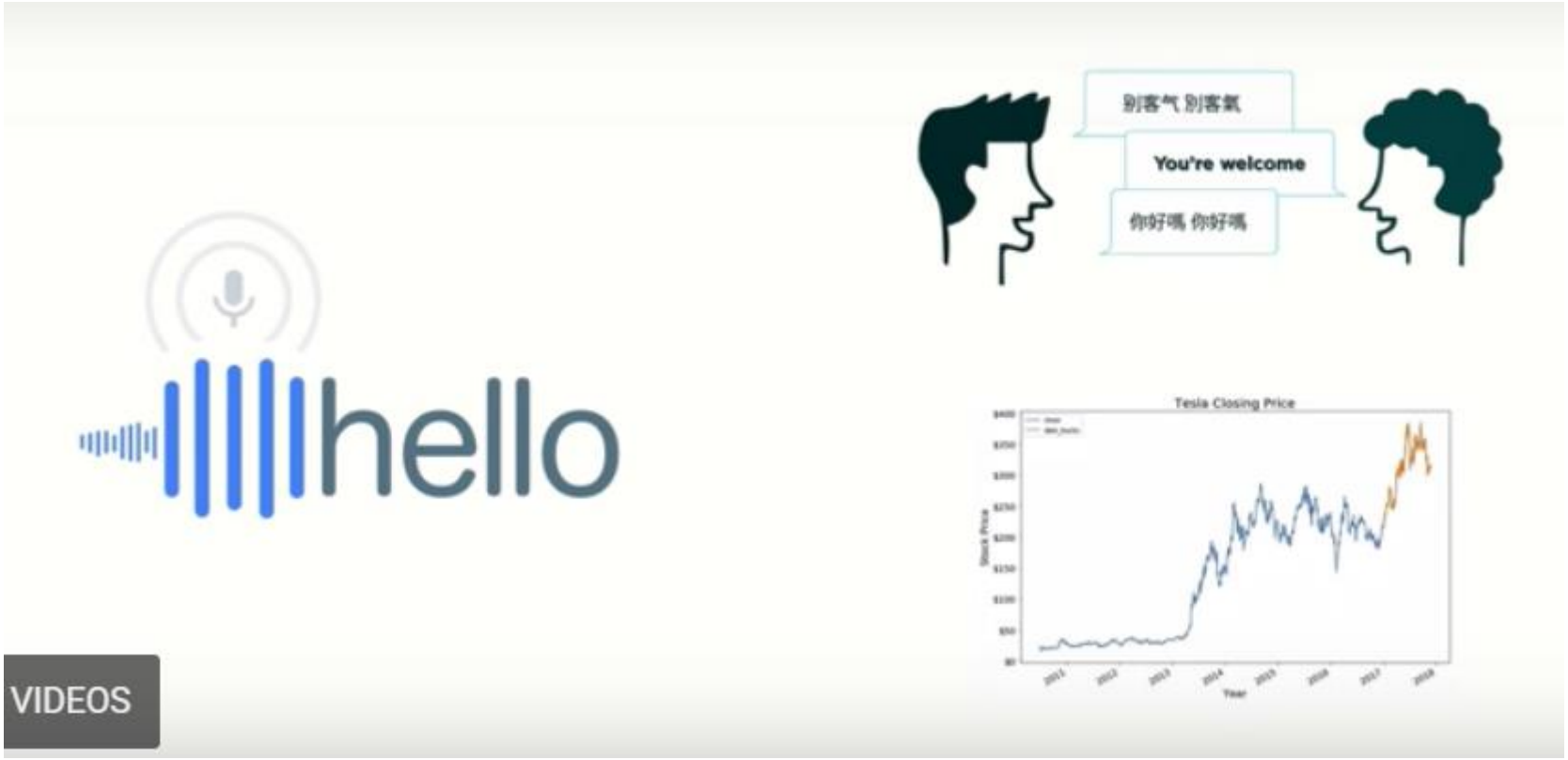
Artificial Intelligence (01CT0703)

Prof. Nishith Kotak

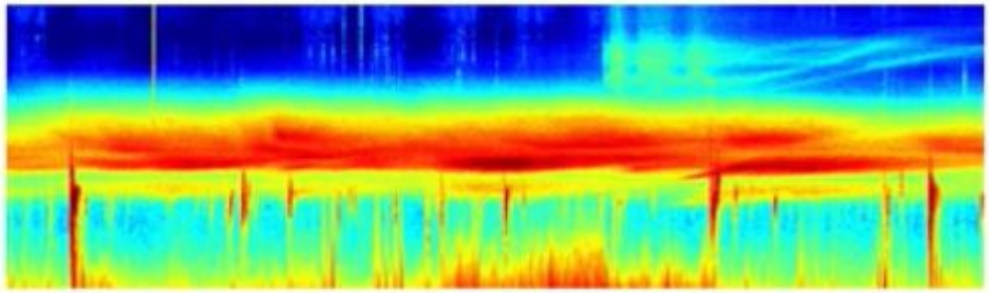
RNN

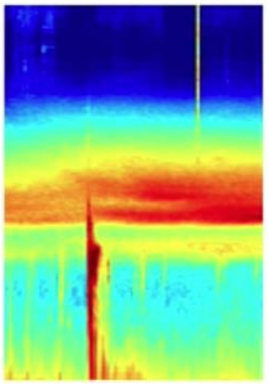
- Sequence Modeling is the task of predicting what word/letter comes next

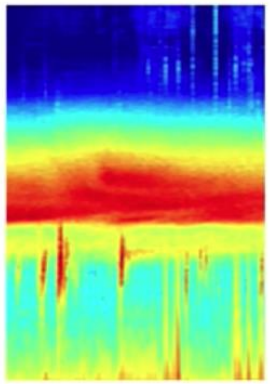


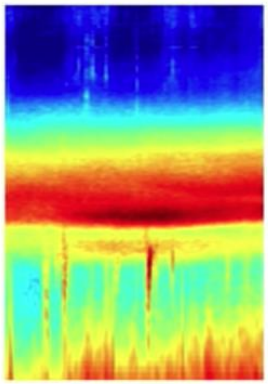


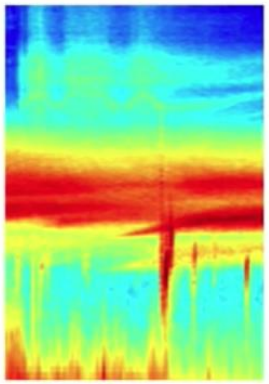
- Inputs can be Videos, Speech, Text, Data, etc…

