INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Ceará

HUAWEI

Recurrent Neural Networks

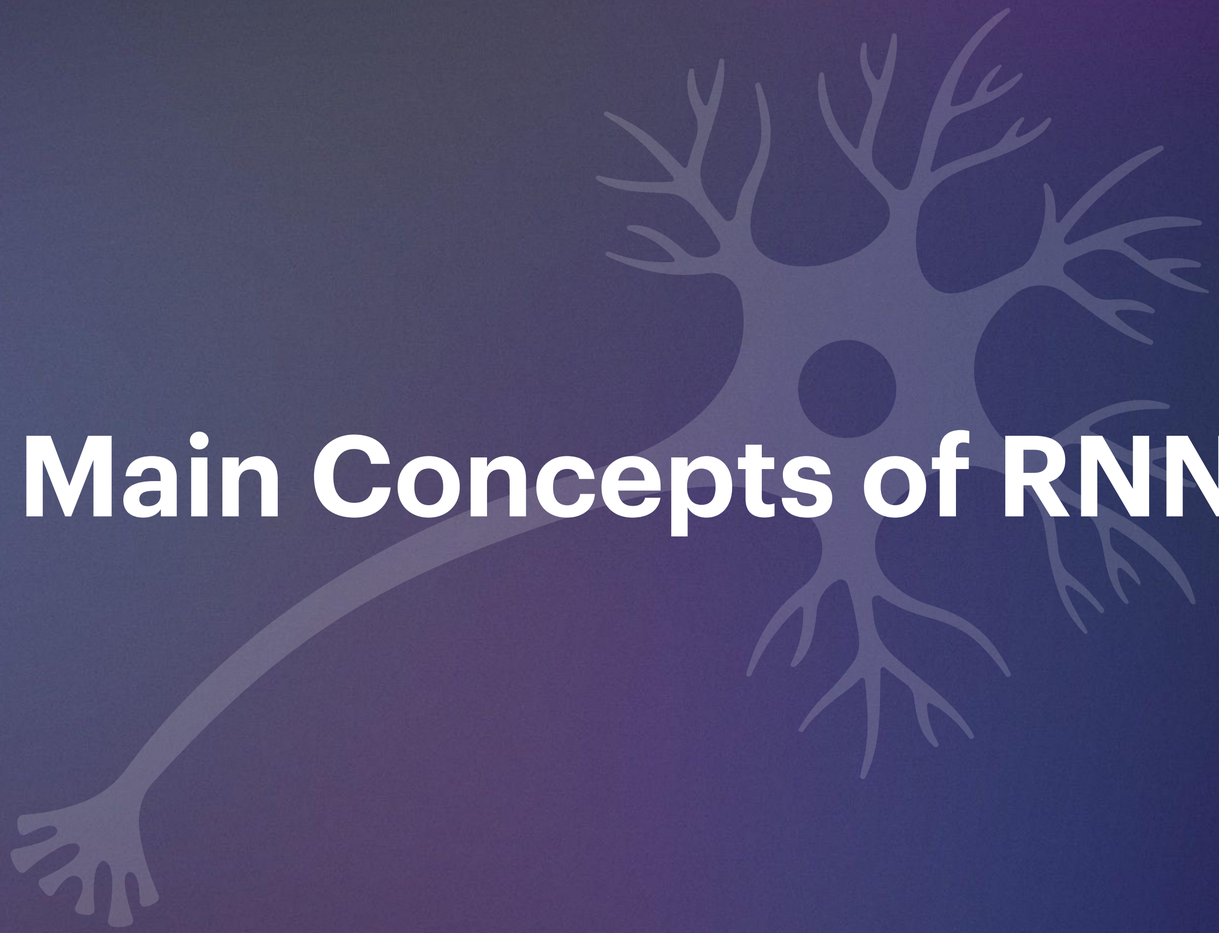
Prof. Me. Saulo A. F. Oliveira
saulo.oliveira@ifce.edu.br

Recurrent Neural Network

- The recurrent neural network (RNN) is a neural network that captures dynamic information in sequential data through periodical connections of hidden layer nodes. It can classify sequential data.
- Unlike other forward neural networks, the RNN can keep a context state and even store, learn, and express related information in context windows of any length. Different from traditional neural networks, it is not limited to the space boundary, but also supports time sequences. In other words, there is a side between the hidden layer of the current moment and the hidden layer of the next moment.
- The RNN is widely used in scenarios related to sequences, such as videos consisting of image frames, audio consisting of clips, and sentences consisting of words.

