RNN TUTORIAL

(RECURRENT NEURAL NETWORK)

simplilearn

What's in it for you?

- What is a Neural Network?
- Popular Neural Networks
- Why Recurrent Neural Network?
- What is a Recurrent Neural Network?
- How does a RNN work?
- Vanishing and Exploding Gradient Problem
- Long Short Term Memory (LSTM)
- Use case implementation of LSTM

simplilearn

Introduction to RNN

Do you know how Google's autocomplete feature predicts the rest of the words a user is typing?

Introduction to RNN

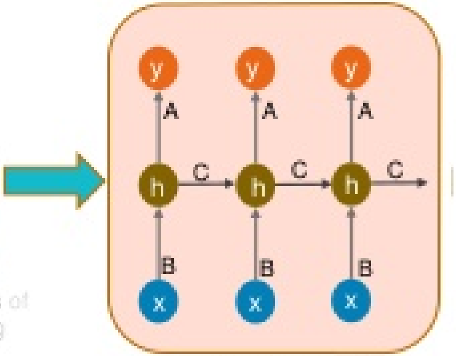
Do you know how Google's autocomplete feature predicts the rest of the words a user is typing?

Collection of large volumes of most frequently occurring consecutive words

Introduction to RNN

Do you know how Google's autocomplete feature predicts the rest of the words a user is typing?

Collection of large volumes of most frequently occurring consecutive words

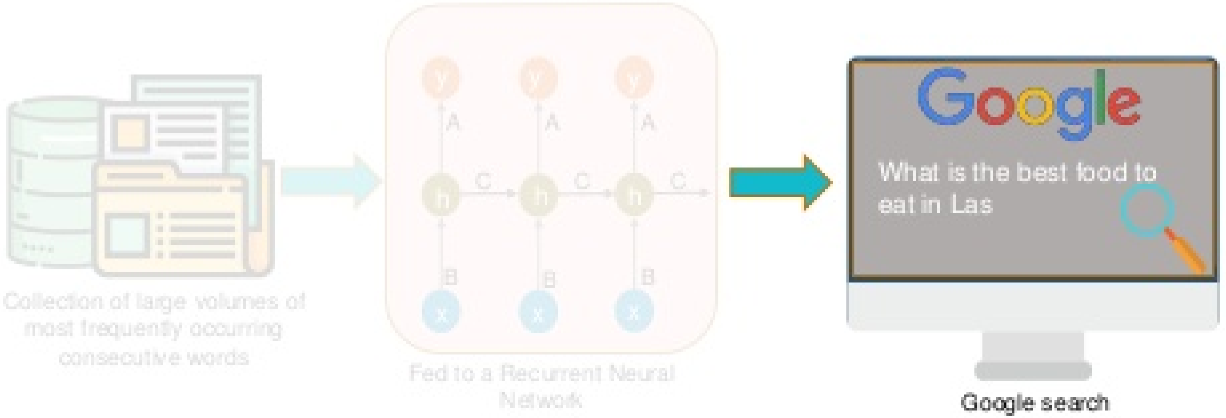


Fed to a Recurrent Neural Network

Analyses the data by finding the sequence of words occurring frequently and builds a model to predict the next word in the sentence

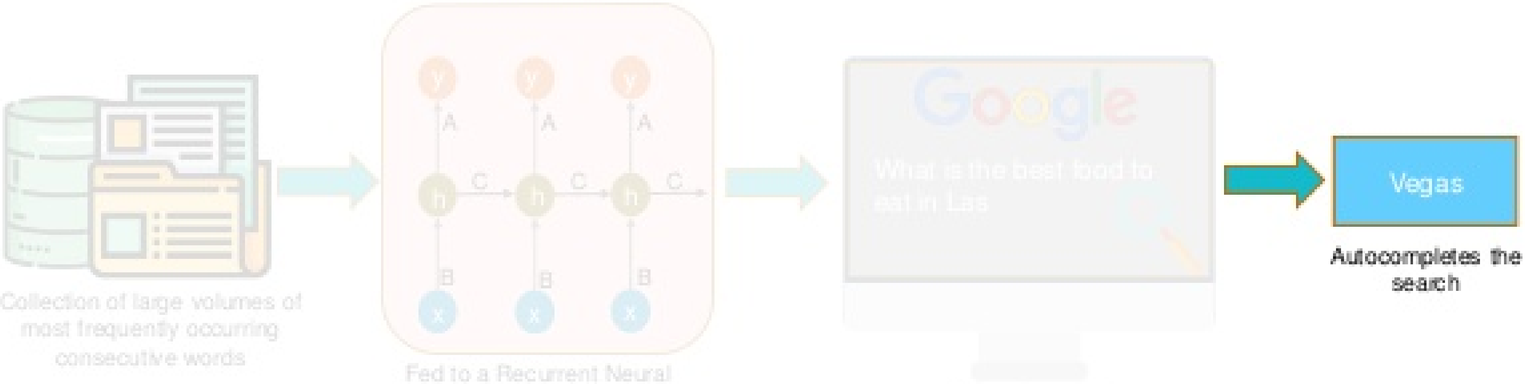
Introduction to RNN

Do you know how Google's autocomplete feature predicts the rest of the words a user is typing?



Introduction to RNN

Do you know how Google's autocomplete feature predicts the rest of the words a user is typing?



Collection of large volumes of most frequently occurring consecutive words

Fed to a Recurrent Neural Network

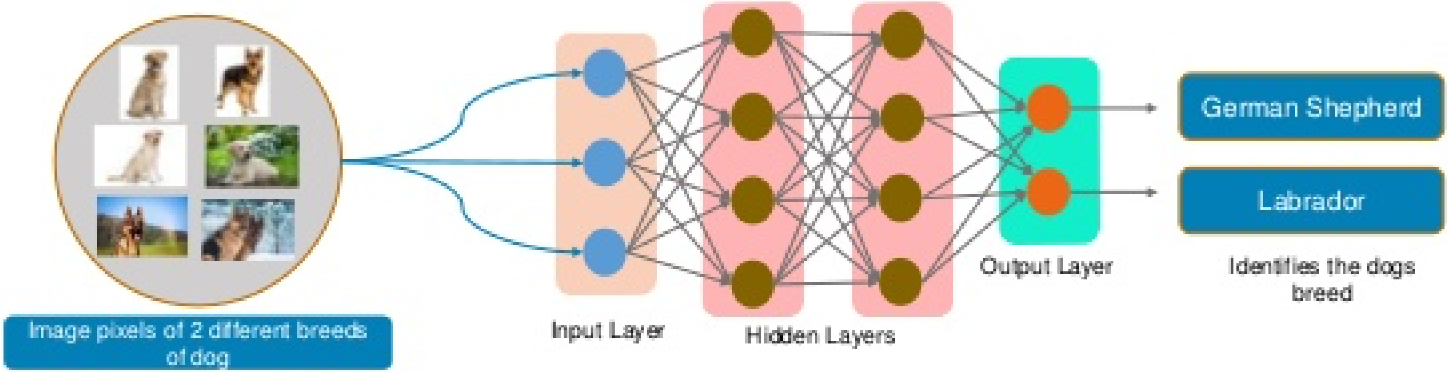
Google search

Autocompletes the search

simplilearn

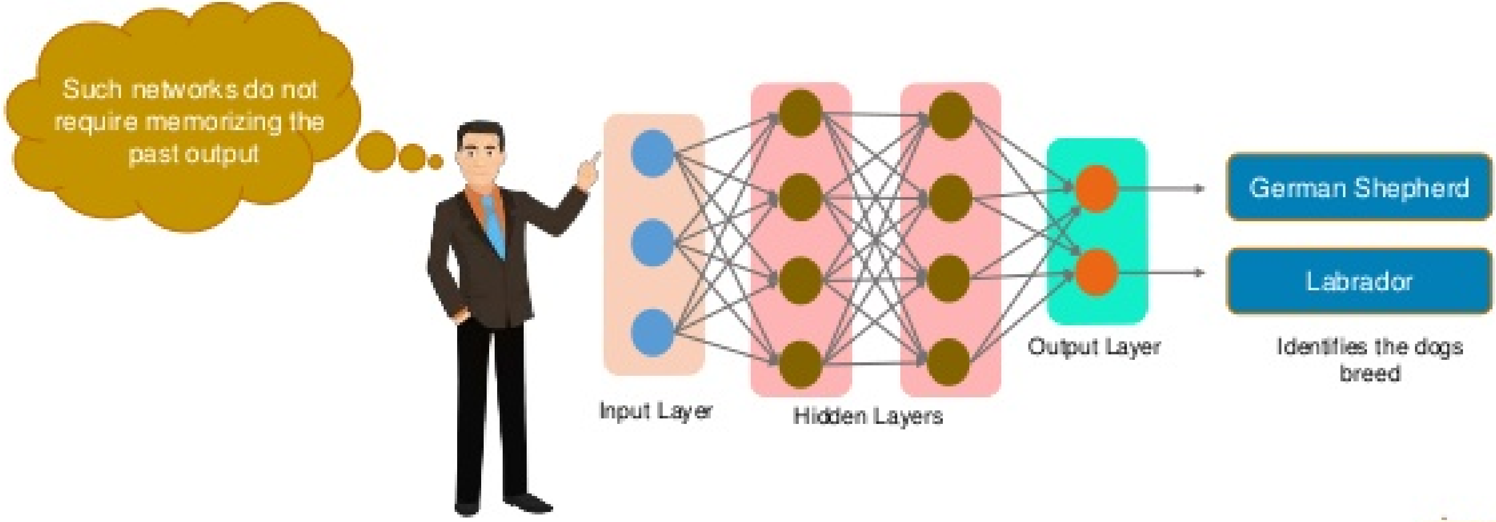
What is a Neural Network?

Neural Networks used in Deep Learning, consists of different layers connected to each other and work on the structure and functions of a human brain. It learns from huge volumes of data and uses complex algorithms to train a neural net.



What is a Neural Network?

Neural Networks used in Deep Learning, consists of different layers connected to each other and work on the structure and functions of a human brain. It learns from huge volumes of data and uses complex algorithms to train a neural net.