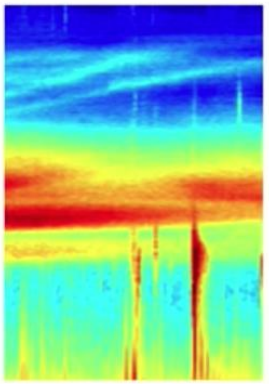




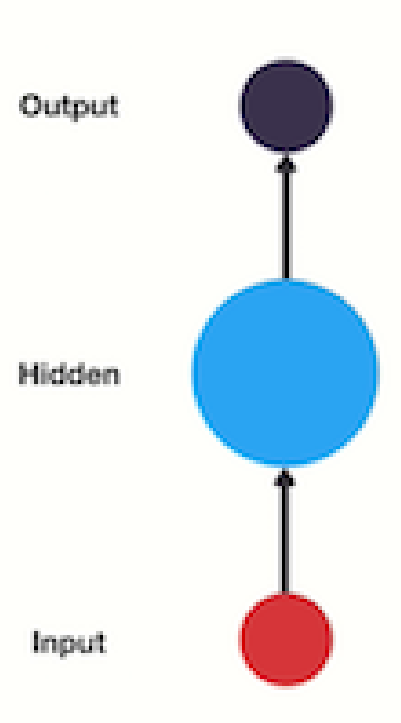




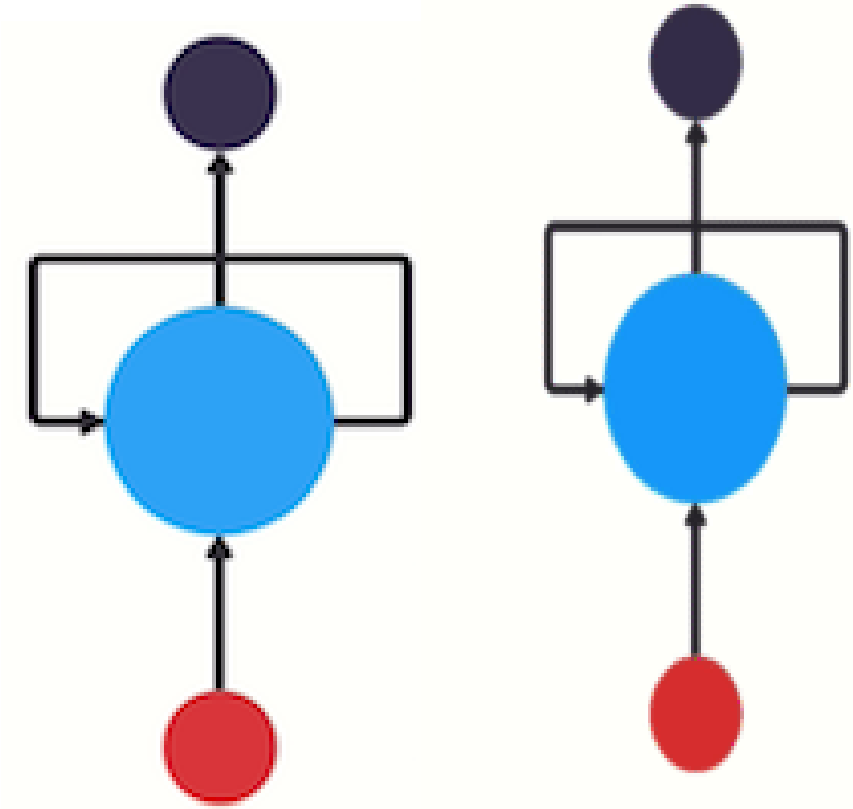




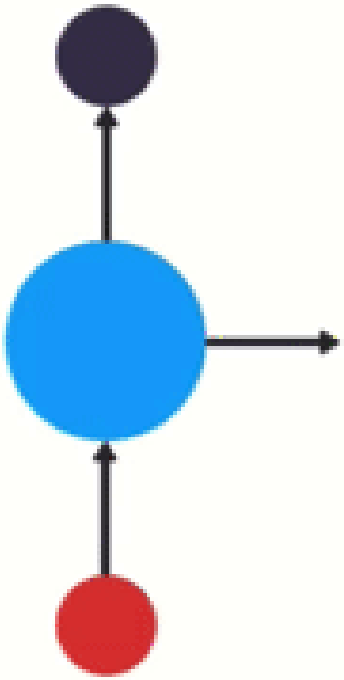
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z



