RNN and LSTM

PRESENTED BY Donguk Shin, Chenwei Wu, Cheng-Yun Tsai

11/15/2022

Recurrent Neural Network (RNN)

01

NYU

What is RNN?

- Recurrent Neural Networks(RNNs) are a type of Neural Network that allow previous output to be used as input while having hidden states
- RNN save the previous output of a particular layer and feed this back to the input in order to predict the output of the layer.
- Used for language translation, natural language processing (nlp), and time series prediction

NYU

Why RNN?

Feed-Forward Neural Network :

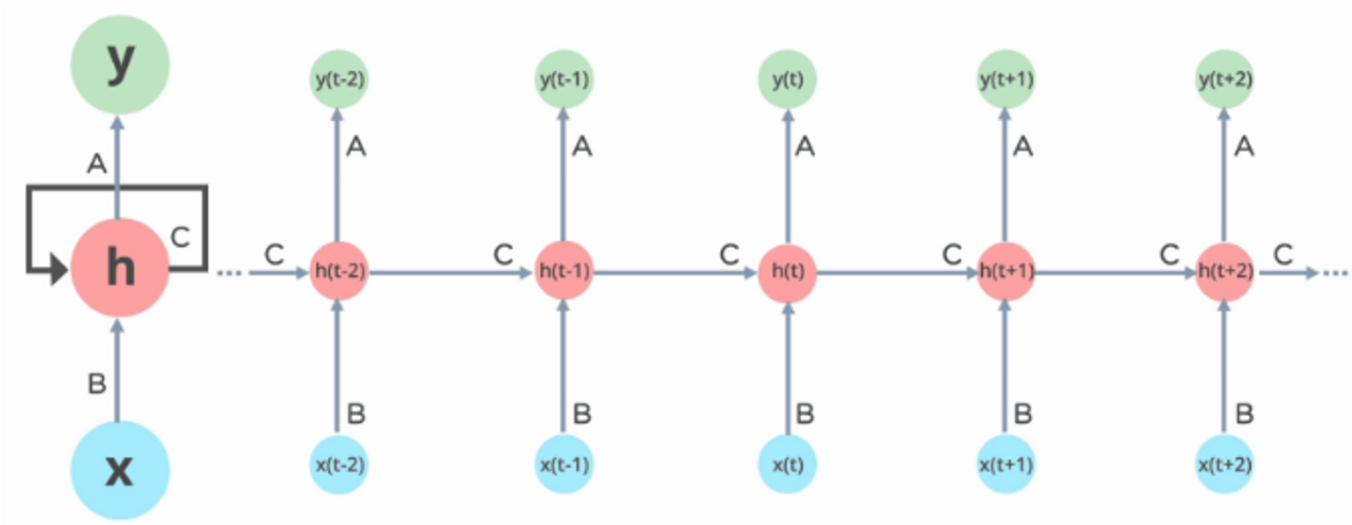
- Cannot handle sequential data
- Considers only the current input
- Cannot memorize previous inputs

RNN solves these issues:

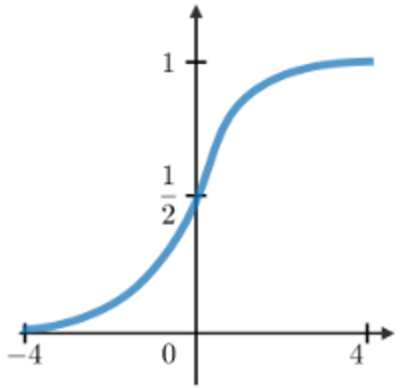
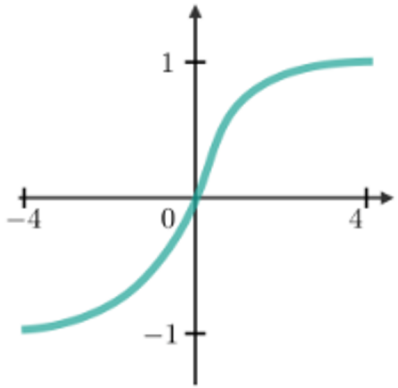
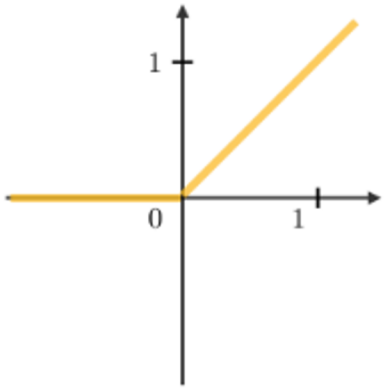
- can handle sequential data
- accepts the current input data and previously received inputs
- can memorize previous inputs