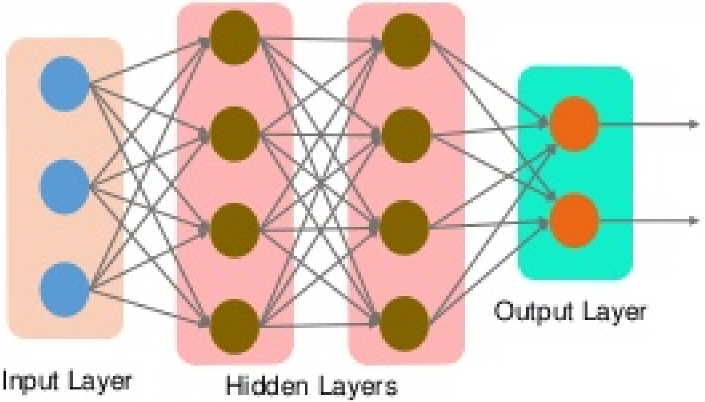


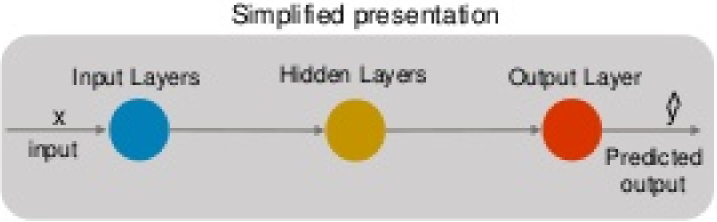
Popular Neural Networks

Feed Forward Neural Network	Used in general Regression and Classification problems
Convolution Neural Network	Used for Image Recognition
Deep Neural Network	Used for Acoustic Modeling
Deep Belief Network	Used for Cancer Detection
Recurrent Neural Network	Used for Speech Recognition

Feed Forward Neural Network

In a Feed-Forward Network , information flows only in forward direction, from the input nodes, through the hidden layers (if any) and to the output nodes. There are no cycles or loops in the network.

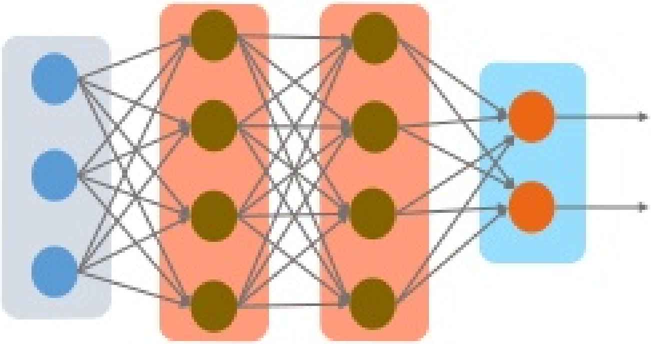




- Decisions are based on current input
- No memory about the past
- No future scope

Why Recurrent Neural Network?

Issues in Feed Forward Neural Network



Feed Forward Neural Network

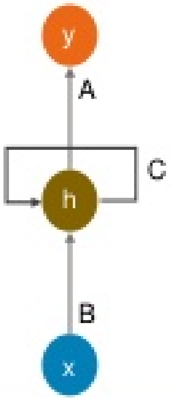
01 cannot handle sequential data

02 considers only the current input

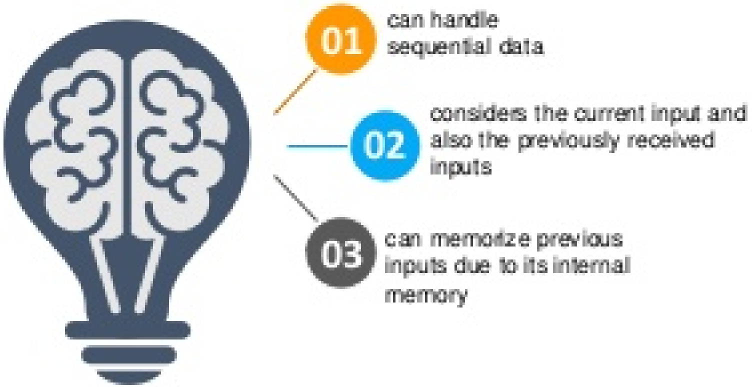
03 cannot memorize previous inputs

Why Recurrent Neural Network?

Solution to Feed Forward Neural Network



Recurrent Neural Network



Applications of RNN

"A Dog catching a ball in mid air"

Image captioning

RNN is used to caption an image by analyzing the activities present in it

Applications of RNN

Time series prediction

Any time series problem like predicting the prices of stocks in a particular month can be solved using RNN

Applications of RNN

Natural Language Processing

When it rains, look for rainbows. When it's dark, look for stars.

Positive Sentiment

Text mining and Sentiment analysis can be carried out using RNN for Natural Language Processing

Applications of RNN

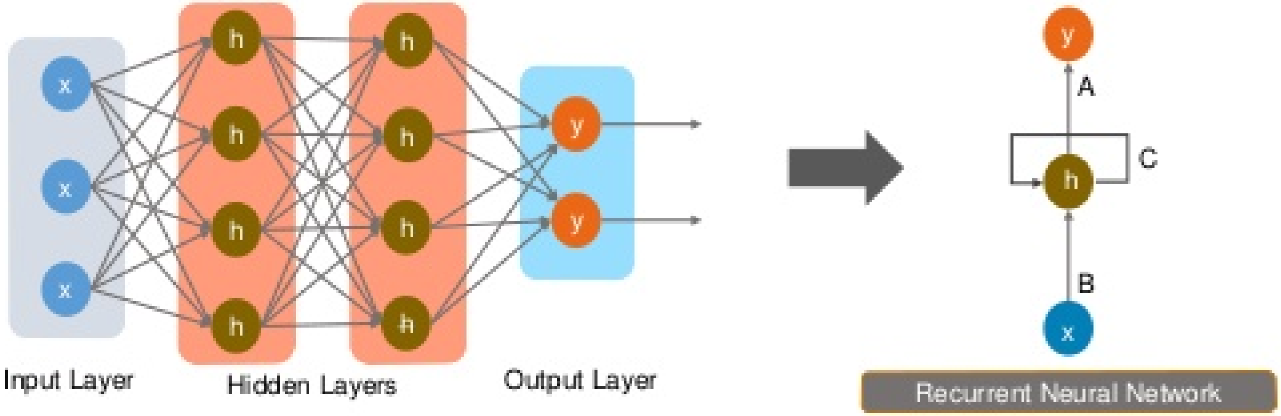
Machine Translation

Here the person is speaking in English and it is getting translated into Chinese, Italian, French, German and Spanish languages

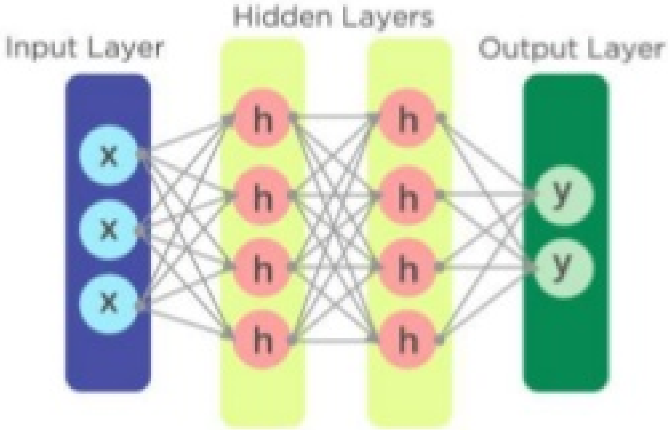
Given an input in one language, RNN can be used to translate the input into different languages as output

What is a Recurrent Neural Network?

Recurrent Neural Network works on the principle of saving the output of a layer and feeding this back to the input in order to predict the output of the layer.

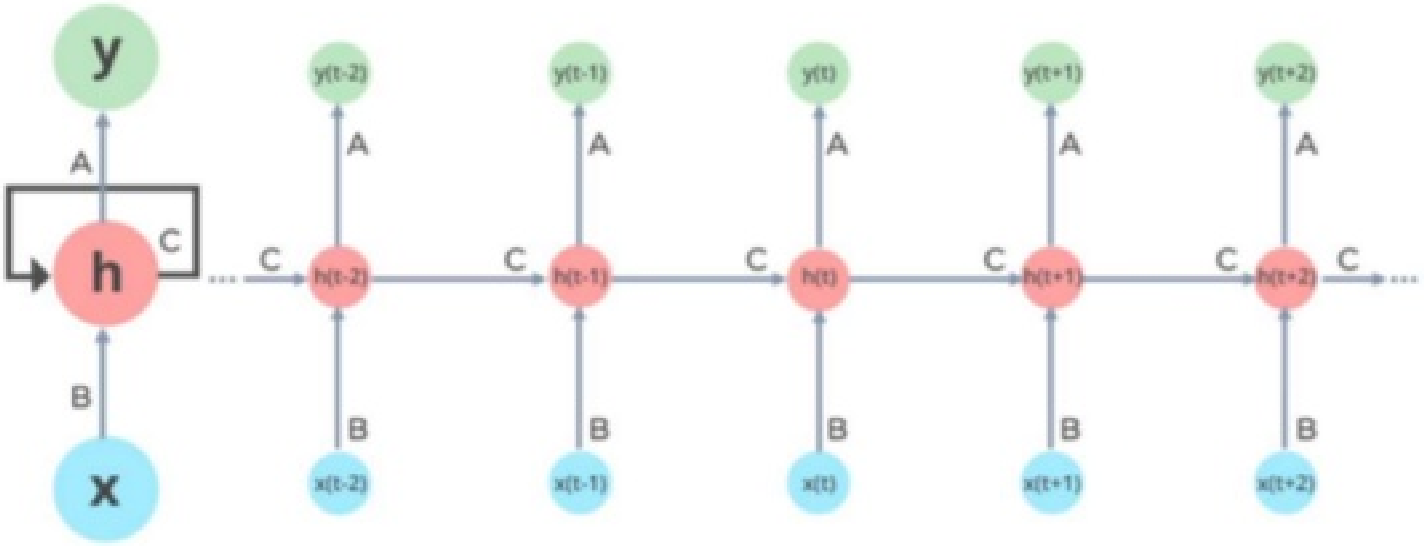


How does a RNN look like?