01

Main Concepts of RNN



INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Ceará

HUAWEI

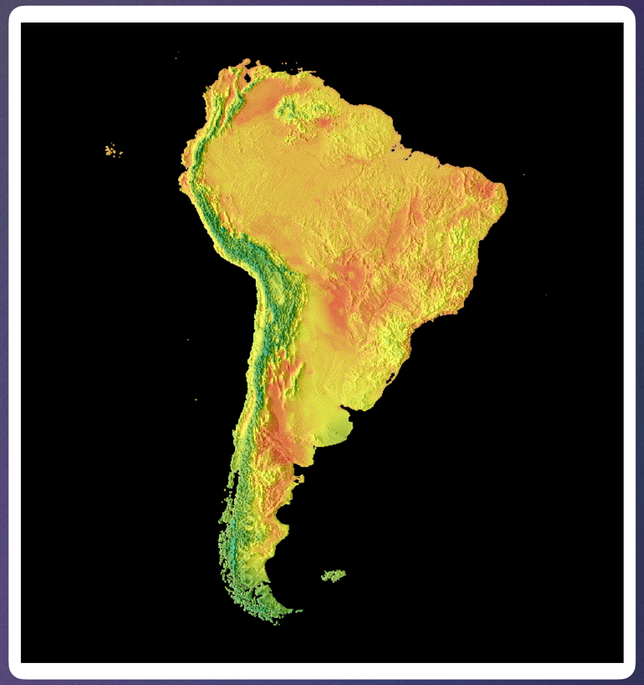
Sequential Data

- **What is sequential data?**

  Whenever the samples are dependent on the other samples in the dataset, the data is sequential. A typical example of such a kind of data is Time-series, such as stock price or sensor data. In both cases, each sample represents an observation at a certain point in time. There are other examples of sequential data like sequences, gene sequences, and weather data.

- **Why is sequential data an issue for the traditional neural network?**

  For a fact, the traditional neural networks can't typically handle these types of data well. However, it is essential to know that the model is stateless. It does not remember the data that it just analyzed—all it does is take input after input and produce an individual classification for every day.



https://developers.google.com/earth-engine/guides/ic_visualization

A traditional neural network assumes the data is non-sequential. Each data point is independent of other data points. As a result, the inputs are analyzed in isolation, which can cause problems if there are dependencies in the data.

https://medium.com/analytics-vidhya/sequential-data-and-the-neural-network-conundrum-b2c005f8f865

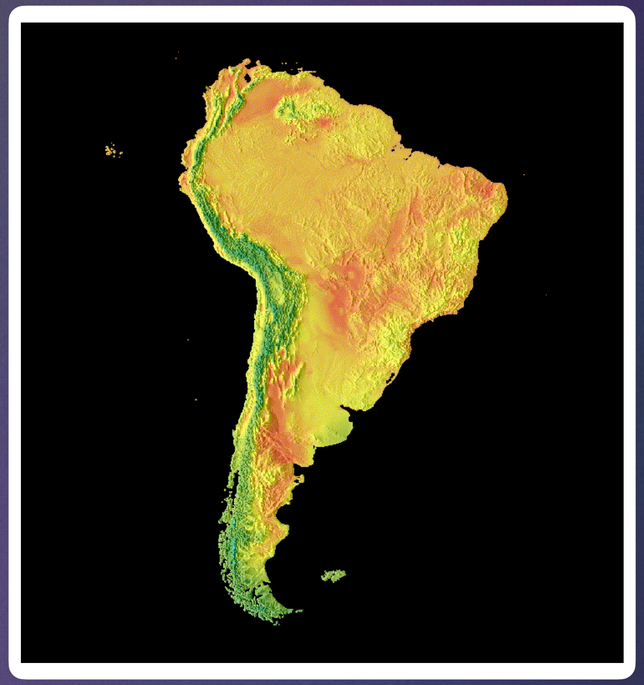
Sequential Data

- **What is sequential data?**

  Whenever the samples are dependent on the other samples in the dataset, the data is sequential. A typical example of such a kind of data is Time-series, such as stock price or sensor data. In both cases, each sample represents an observation at a certain point in time. There are other examples of sequential data like sequences, gene sequences, and weather data.

