Long Short Term Memory (LSTM)

By - Gaurav Kumar Yadav

May 23rd, 2020

3.00PM onwards

Agenda of today

Overview

Understanding the problems with RNN

LSTM Objective

Features of LSTM

Types of RNN (LSTM)

One to one lstm

One to many lstm

Many to many lstm

Many to one lstm

Implementation on Jupyter notebook

Overview

Why Recurrent Neural Network?

To learn the sequential data like Video, audio, paragraph writing, music generation, sentiment analysis, language translation from one medium to another medium(google translator).

ANN output does not depend on the previous output, so it is unable to predict correctly to those data where current output is highly dependent on sequence of previous output.

Number of input and output unit can not varies during the testing phase, means if you build a model which take 10 input feature and predict the output as two 0 and 1 (just example), then the output size should not varies during the test time whereas in RNN (LSTM) it can varies.

Understanding the problems
If RNN were there then why LSTM

01 During the backpropagation through time due to chain rule of multiplication of the gradient there is high probability that after some multiplication the loss gradient will become either very small , vanish, or very big , explode.

02 The case when gradient explode from certain limit, it will also affect in weight updation of starting layers neuran compare to last layer neurons. To counter this problem we use concept of gradient clipping. Where we set a limit, when gradient will cross that limit we clip then or normalize them

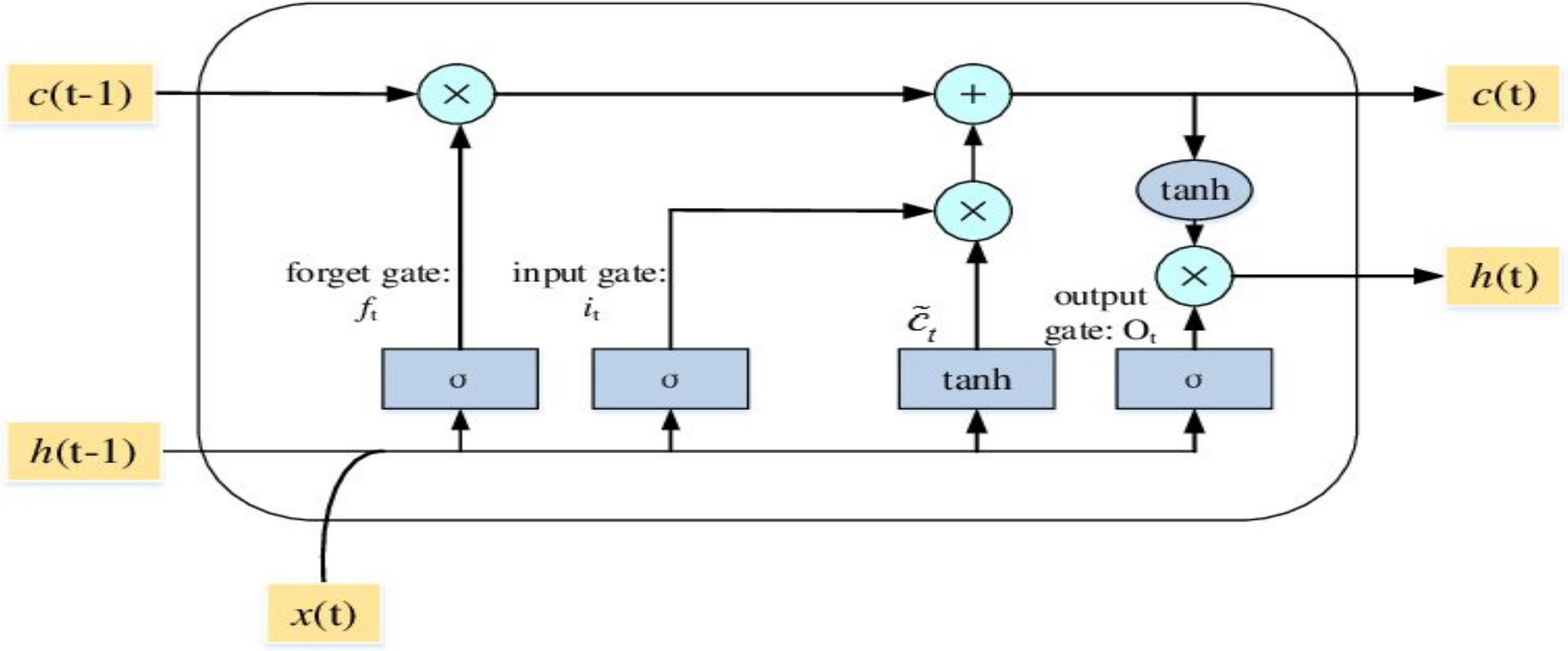
03 The most important problem which occur generally because the output of the activation function varies between 0 and 1 or -1 to 1. There is a huge probability that after some multiplication gradient will reach nearer to zero, and vanish the effect of updation in start layer of neurons weights. This is called Vanishing Gradient Problem.