Feed Forward Network

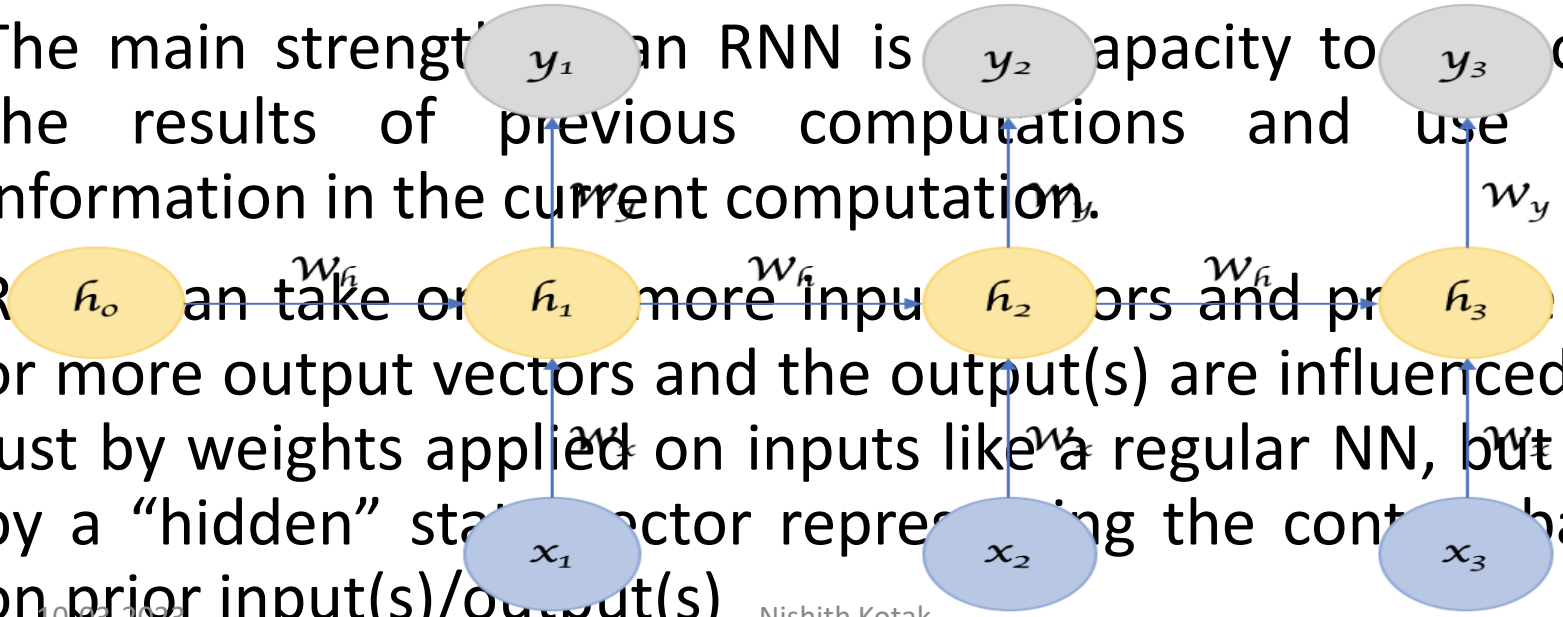


Recurrent Neural Network



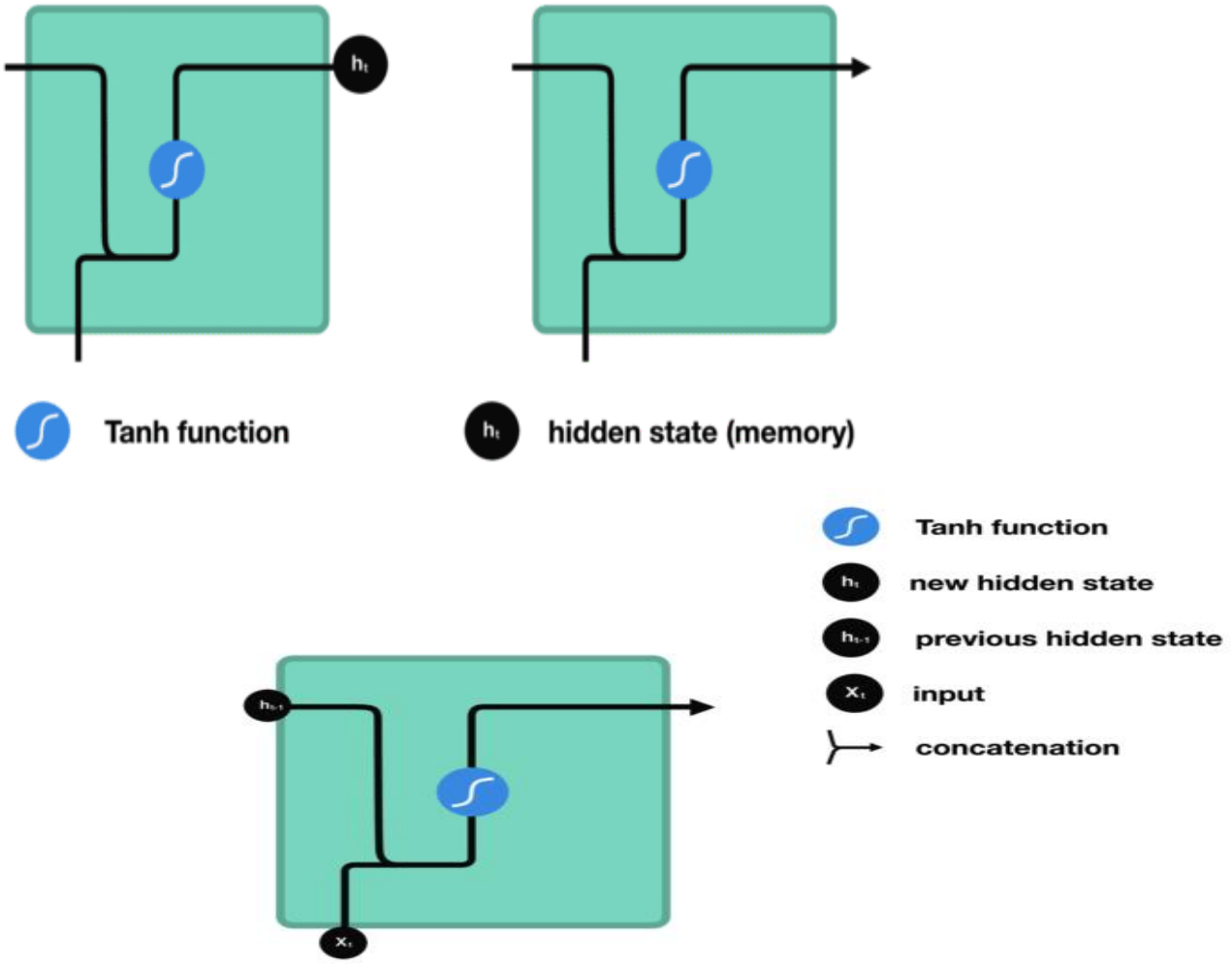
Recurrent Neural Network

- An RNN recursively applies a computation to every instance of an input sequence conditioned on the previous computed results.
- RNN remembers the past and it's decisions are influenced by what it has learnt from the past.
- The main strength of an RNN is its capacity to memorize the results of previous computations and use that information in the current computation.
- RNNs can take one or more input vectors and produce one or more output vectors and the output(s) are influenced not just by weights applied on inputs like a regular NN, but also by a "hidden" state vector representing the context based on prior input(s)/output(s)



https://medium.com/dair-ai/deep-learning-for-nlp-an-overview-of-recent-trends-d0d8f40a776d