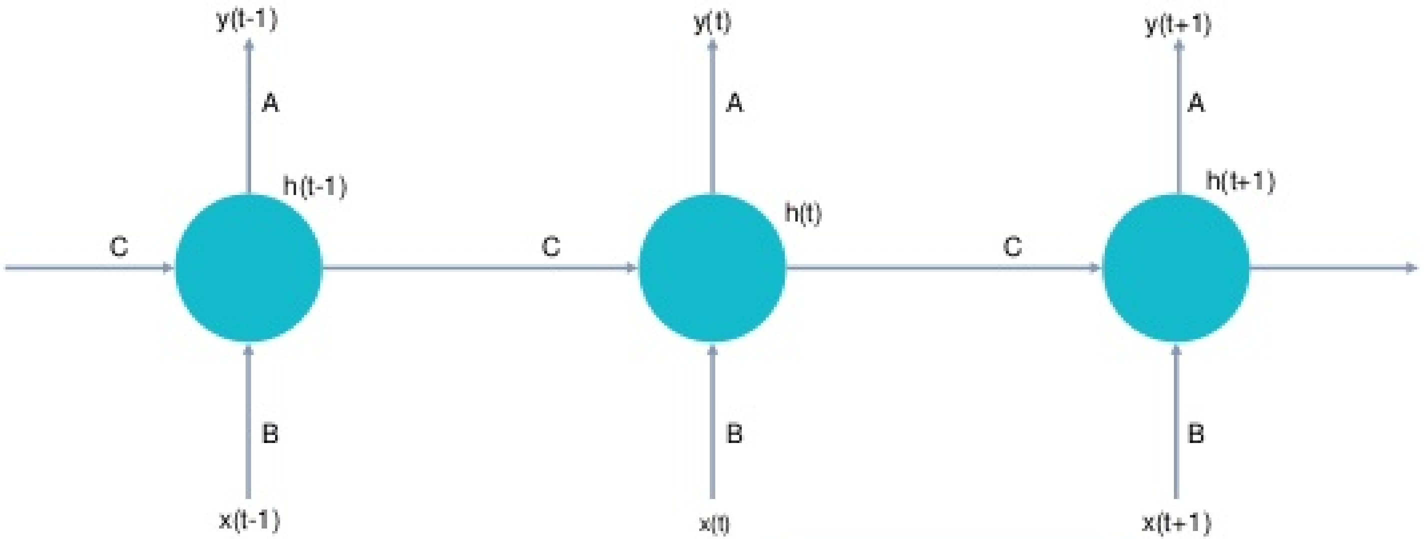


A, B and C are the parameters

How does a RNN look like?



How does a RNN work?

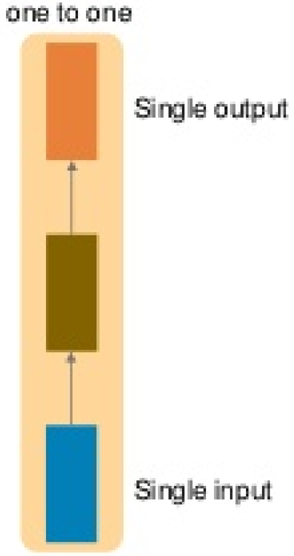


$$h(t) = f_c\,(h(t\text{-}1)\ ,x(t))$$

$h(t)$ = new state
f_c = function with parameter c
$h(t-1)$ = old state
$x(t)$ = input vector at time step t

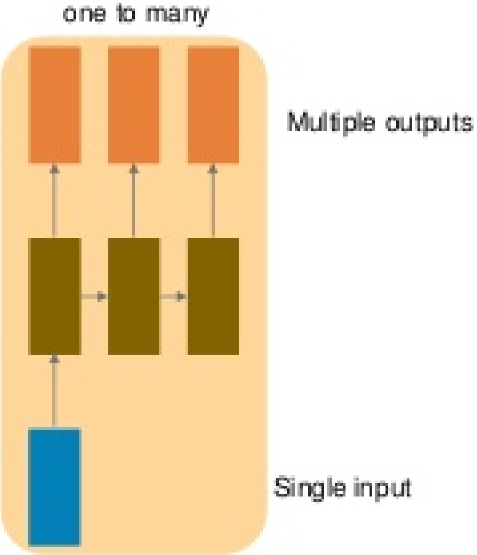
simplilearn

Types of Recurrent Neural Network



one to one network is known as the Vanilla Neural Network. Used for regular machine learning problems

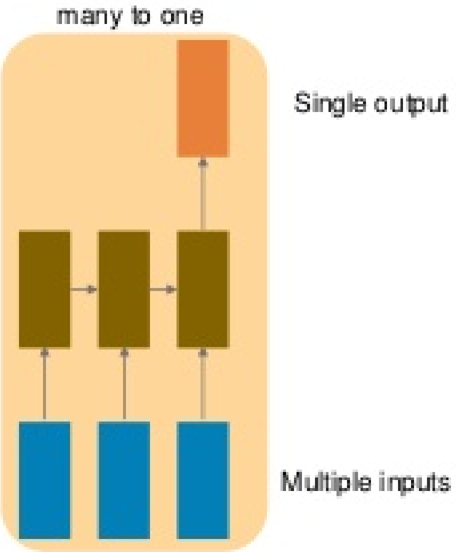
Types of Recurrent Neural Network



one to many network generates sequence of outputs. Example: Image captioning

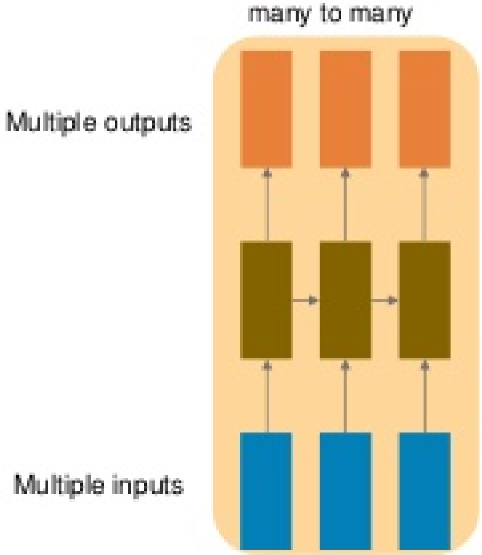
Types of Recurrent Neural Network

many to one network takes in a sequence of inputs. Example: Sentiment analysis where a given sentence can be classified as expressing positive or negative sentiments

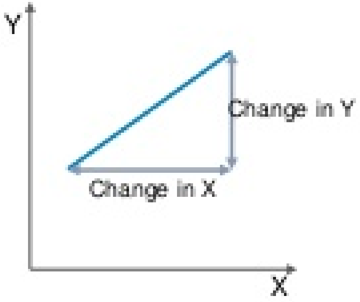


Types of Recurrent Neural Network

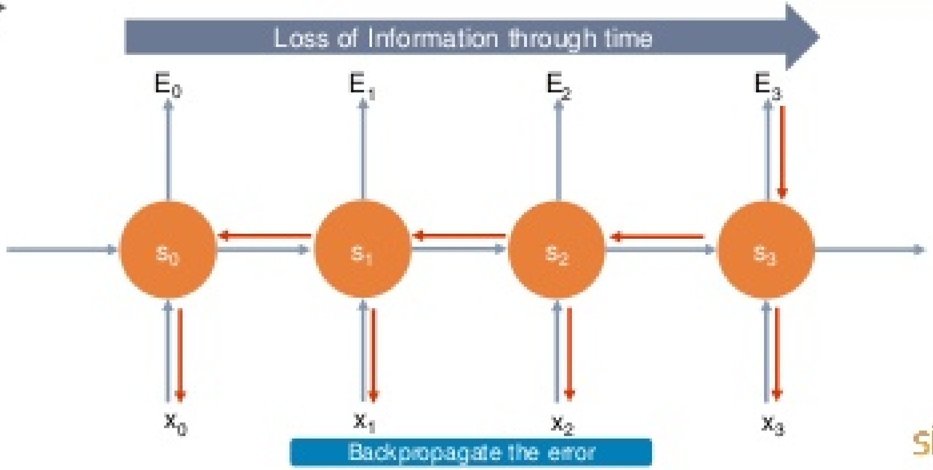
many to many network takes in a sequence of inputs and generates a sequence of outputs.
Example: Machine Translation



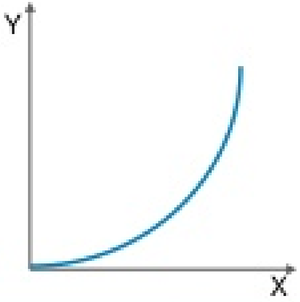
Vanishing Gradient Problem



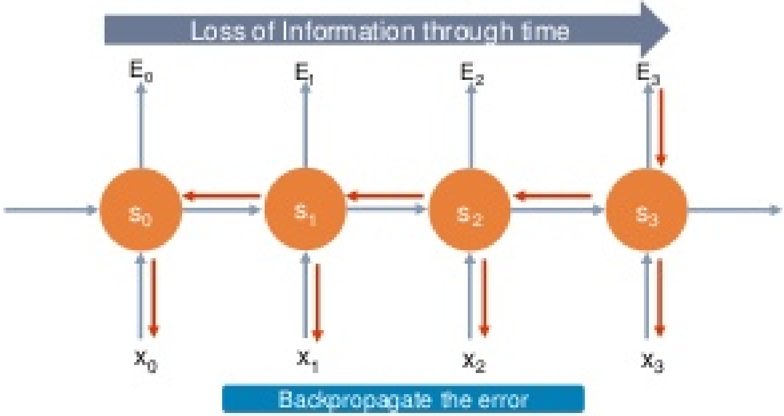
While training a RNN, your slope can be either too small or very large and this makes training difficult. When the slope is too small, the problem is know as ***Vanishing gradient.***



Exploding Gradient Problem



When the slope tends to grow exponentially instead of decaying, this problem is called ***Exploding gradient***.



Issues in Gradient Problem

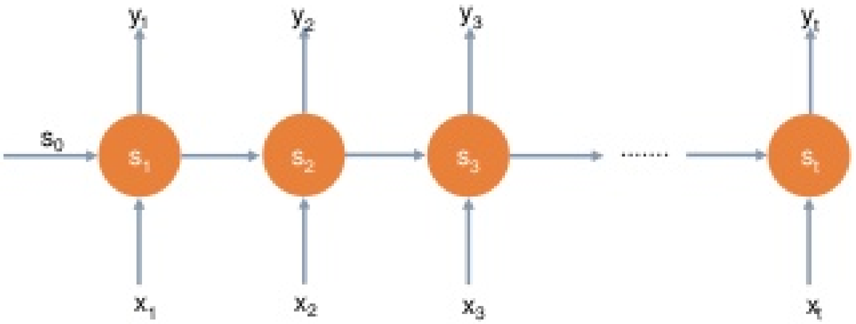
- Long training time
- Poor performance
- Bad accuracy

Explaining Gradient Problem

Consider the following 2 examples to understand what should be the next word in the sequence:

The person who took my bike and...was. a thief.