LSTM objective

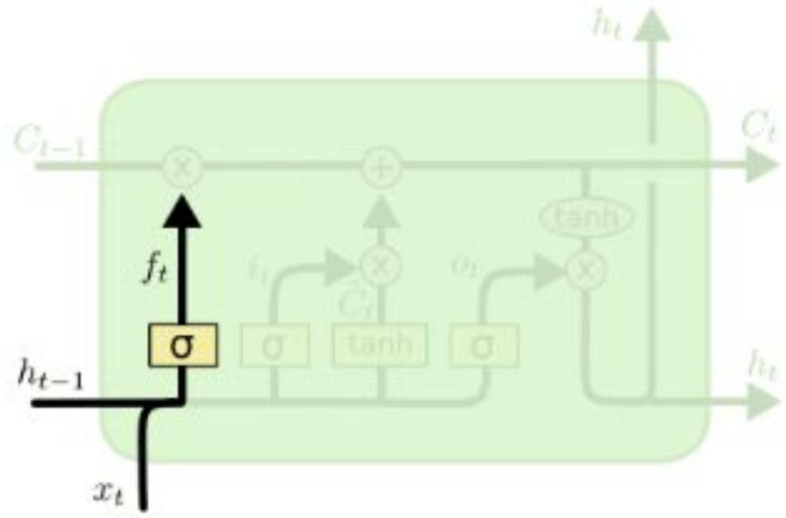
To counter or finish the problem of vanishing gradient we use Long short term memory concept. In which we gate concept to remember the dependencies of the previous layer in the sequence.

It uses additive concept instead of multiplication in chain rule.

Structure of LSTM

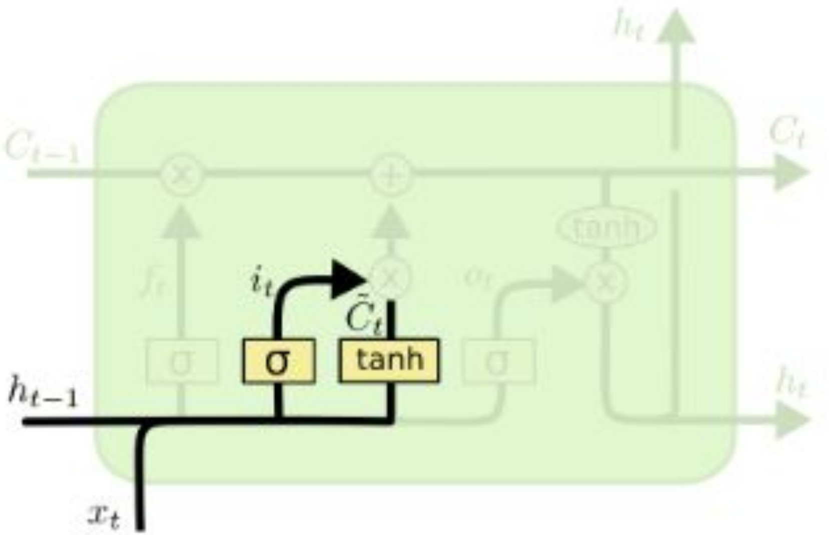


Forget gate



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

Input gate and memory cell



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

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

Whole parameters used in LSTM

$$
\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \\
g_t &= \phi(W_{xg}x_t + W_{hg}h_{t-1} + b_g) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot \phi(c_t)
\end{aligned}
$$