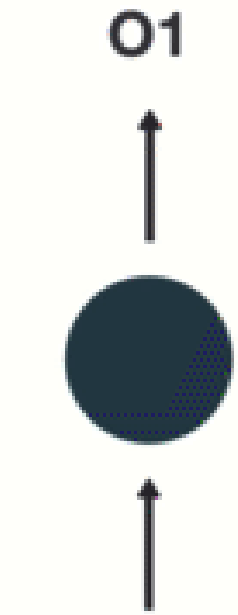
Recurrent Neural Network



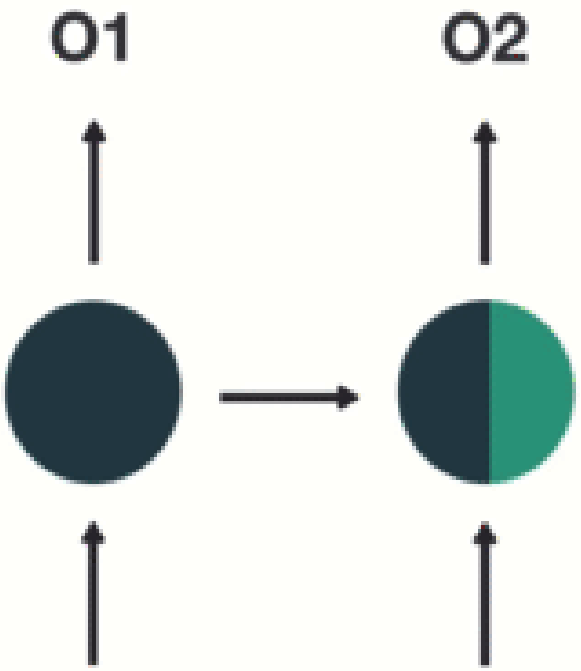
Source: https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

What time is it?

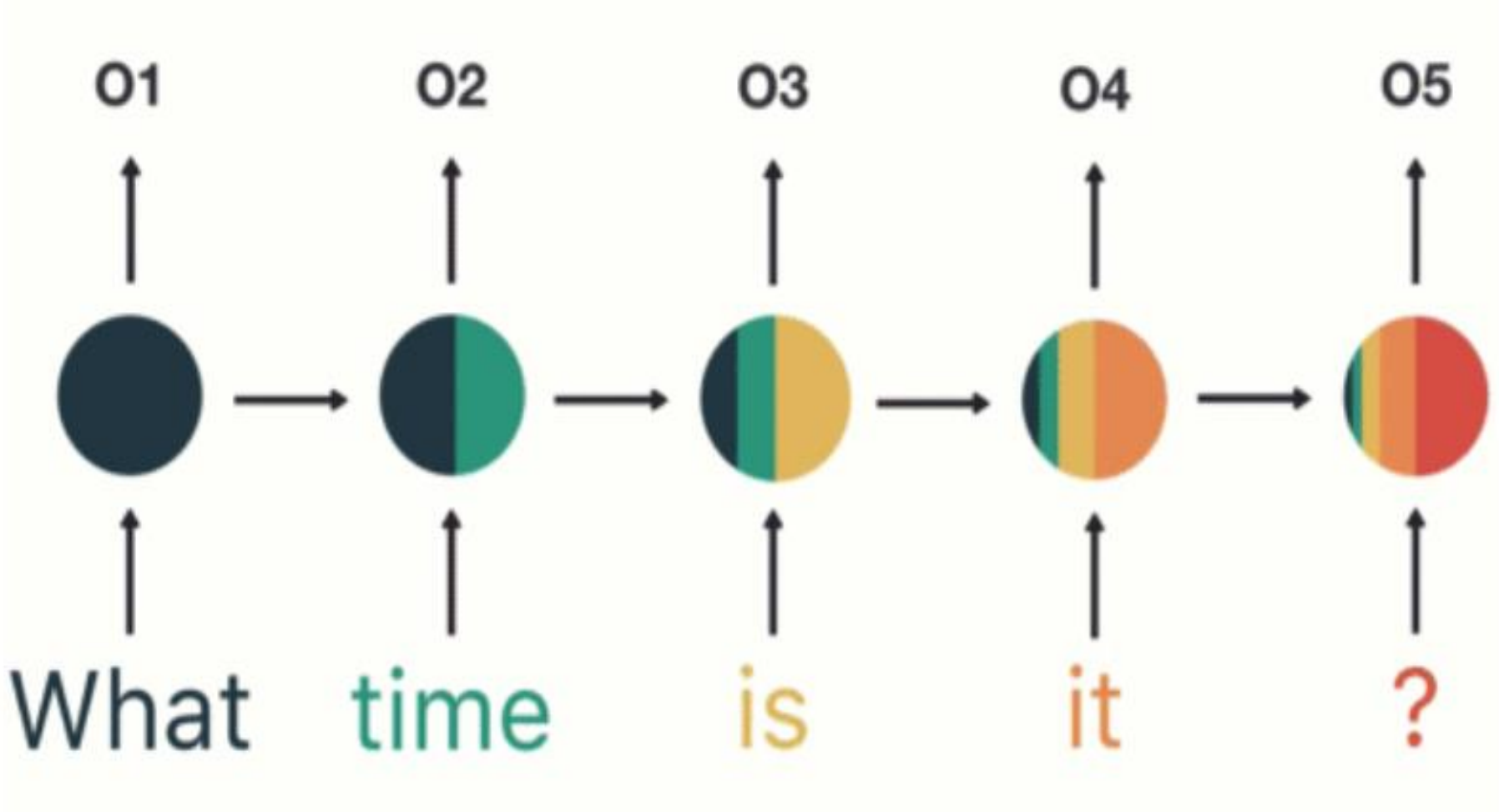

What time is it ?



What time is it ?



What time is it ?


O1
O2
O3
O4
O5
What
time
is
it
?

Sometimes we don’t need our network to learn only from immediate past information.

Example: Suppose we want to predict the blank word in the text ‘ David, a 36-year old man lives in San Francisco. He has a female friend Maria. Maria works as a cook in a famous restaurant in New York whom he met recently in a school alumni meet. Maria told him that she always had a passion for _____ .

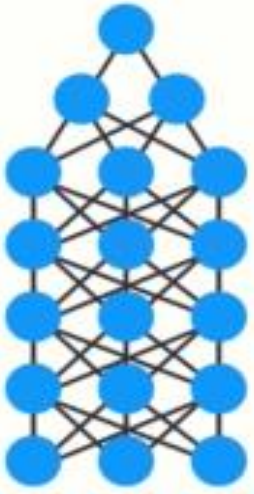
Here, we want our network to learn from dependency ‘cook’ to predict ‘cooking.
There is a gap between the information what we want to predict and from where we want it to get predicted . This is called long-term dependency.

Unfortunately, RNN does not work practically in this situation.