The students who got into Engineering with ....were from Asia.



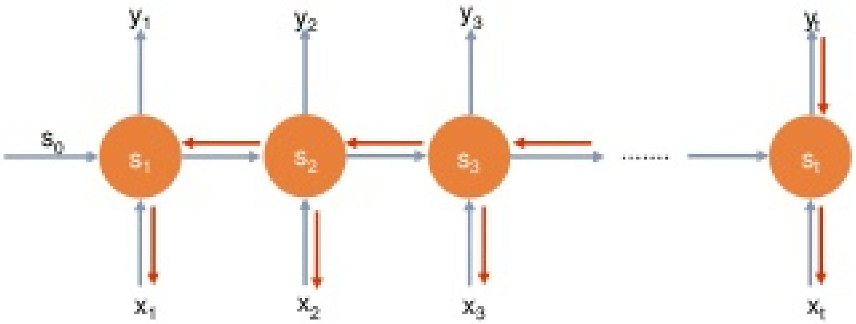
Explaining Gradient Problem

Consider the following 2 examples:

The person who took my bike and……………………………………………was. a thief.

The students who got into Engineering with ……………………………………, were from Asia .

In order to understand what would be the next word in the sequence, the RNN must memorize the previous context whether the subject was singular noun or plural noun



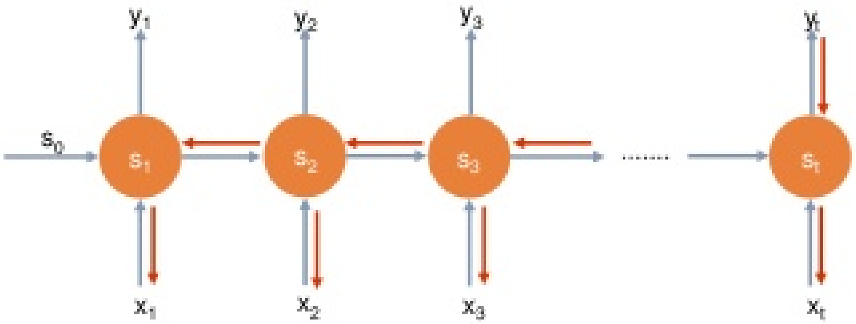
Explaining Gradient Problem

Consider the following 2 examples:

The person who took my bike and………………………………………………, was a thief.

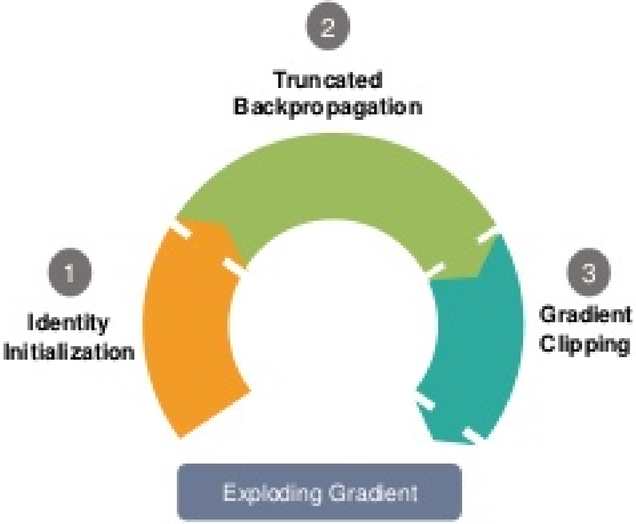
The students who got into Engineering with …………………………………….……were from Asia .

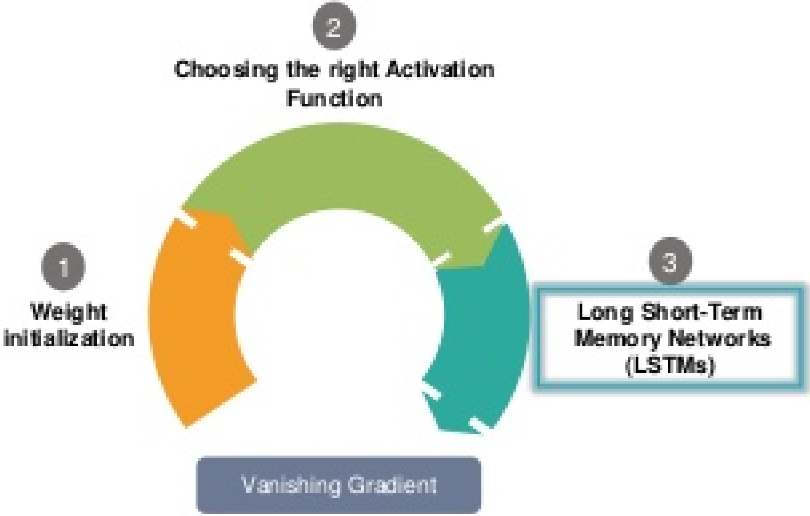
In order to understand what would be the next word in the sequence, the RNN must memorize the previous context whether the subject was singular noun or plural noun



It might be sometimes difficult for the error to backpropagate to the beginning of the sequence to predict what should be the output

Solution to Gradient Problem





Long-Term Dependencies

Suppose we try to predict the last word in the text

" The clouds are in the _sky"

Here we do not need any further context. It's pretty clear the last word is going to be "*sky*".

Long-Term Dependencies

Suppose we try to predict the last word in the text

"I have been staying in Spain for the last 10 years… I can speak fluent ______ *Spanish*

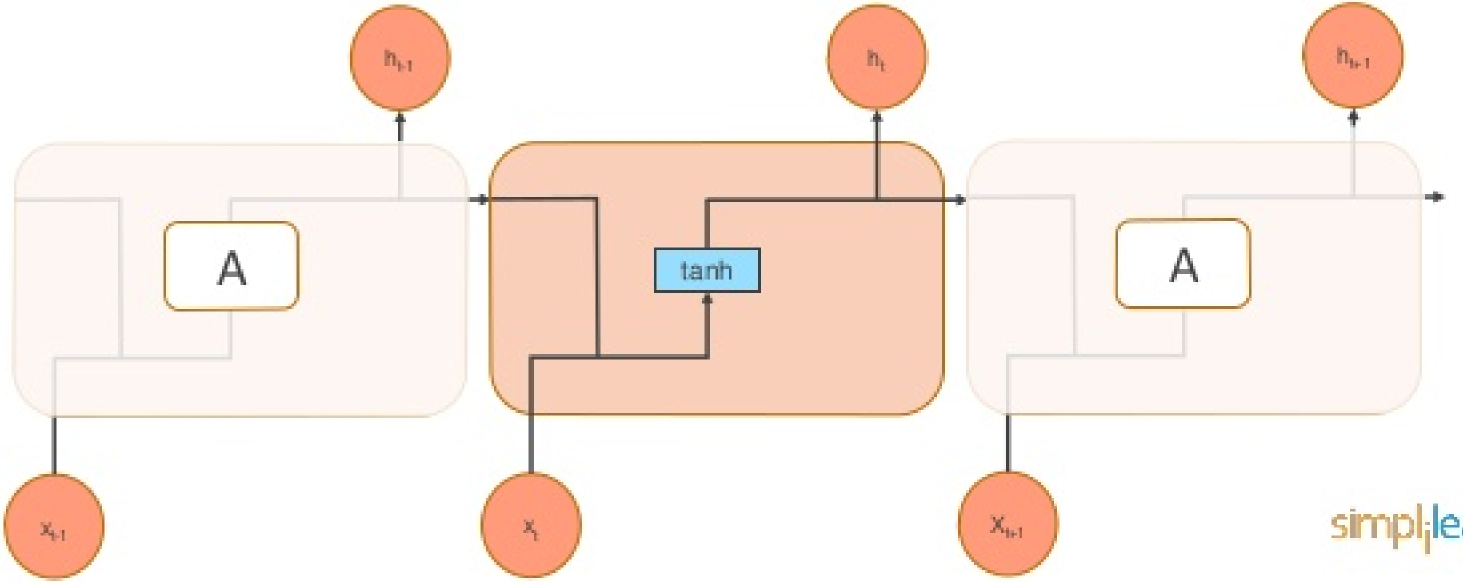
The word we predict will depend on the previous few words in context

- Here we need the context of **Spain** to predict the last word in the text.
- It's possible that the gap between the relevant information and the point where it is needed to become very large.
- **LSTMs** help us solve this problem.

Long Short-Term Memory Networks

LSTMs are special kind of Recurrent Neural Networks, capable of learning long-term dependencies. Remembering information for long periods of time is their default behavior.

All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.