NYU

Architecture of RNN



Common Activation functions

Sigmoid	Tanh	RELU
$g(z) = \frac{1}{1+e^{-z}}$	$g(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}}$	$g(z) = \max(0, z)$
		

Pros and Cons

Pros

- Process input of any length
- Model size not increase with size of input
- Remember historical information which is useful for time series prediction

Cons

- Gradient vanishing and exploding problems.
- Training an RNN is a very difficult task.
- Difficult to access information from a long time ago

NYU

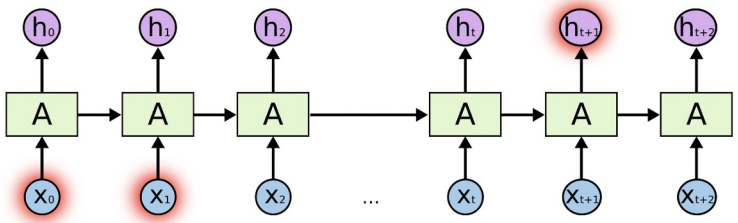
Long Short-Term Memory (LSTM)

02

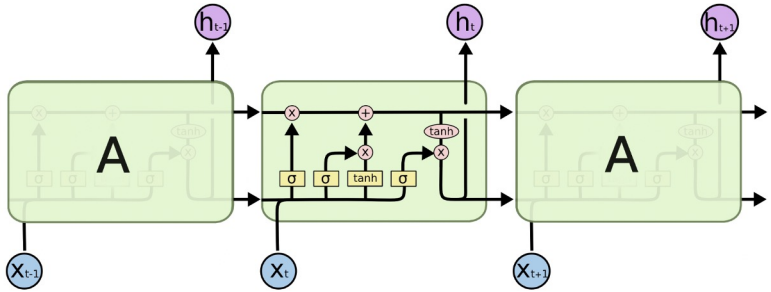
NYU

2. Long short-term memory(LSTM)

When training the traditional RNN using back-propagation, the long-term gradients which are back-propagated can vanish or explode, making it difficult to handle a long sequences of data.



LSTM model is a special type of RNN that has both "long-term memory" and "short-term memory". Its architecture allows each cell to remember values over arbitrary time intervals and regulate the flow of information into and out of the cell. And this model was developed to deal with the vanishing gradient problem

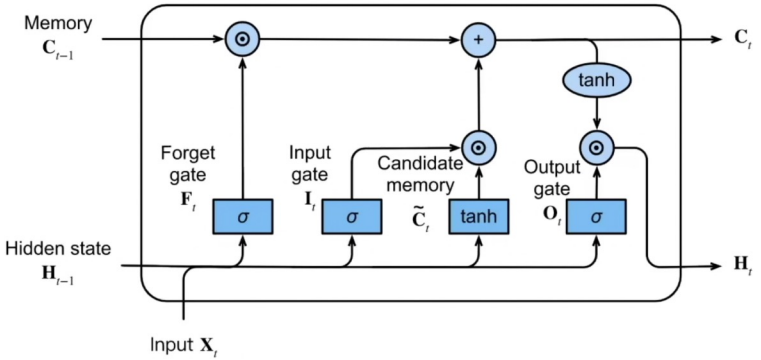


NYU

2. Long short-term memory(LSTM)

A common LSTM unit is composed of **a cell, three gates, and two hidden states**.

- The first step is to decide what information we're going to throw. This decision is made by a sigmoid layer called the "forget gate."
- The next step is to decide what new information we're going to store in the cell state. This is achieved by two parts, "input gate" and "candidate memory".
- Finally, we need to decide what we're going to output. This is based on the new memory state and another sigmoid layer called "output gate".



$$\boldsymbol{I}_t = \sigma(\boldsymbol{X}_t \boldsymbol{W}_{xi} + \boldsymbol{H}_{t-1} \boldsymbol{W}_{hi} + \boldsymbol{b}_i)$$
$$\boldsymbol{F}_t = \sigma(\boldsymbol{X}_t \boldsymbol{W}_{xf} + \boldsymbol{H}_{t-1} \boldsymbol{W}_{hf} + \boldsymbol{b}_f)$$
$$\boldsymbol{O}_t = \sigma(\boldsymbol{X}_t \boldsymbol{W}_{xo} + \boldsymbol{H}_{t-1} \boldsymbol{W}_{ho} + \boldsymbol{b}_o)$$
$$\tilde{\boldsymbol{C}}_t = \tanh(\boldsymbol{X}_t \boldsymbol{W}_{xc} + \boldsymbol{H}_{t-1} \boldsymbol{W}_{hc} + \boldsymbol{b}_c)$$
$$\boldsymbol{C}_t = \boldsymbol{F}_t \odot \boldsymbol{C}_{t-1} + \boldsymbol{I}_t \odot \tilde{\boldsymbol{C}}_t$$
$$\boldsymbol{H}_t = \boldsymbol{O}_t \odot \tanh(\boldsymbol{C}_t)$$