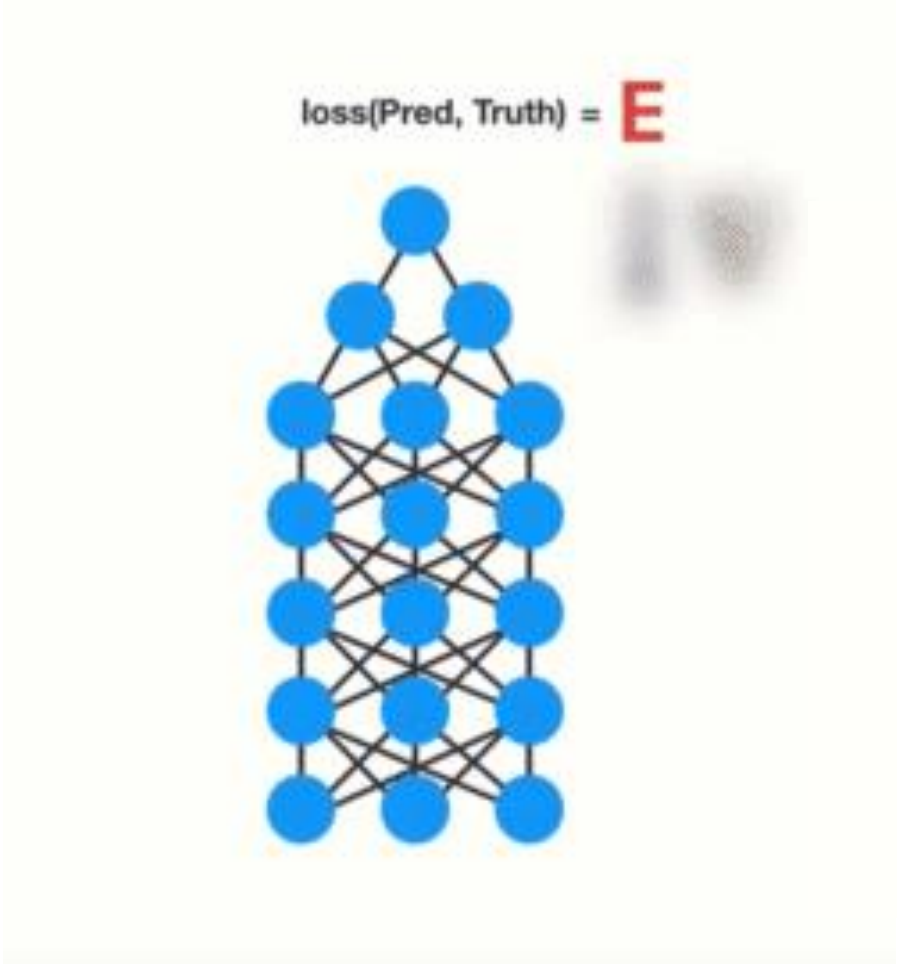
Short-Term memory and the vanishing gradient is due to the nature of back-propagation; an algorithm used to train and optimize neural networks.

Struggles with Long-Term Dependancy.

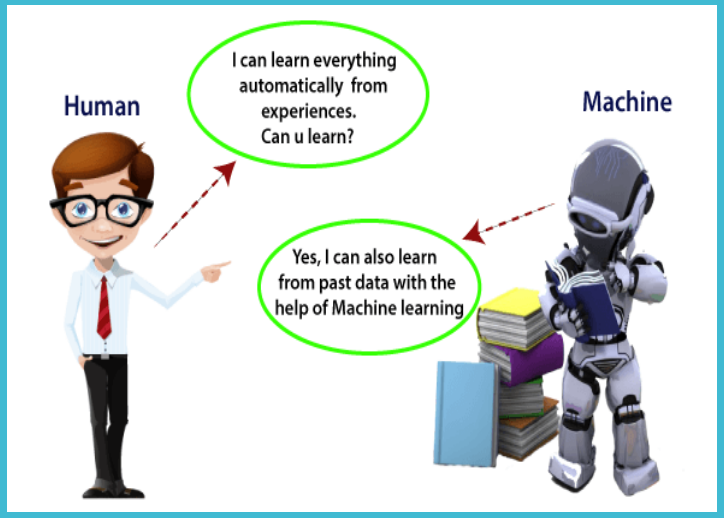
Training a neural network has three major steps. First, it does a forward pass and makes a prediction. Second, it compares the prediction to the ground truth using a loss function. The loss function outputs an error value which is an estimate of how poorly the network is performing. Last, it uses that error value to do back propagation which calculates the gradients for each node in the network.



But in RNN it observes the phenomenon of Vanishing Gradient!!



Long Short Term Memory (LSTM)



Marwadi University

Department of Information and Communication Technology

Unit 7:
Deep Learning

Artificial Intelligence (01CT0703)

Prof. Nishith Kotak

INTUITION

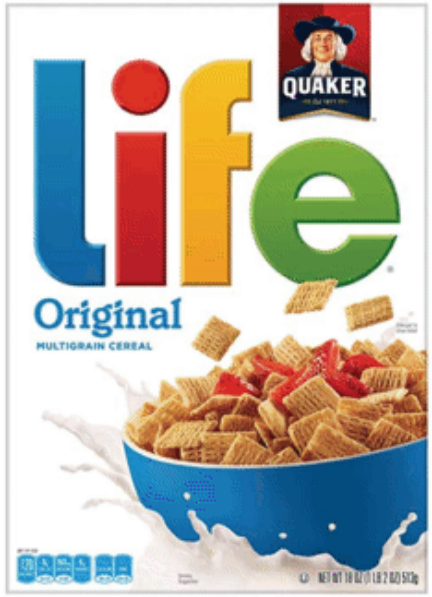
Customers Review 2,491

Thanos

September 2018

Verified Purchase

Amazing! This box of cereal gave me a perfectly balanced breakfast, as all things should be. I only ate half of it but will definitely be buying again!