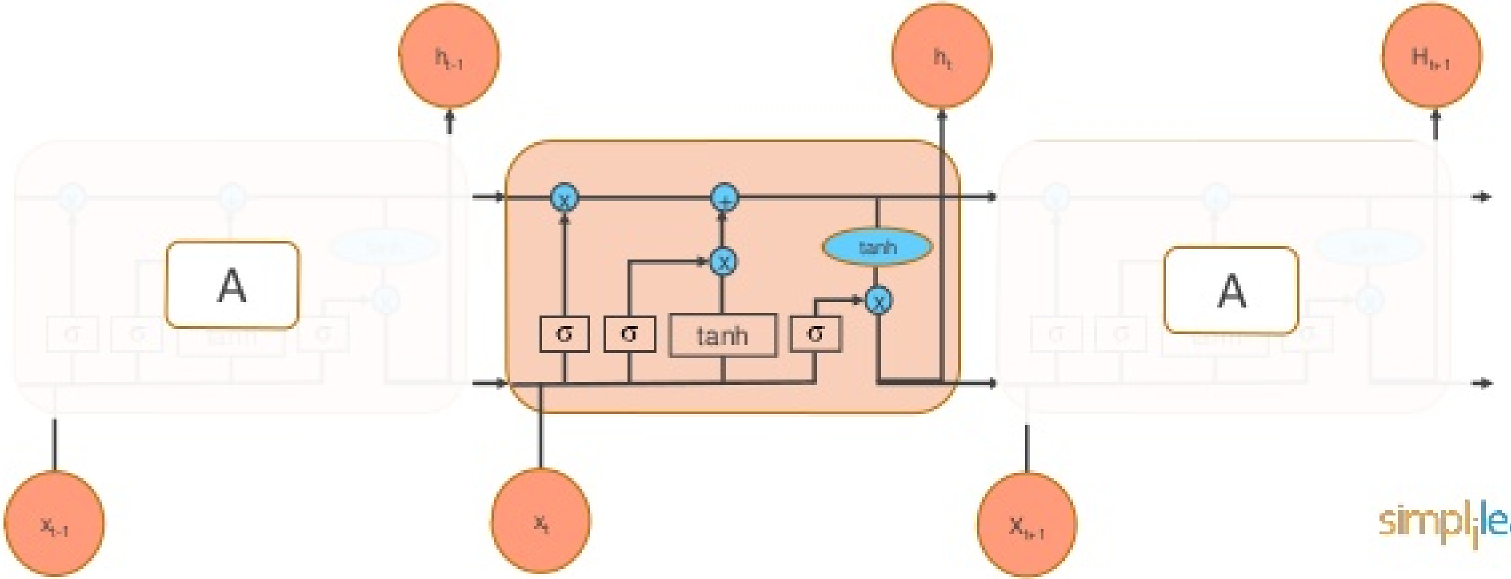


simplilearn

Long Short-Term Memory Networks

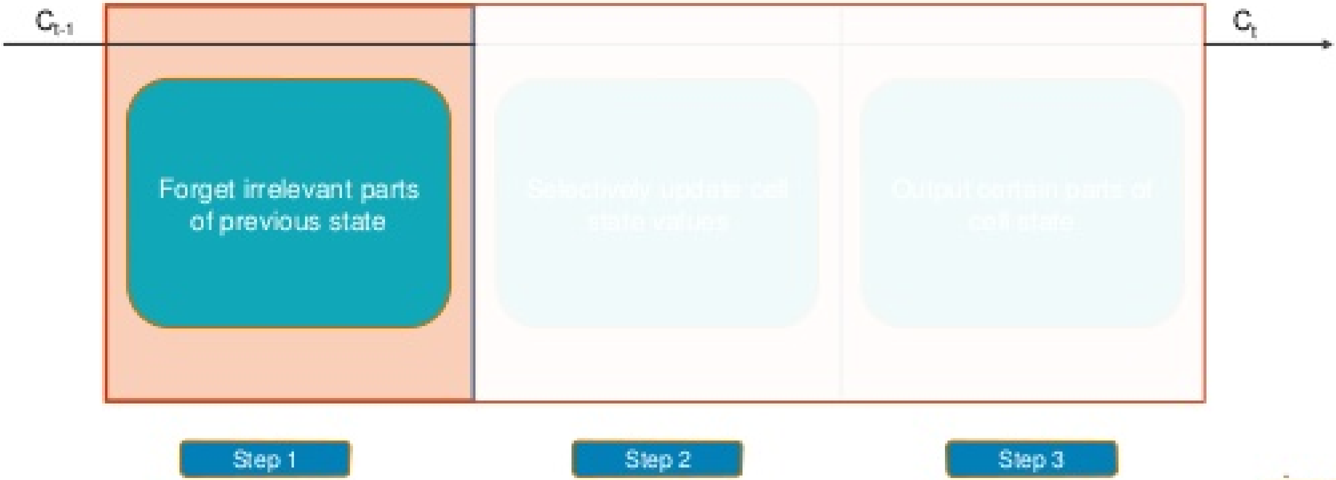
LSTMs are special kind of Recurrent Neural Networks, capable of learning long-term dependencies. Remembering information for long periods of time is their default behavior.

LSTMs also have a chain like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four interacting layers communicating in a very special way.



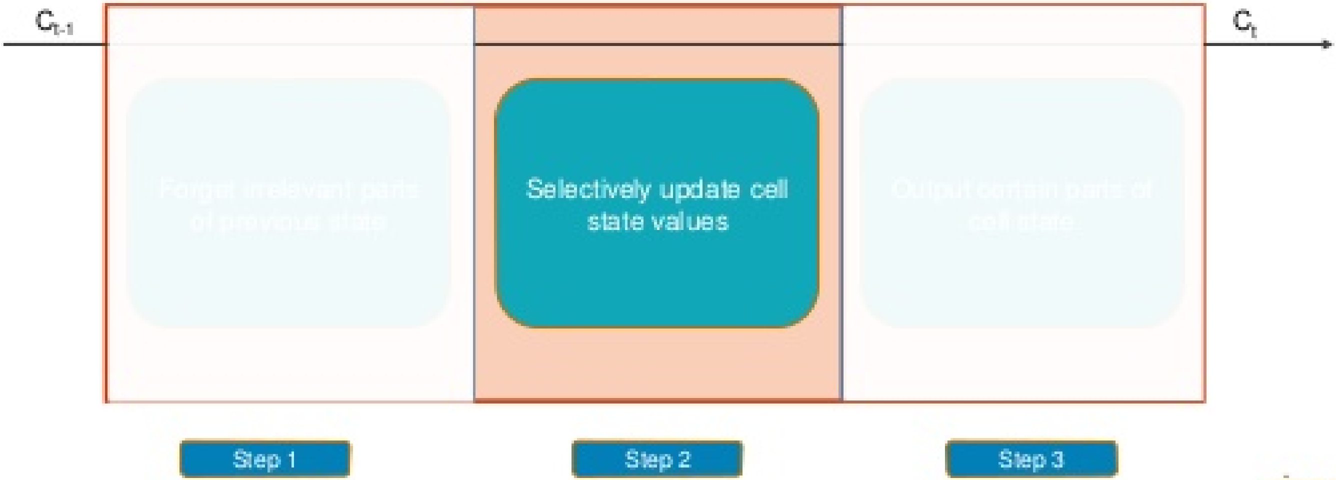
Long Short-Term Memory Networks

3 step process of LSTMs



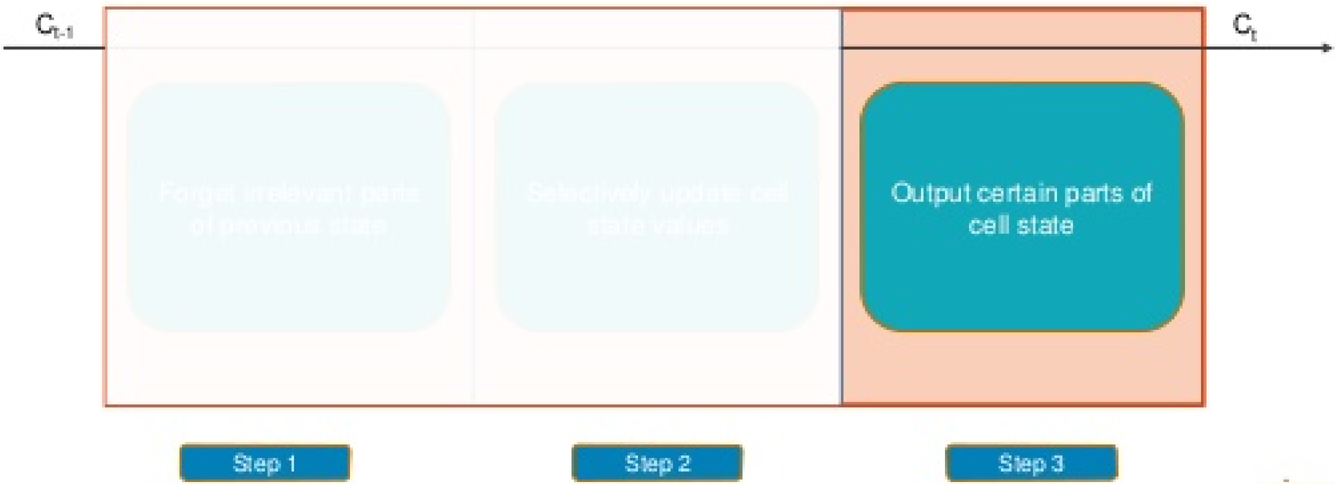
Long Short-Term Memory Networks

3 step process of LSTMs



Long Short-Term Memory Networks

3 step process of LSTMs



simplilearn

Working of LSTMs

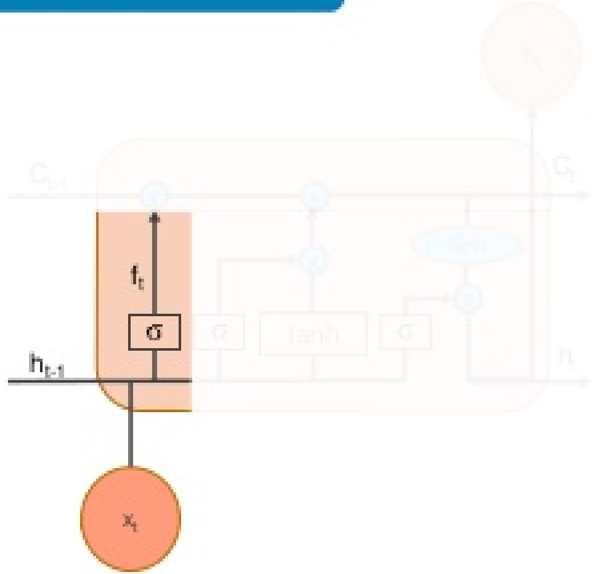
Step-1

Decides how much of the past it should remember

First step in the LSTM is to decide which information to be omitted in from the cell in that particular time step. It is decided by the sigmoid function. It looks at the previous state (h_{t-1}) and the current input x_t and computes the function.

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

f_t = forget gate
Decides which information to delete that is not important from previous time step

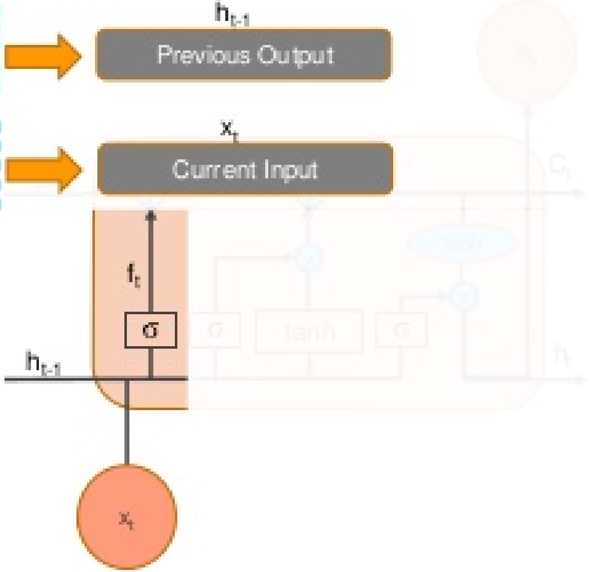


Working of LSTMs

Consider an LSTM is fed with the following inputs from previous and present time step :

Alice is good in Physics. John on the other hand is good in Chemistry.