Types of RNN (LSTM)

Now a days Four types of RNN (LSTM) we are using

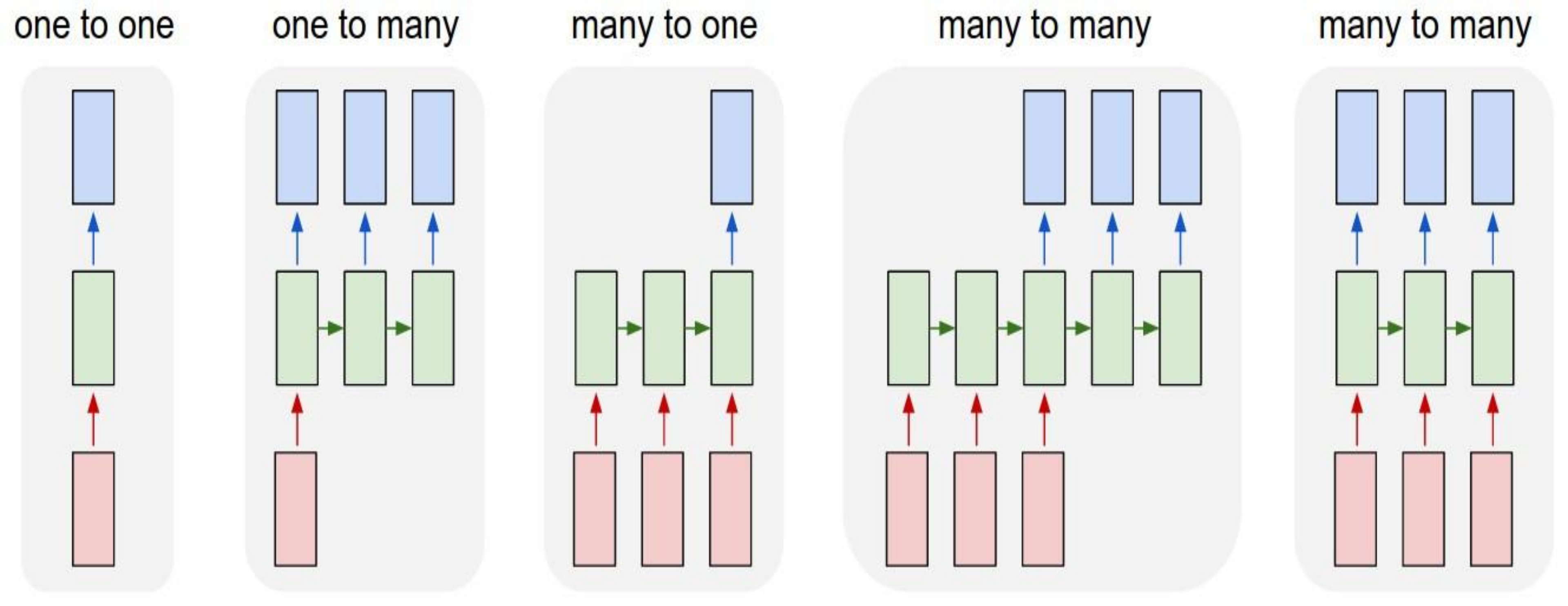
1- ONE TO ONE → one input one output (less commonaly use)

2- ONE TO MANY → Music generation

3- MANY TO ONE → Sentiment analysis (Movie rating)

4- MANY TO MANY → Machine translation , Name entity

Types of RNN (LSTM)



Implementation of LSTM using keras API on google colab

https://github.com/gaurav-vision/Recurrent-Neural-Network

Thank you!