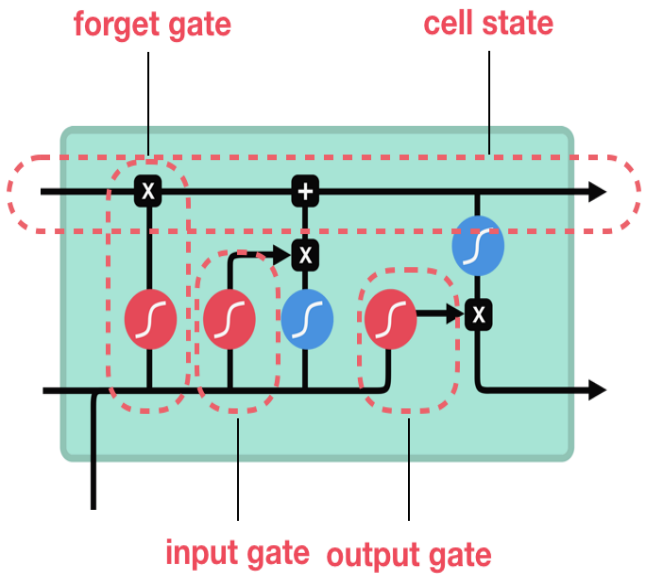
A Box of Cereal

$3.99

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

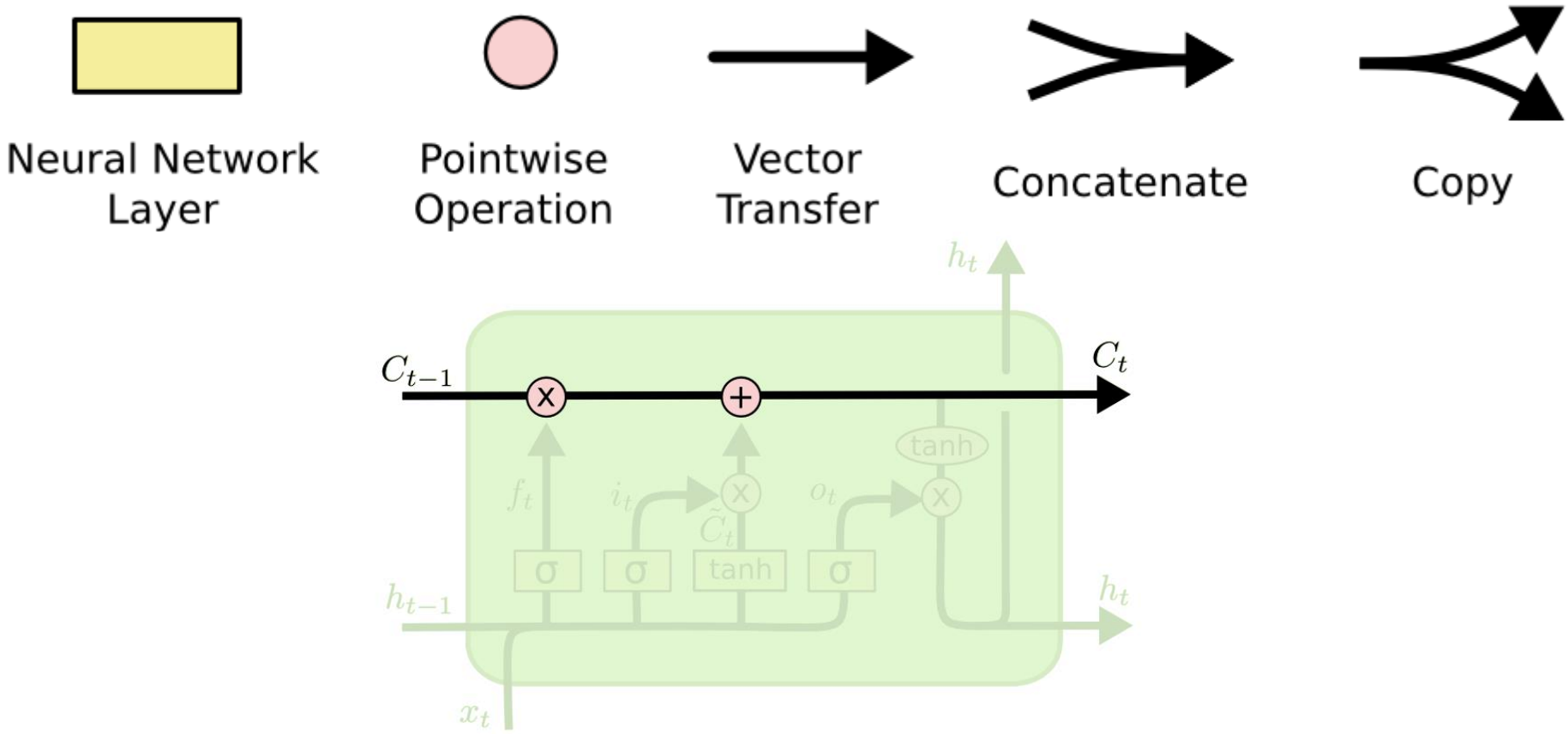
Long Short Term Memory

- For a long sequence of data, RNN stops learning while backpropagation due to vanishing gradient phenomenon
- LSTM have gates that regulates the flow of information
- The gates are different neural networks that decide which information is allowed on the cell state.
- These gates can learn which data in a sequence is important to keep or throw away in the long chain of sequences to make predictions



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

Long Short Term Memory

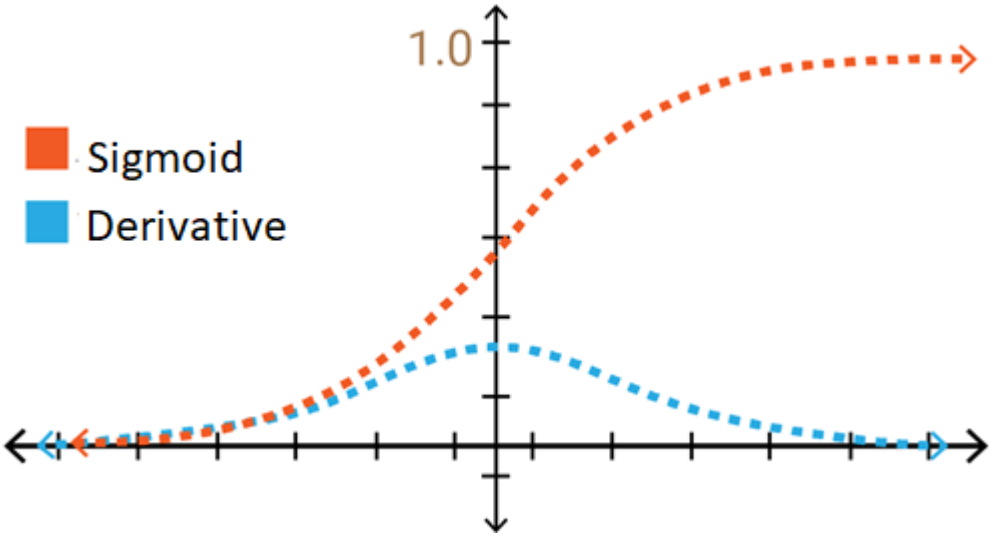


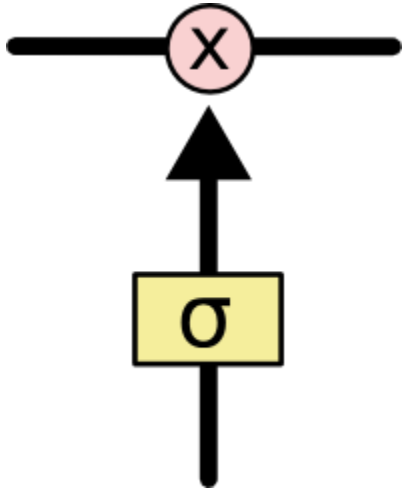
Source: https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

Long Short Term Memory

- The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates.