John plays football well. He told me yesterday over phone that he had served as the captain of his college football team.

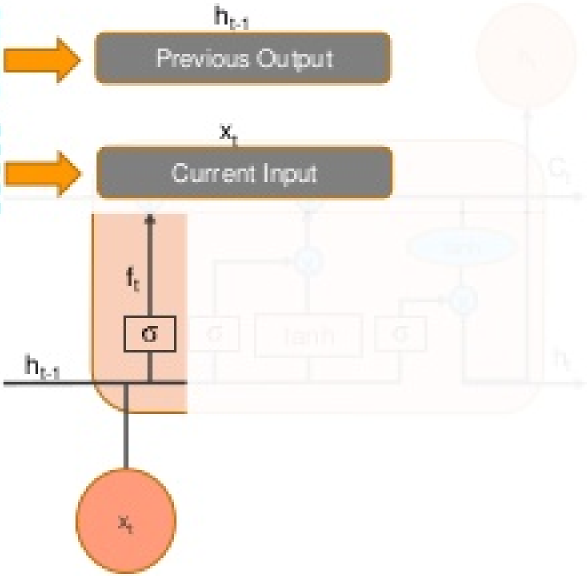


Working of LSTMs

Consider an LSTM is fed with the following inputs from previous and present time step :

Alice is good in Physics. John on the other hand is good in Chemistry.

John plays football well. He told me yesterday over phone that he had served as the captain of his college football team.



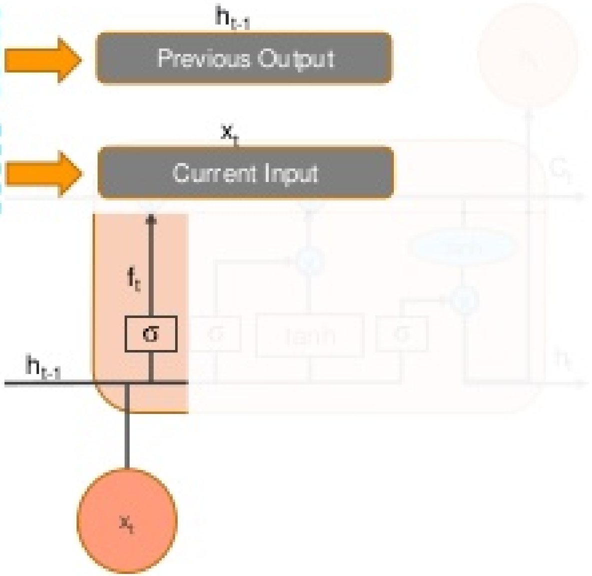
- Forget gate realizes there might be a change in context after encountering the first full stop.
- Compares with the current input sentence at x_t
- The next sentence talks about John, so the information on Alice is deleted.
- The position of subject is vacated and is assigned to John

Working of LSTMs

Consider an LSTM is fed with the following inputs from previous and present time step :

Alice is good in Physics. John on the other hand is good in Chemistry.

John plays football well. He told me yesterday over phone that he had served as the captain of his college football team.



Forget gate realizes there might be a change in context after encountering the first full stop.

Compares with the current input sentence at x_t

The next sentence talks about John, so the information on Alice is deleted.

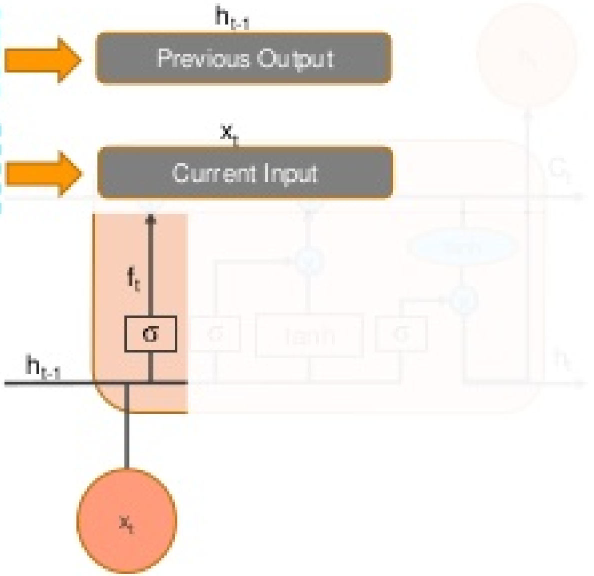
The position of subject is vacated and is assigned to John

Working of LSTMs

Consider an LSTM is fed with the following inputs from previous and present time step :

Alice is good in Physics. John on the other hand is good in Chemistry.

John plays football well. He told me yesterday over phone that he had served as the captain of his college football team.



Forget gate realizes there might be a change in context after encountering the first full stop.

Compares with the current input sentence at x_t

The next sentence talks about John, so the information on Alice is deleted.

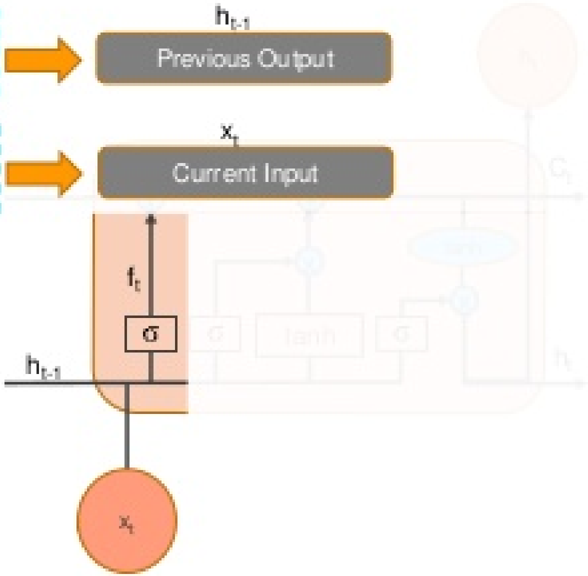
The position of subject is vacated and is assigned to John

Working of LSTMs

Consider an LSTM is fed with the following inputs from previous and present time step :

Alice is good in Physics. John on the other hand is good in Chemistry.

John plays football well. He told me yesterday over phone that he had served as the captain of his college football team.



- Forget gate realizes there might be a change in context after encountering the first full stop.
- Compares with the current input sentence at x_t
- The next sentence talks about John, so the information on Alice is deleted.
- The position of subject is vacated and is assigned to John

Working of LSTMs

Step-2

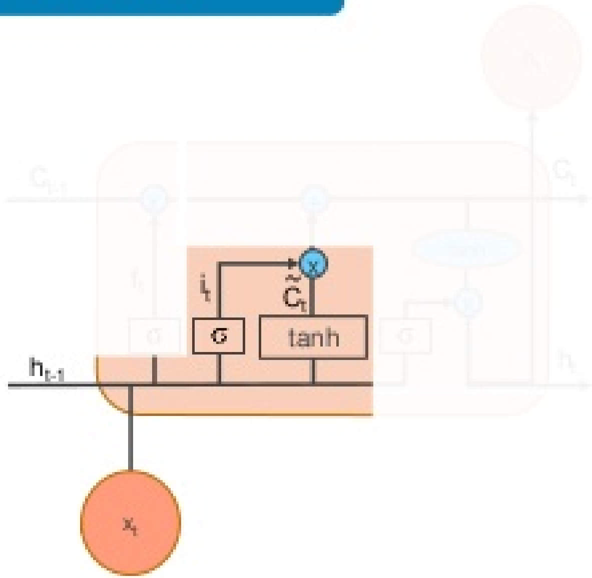
Decides how much should this unit add to the current state

In the second layer, there are 2 parts. One is the sigmoid function and the other is the tanh. In the ***sigmoid*** function, it decides which values to let through(0 or 1). ***tanh*** function gives the weightage to the values which are passed deciding their level of importance(-1 to 1).

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

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

i_t = input gate
Determines which information to let through based on its significance in the current time step

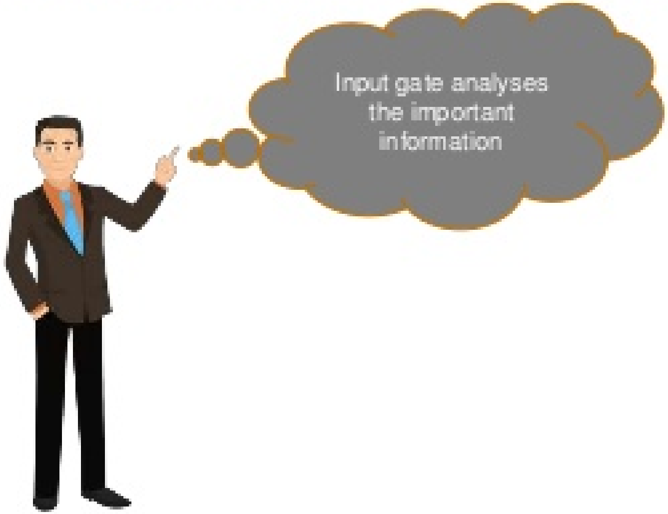


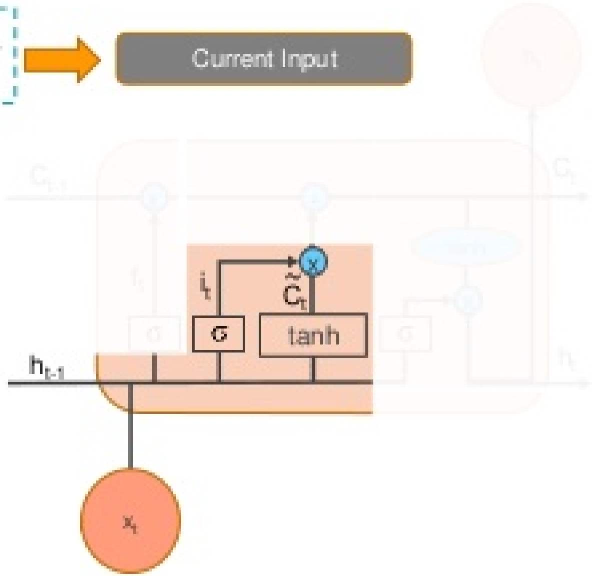
Working of LSTMs

Consider the current input at x_t

John plays football well. He told me yesterday over phone that he had served as the captain of his college football team.