NYU

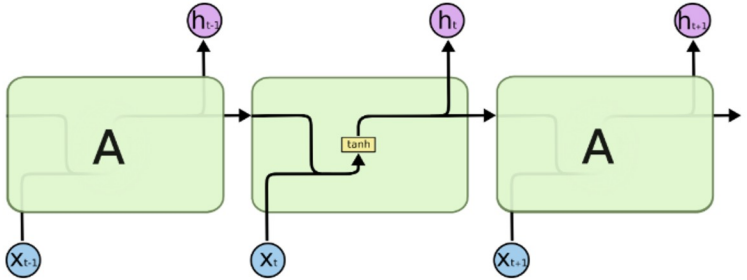
RNN vs. LSTM

03

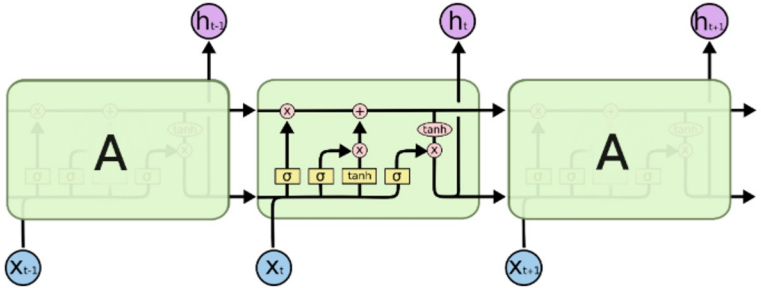
NYU

RNN and LSTM

- Recurrent Neural Network is a generalization of feedforward neural network that has an internal memory.
- Simple RNN cannot process very long sequences because the gradient of the loss function decays exponentially with time.
- LSTM model is a special type of RNN (Gated Recurrent Neural Networks) which enables RNNs to remember inputs over a long period of time.



The repeating module in a standard RNN contains a single layer.



The repeating module in an LSTM contains four interacting layers.

NYU

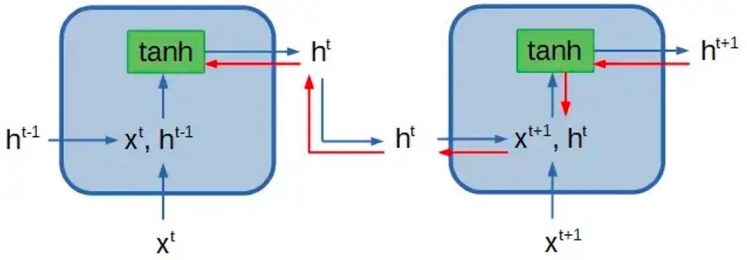
Vanishing and Exploding Gradient Problem

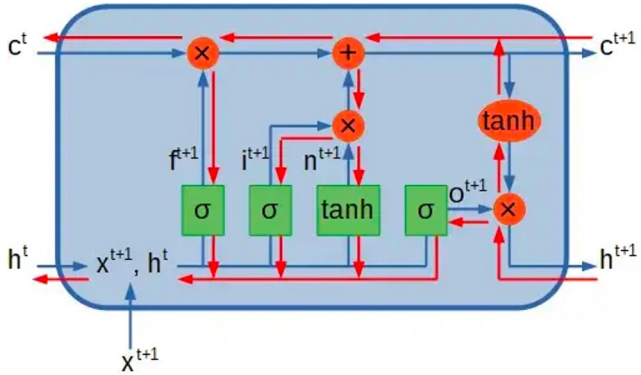
- Everytime we go back a time step, we need to make an inner product between our current gradient and the weight matrix.
- Vanishing gradient issue for 0.9:

$$0.9^{100} = 0.000017...$$

- For the LSTM, there's is a set of weights which can be learned such that

$$\sigma(\cdot) \approx 1 \qquad \frac{\partial s_{t'}}{\partial s_t} = \prod_{k=1}^{t'-t} \sigma(v_{t+k})$$





Types of RNN in Keras

SimpleRNN

- A fully-connected RNN where the output from previous timestep is to be fed to next timestep.

GRU
(Gated Recurrent Unit)

- First proposed in Cho et al., 2014.
- Computationally easier than LSTM (GRU only has 2 gates).
- Similar performance with LSTM and more efficient (less parameters).

LSTM

- First proposed in Hochreiter & Schmidhuber, 1997.

NYU

References

https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn

https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks

https://en.wikipedia.org/wiki/Long_short-term_memory

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

https://d2l.ai/chapter_recurrent-modern/lstm.html