- Gates are a way to optionally let information through. They are composed out of a sigmoid neural net layer and a pointwise multiplication operation.





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

Long Short Term Memory

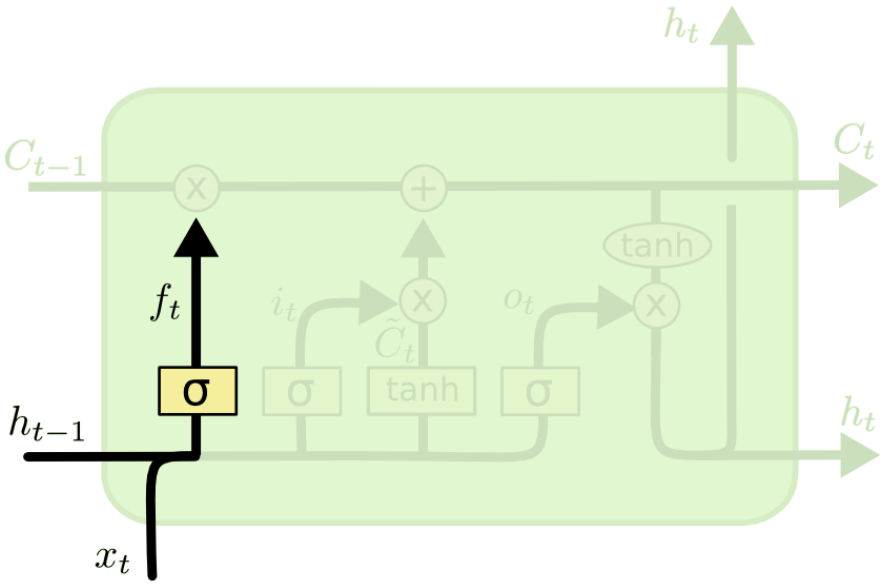
LSTM has three gates to protect and control the flow of the information

- The Forget gate decides what is relevant to keep from prior steps.
- The input gate decides what information is relevant to add from the current step.
- The output gate determines what the next hidden state should be.

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

Long Short Term Memory

The first step in our LSTM is to decide what information we're going to throw away from the cell state. This decision is made by a sigmoid layer called the "forget gate layer."



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

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

Long Short Term Memory

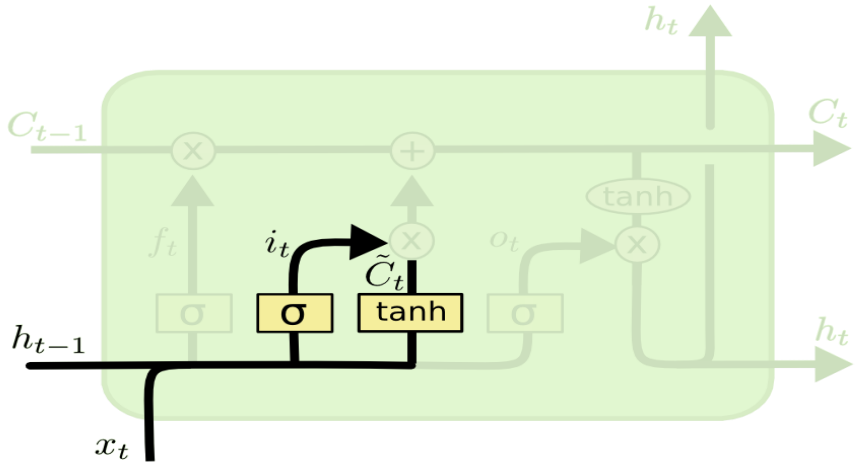
- The next step is to decide what new information we're going to store in the cell state.

- **TanH Function:**

To overcome the vanishing gradient problem, we need a function whose second derivative can sustain for a long range before going to zero. *tanh* is a suitable function with the above property.

- **Sigmoid Function:**

As Sigmoid can output 0 or 1, it can be used to forget or remember the information.



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

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

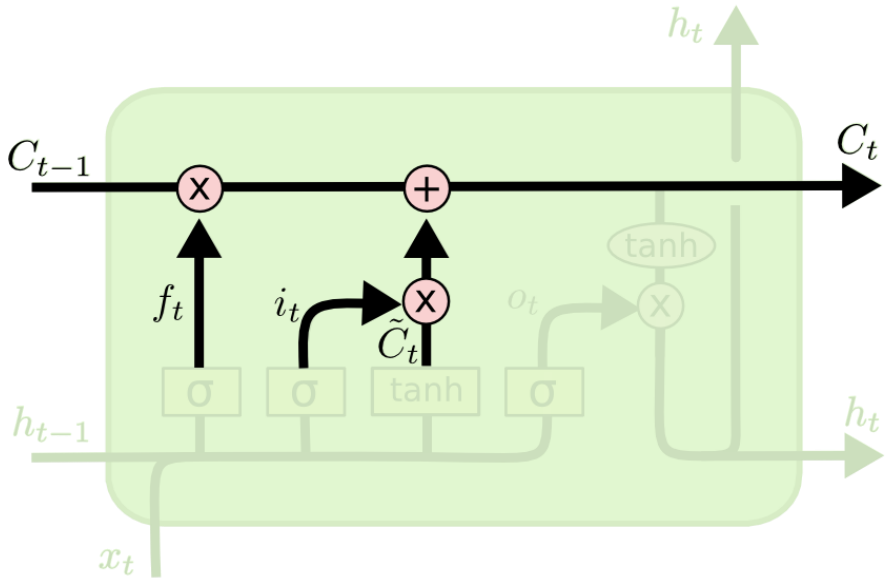
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Long Short Term Memory

It's now time to update the old cell state into the new cell state.