Current Input



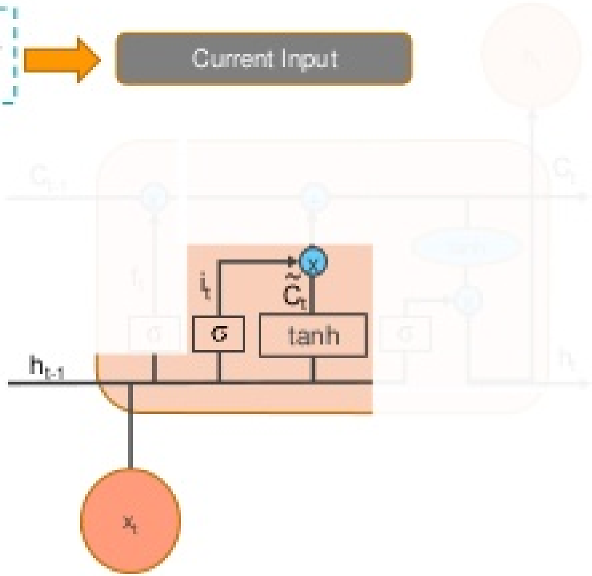


Working of LSTMs

Consider the current input at x_t

John plays football well. He told me yesterday over phone that he had served as the captain of his college football team.

Current Input

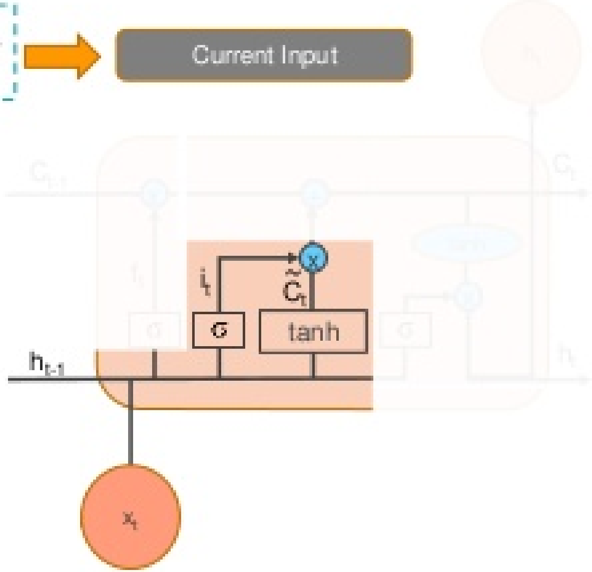


Working of LSTMs

Consider the current input at x_t

John plays football well. He told me yesterday over phone that he had served as the captain of his college football team.

Current Input

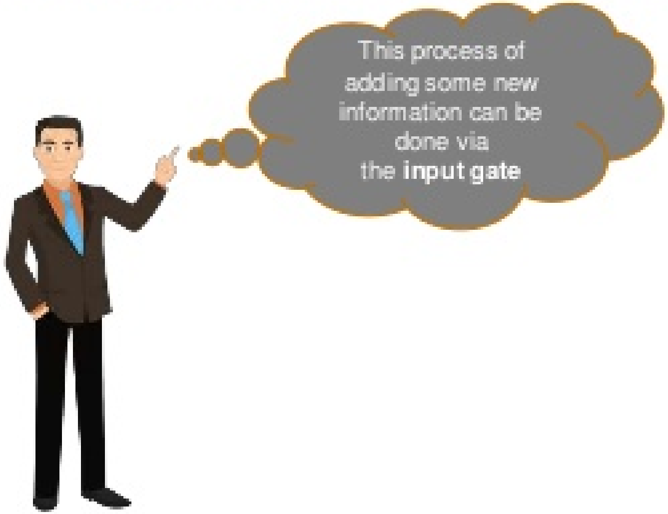


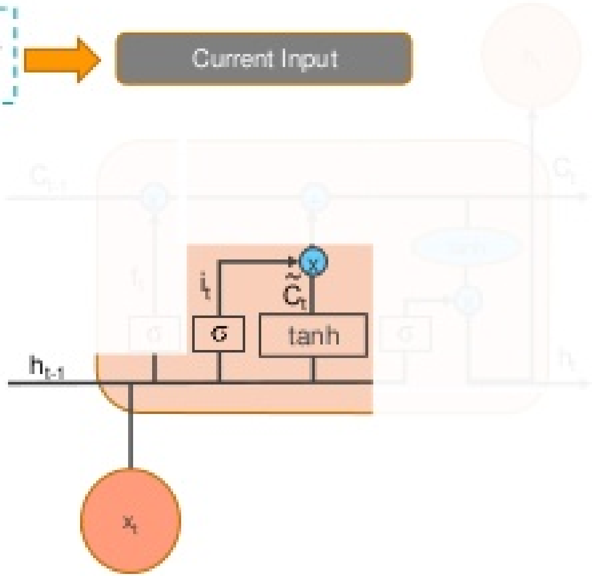
Working of LSTMs

Consider the current input at x_t

John plays football well. He told me yesterday over phone that he had served as the captain of his college football team.

Current Input





simplilearn

Working of LSTMs

Step-3

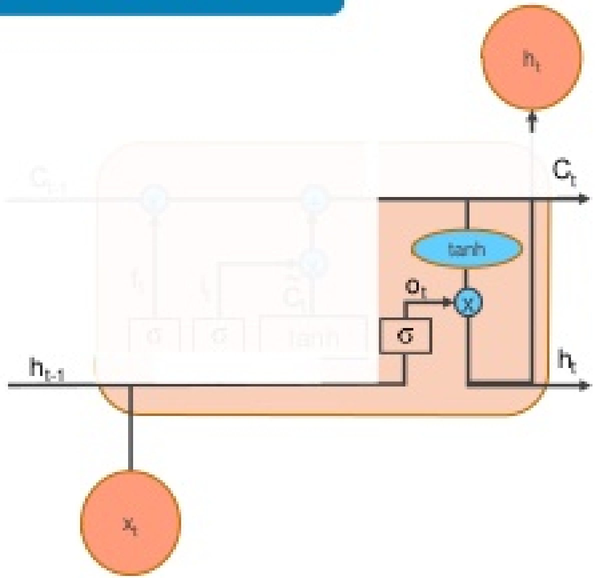
Decides what part of the current cell state makes it to the output

The third step is to decide what will be our output. First, we run a sigmoid layer which decides what parts of the cell state make it to the 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.

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

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

o_t = output gate
Allows the passed in information to impact the output in the current time step

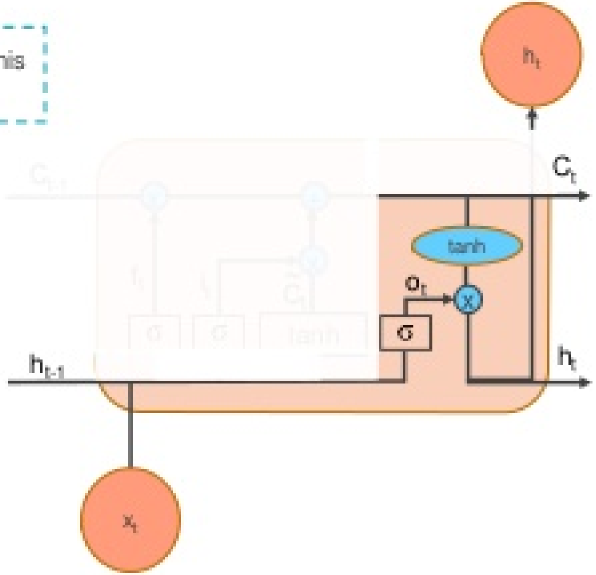


simplilearn

Working of LSTMs

Let's consider this example to predicting the next word in the sentence:

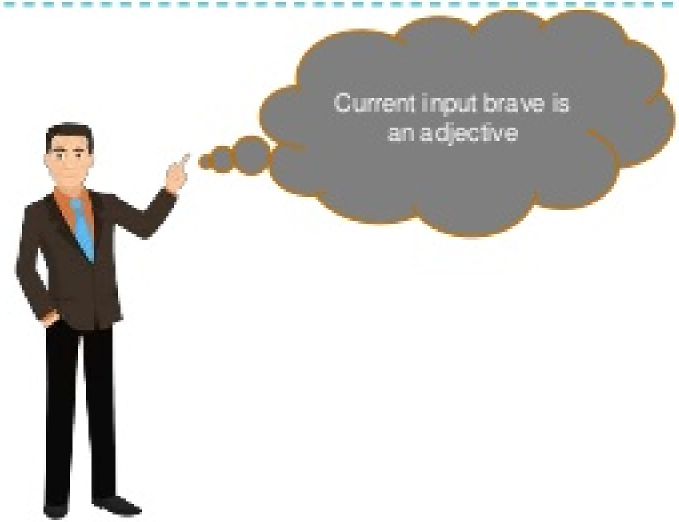
John played tremendously well against the opponent and won for his team. For his contributions, brave ____ was awarded player of the match.

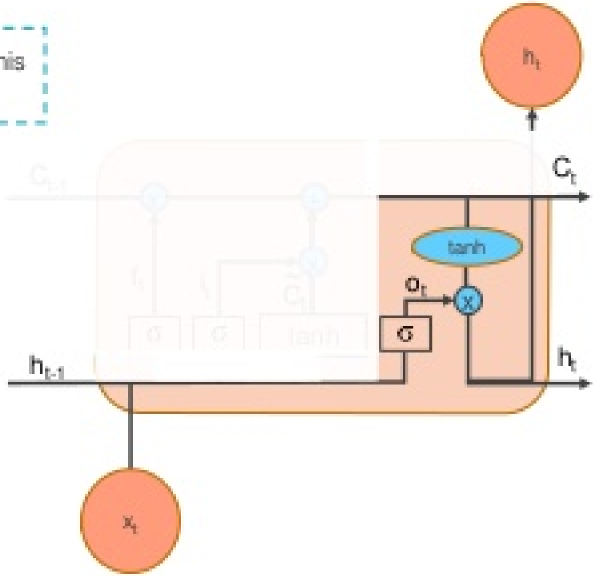


Working of LSTMs

Let's consider this example to predicting the next word in the sentence:

John played tremendously well against the opponent and won for his team. For his contributions, brave ____ was awarded player of the match.