We multiply the old state by ft, forgetting the things we decided to forget earlier.

Then we add it*Ct. This is the new candidate values, scaled by how much we decided to update each state value.

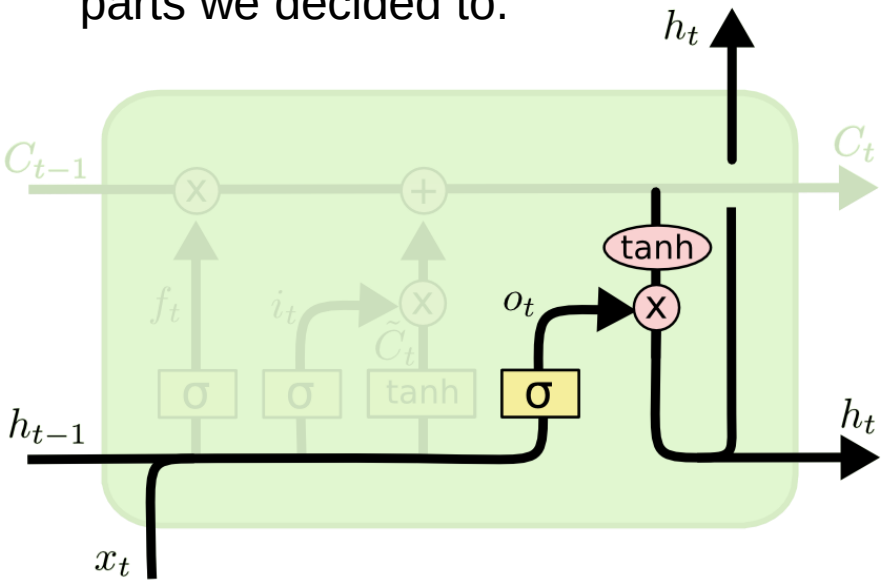


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

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

Long Short Term Memory

- Finally, we need to decide what we're going to output. This output will be based on our cell state, but will be a filtered version.
- First, we run a sigmoid layer which decides what parts of the cell state we're going to output.
- Then, we put the cell state through tanh (to push the values to be between −1 and 1) and multiply it by the output of the sigmoid gate, so that we only output the parts we decided to.



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

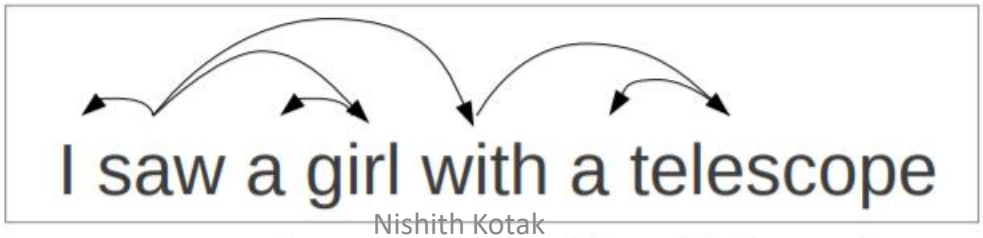
$$h_t = o_t * \tanh\left(C_t\right)$$

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

LSTM can be understood as T-Shaped Valve!!

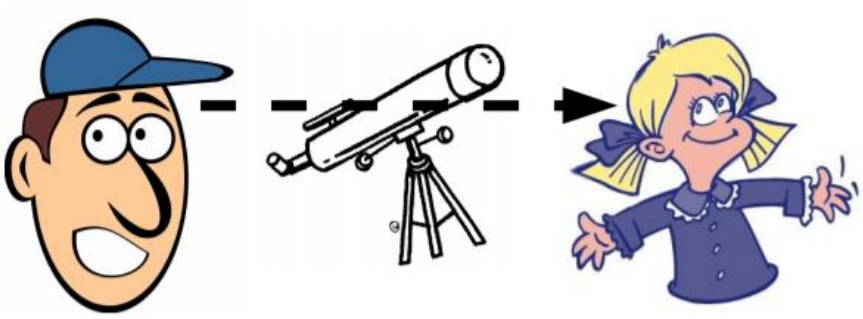
Dependancy Parsing

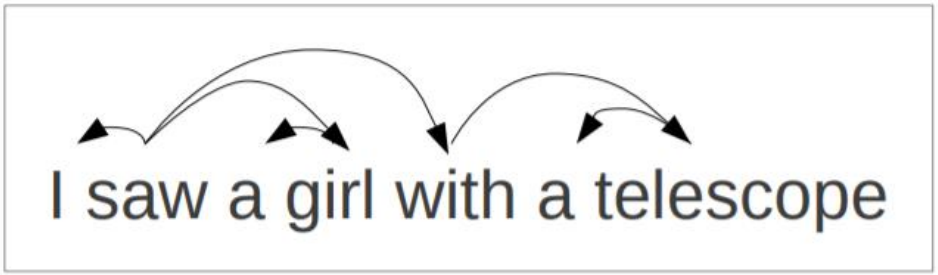
- Words don't stand by their own but they are connected with each other by some hidden structures.
- It is the task of **extracting a dependency parse** of a sentence that represents its grammatical structure and defines the relationships between "**head**" words and words, which modify those heads i.e. **dependents**.
- Each word is connected with other word by a direct link called dependencies.

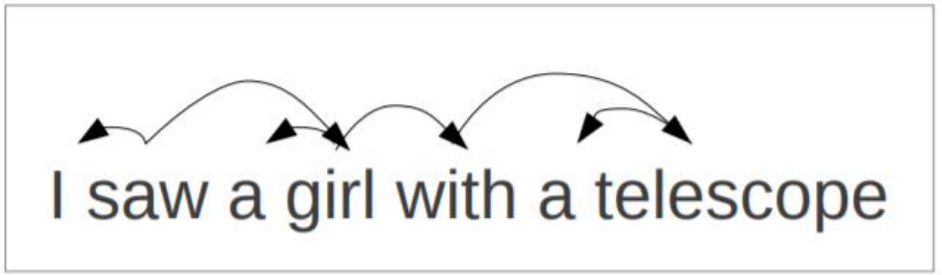


Dependency Parser

Dependencies also resolves the ambiguities







Word Embeddings

- Word2Vec (One Hot Encoding)
- CBOW
- Skip N-gram