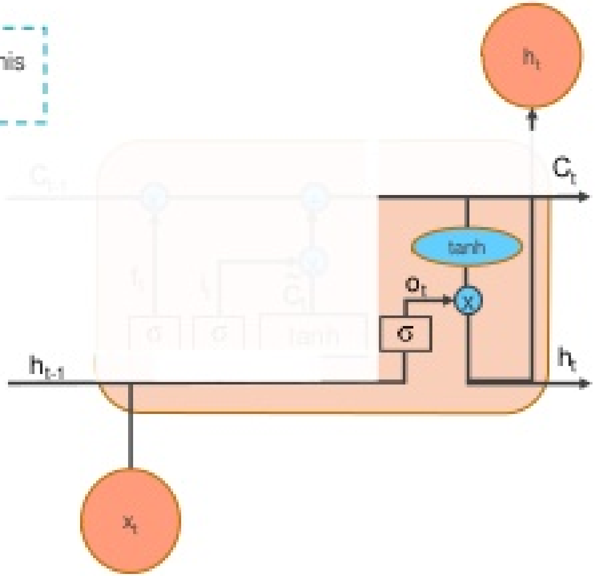




Working of LSTMs

Let's consider this example to predicting the next word in the sentence:

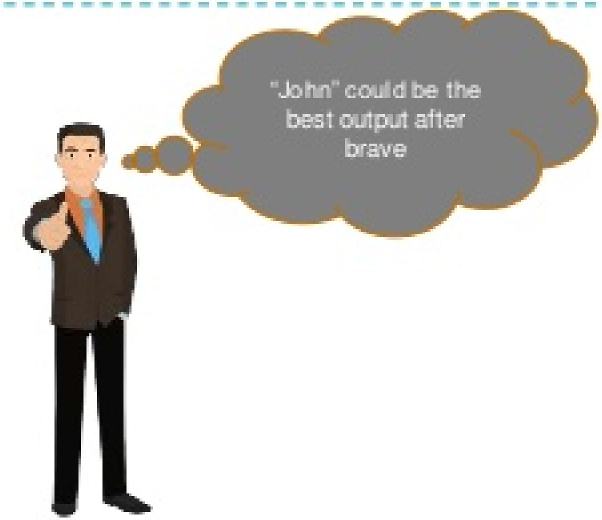
John played tremendously well against the opponent and won for his team. For his contributions, brave ____ was awarded player of the match.

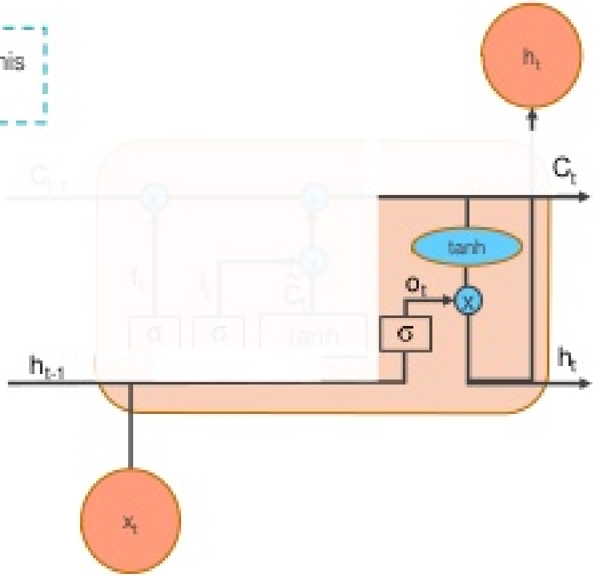


Working of LSTMs

Let's consider this example to predicting the next word in the sentence:

John played tremendously well against the opponent and won for his team. For his contributions, brave ____ was awarded player of the match.

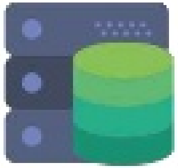




Use case implementation of LSTM

Let's predict the prices of stocks using LSTM network

Based on the stock price data between 2012-2016



Predict the stock prices of 2017

simplilearn

Use case implementation of LSTM

1. Import the Libraries

```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

2. Import the training dataset

```
# Importing the training set
dataset_train = pd.read_csv('/home/ubuntu/Downloads/Google_Stock_Price_Train.csv')
training_set = dataset_train.iloc[:, 1:2].values
```

3. Feature Scaling

```
# Feature Scaling
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler(feature_range = (0, 1))
training_set_scaled = sc.fit_transform(training_set)
```

Use case implementation of LSTM

4. Create a data structure with 60 timesteps and 1 output

```
# Creating a data structure with 60 timesteps and 1 output
X_train = []
y_train = []
for i in range(60, 1258):
    X_train.append(training_set_scaled[i-60:i, 0])
    y_train.append(training_set_scaled[i, 0])
X_train, y_train = np.array(X_train), np.array(y_train)

# Reshaping
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
```

5. Import keras libraries and packages

```
# Importing the Keras libraries and packages
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Dropout
```

```
/opt/anaconda3/lib/python3.6/site-packages/h5py/__init__.py:36: FutureWarning: Conversion of the second argument of issubdtype
from `float` to `np.floating` is deprecated. In future, it will be treated as `np.float64 == np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
Using TensorFlow backend.
```

simplilearn

Use case implementation of LSTM

6. Initialize the RNN

```
# Initialising the RNN
regressor = Sequential()
```

7. Adding the LSTM layers and some Dropout regularization

```
# Adding the first LSTM layer and some Dropout regularisation
regressor.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.shape[1], 1)))
regressor.add(Dropout(0.2))
```

```
# Adding a second LSTM layer and some Dropout regularisation
regressor.add(LSTM(units = 50, return_sequences = True))
regressor.add(Dropout(0.2))
```

```
# Adding a third LSTM layer and some Dropout regularisation
regressor.add(LSTM(units = 50, return_sequences = True))
regressor.add(Dropout(0.2))
```

```
# Adding a fourth LSTM layer and some Dropout regularisation
regressor.add(LSTM(units = 50))
regressor.add(Dropout(0.2))
```

Use case implementation of LSTM

8. Adding the output layer

```
# Adding the output layer
regressor.add(Dense(units = 1))
```

9. Compile the RNN

```
# Compiling the RNN
regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')
```

10. Fit the RNN to the training set

```
# Fitting the RNN to the Training set
regressor.fit(X_train, y_train, epochs = 100, batch_size = 32)

Epoch 1/100
1198/1198 [==============================] - 18s 15ms/step - loss: 0.0031
Epoch 2/100
1198/1198 [==============================] - 13s 11ms/step - loss: 0.0070
Epoch 3/100
1198/1198 [==============================] - 13s 11ms/step - loss: 0.0054
Epoch 4/100
1198/1198 [==============================] - 13s 11ms/step - loss: 0.0058
Epoch 5/100
1198/1198 [==============================] - 13s 11ms/step - loss: 0.0053
Epoch 6/100
1198/1198 [==============================] - 13s 11ms/step - loss: 0.0044
Epoch 7/100
1198/1198 [==============================] - 13s 11ms/step - loss: 0.0045
Epoch 8/100
1198/1198 [==============================] - 13s 11ms/step - loss: 0.0047
Epoch 9/100
```

Use case implementation of LSTM

11. Load the stock price test data for 2017

```
# Load the real stock price of 2017
dataset_test = pd.read_csv('/home/ubuntu/Downloads/Google_Stock_Price_Test.csv')
real_stock_price = dataset_test.iloc[:, 1:2].values
```

12. Get the predicted stock price of 2017

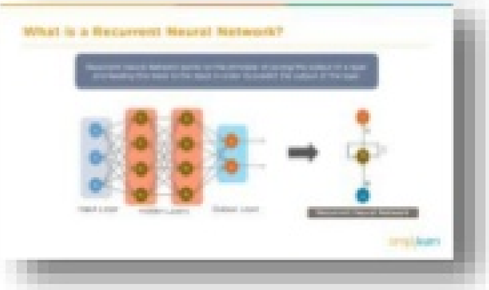
```
# Getting the predicted stock price of 2017
dataset_total = pd.concat((dataset_train['Open'], dataset_test['Open']), axis = 0)
inputs = dataset_total[len(dataset_total) - len(dataset_test) - 60:].values
inputs = inputs.reshape(-1,1)
inputs = sc.transform(inputs)
X_test = []
for i in range(60, 80):
    X_test.append(inputs[i-60:i, 0])
X_test = np.array(X_test)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
predicted_stock_price = regressor.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price)
```

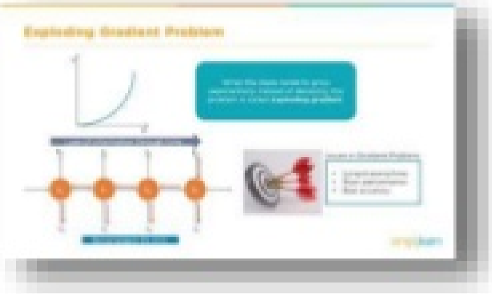
Use case implementation of LSTM

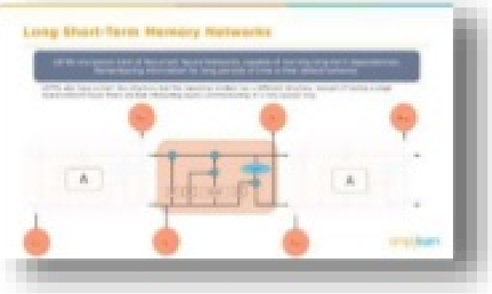
13. Visualize the results of predicted and real stock price

```
# Visualising the results
plt.plot(real_stock_price, color = 'red', label = 'Real Google Stock Price')
plt.plot(predicted_stock_price, color = 'blue', label = 'Predicted Google Stock Price')
plt.title('Google Stock Price Prediction')
plt.xlabel('Time')
plt.ylabel('Google Stock Price')
plt.legend()
plt.show()
```

Key Takeaways








THANK YOU
For more information, visit
www.simplilearn.com
simplilearn