- **Why is sequential data an issue for the traditional neural network?**

  For a fact, the traditional neural networks can't typically handle these types of data well. However, it is essential to know that the model is stateless. It does not remember the data that it just analyzed—all it does is take input after input and produce an individual classification for every day.



https://developers.google.com/earth-engine/guides/ic_visualization

A traditional neural network assumes the data is non-sequential. Each data point is independent of other data points. As a result, the inputs are analyzed in isolation, which can cause problems if there are dependencies in the data.

https://medium.com/analytics-vidhya/sequential-data-and-the-neural-network-conundrum-b2c005f8f865

The Problem of Long-Term Dependencies

- Sometimes, we only need to look at recent information to perform the present task. For example, consider a language model trying to predict the next word based on the previous ones. If we are trying to predict the last word in **The clouds are in the ...** we don't need any further context – it's pretty obvious the next word is going to be sky. In such cases, where the gap between the relevant information and the place that it's needed is small, RNNs can learn to use the past information.

- But there are also cases where we need more context. Consider trying to predict the last word in the text. **I grew up in Canada... I speak fluently** . Recent information suggests that the next word is probably the name of a language, but if we want to narrow down which language, we need the context of Canada, from further back. It's entirely possible for the gap between the relevant information and the point where it is needed to become very large.

https://colah.github.io/posts/2015-08-Understanding-LSTMs/?source=post_page-----79e5eb8049c9----------

The Problem of Long-Term Dependencies

- Sometimes, we only need to look at recent information to perform the present task. For example, consider a language model trying to predict the next word based on the previous ones. If we are trying to predict the last word in **The clouds are in the ...** **sky** we don't need any further context – it's pretty obvious the next word is going to be sky. In such cases, where the gap between the relevant information and the place that it's needed is small, RNNs can learn to use the past information.

- But there are also cases where we need more context. Consider trying to predict the last word in the text. **I grew up in Canada... I speak fluently** . Recent information suggests that the next word is probably the name of a language, but if we want to narrow down which language, we need the context of Canada, from further back. It's entirely possible for the gap between the relevant information and the point where it is needed to become very large.

https://colah.github.io/posts/2015-08-Understanding-LSTMs/?source=post_page-----79e5eb8049c9----------

The Problem of Long-Term Dependencies

- Sometimes, we only need to look at recent information to perform the present task. For example, consider a language model trying to predict the next word based on the previous ones. If we are trying to predict the last word in **The clouds are in the ...** **sky** we don't need any further context – it's pretty obvious the next word is going to be sky. In such cases, where the gap between the relevant information and the place that it's needed is small, RNNs can learn to use the past information.

- But there are also cases where we need more context. Consider trying to predict the last word in the text. **I grew up in Canada... I speak fluently** **French** . Recent information suggests that the next word is probably the name of a language, but if we want to narrow down which language, we need the context of Canada, from further back. It's entirely possible for the gap between the relevant information and the point where it is needed to become very large.

https://colah.github.io/posts/2015-08-Understanding-LSTMs/?source=post_page-----79e5eb8049c9----------

Recurrent Neural Network Architecture (1)

Although the following figure has the sequence mark t, one can notice the recurrence by sharing $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$.

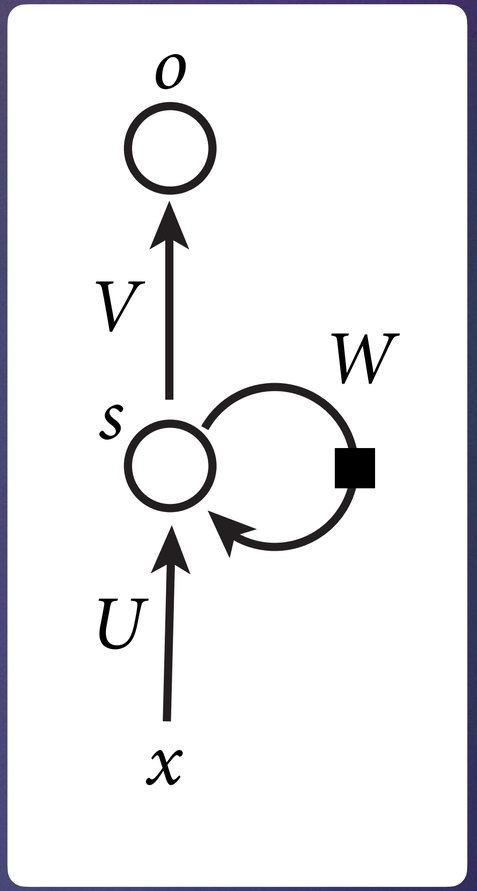
- x_t is the input of the input sequence at time t.
- s_t is the memory unit of the sequence at time t and caches previous information:

$$s_t = \tanh(\mathbf{U}x_t + \mathbf{W}s_{t-1}).$$

- O_t is the output of the hidden layer of the sequence at time t:

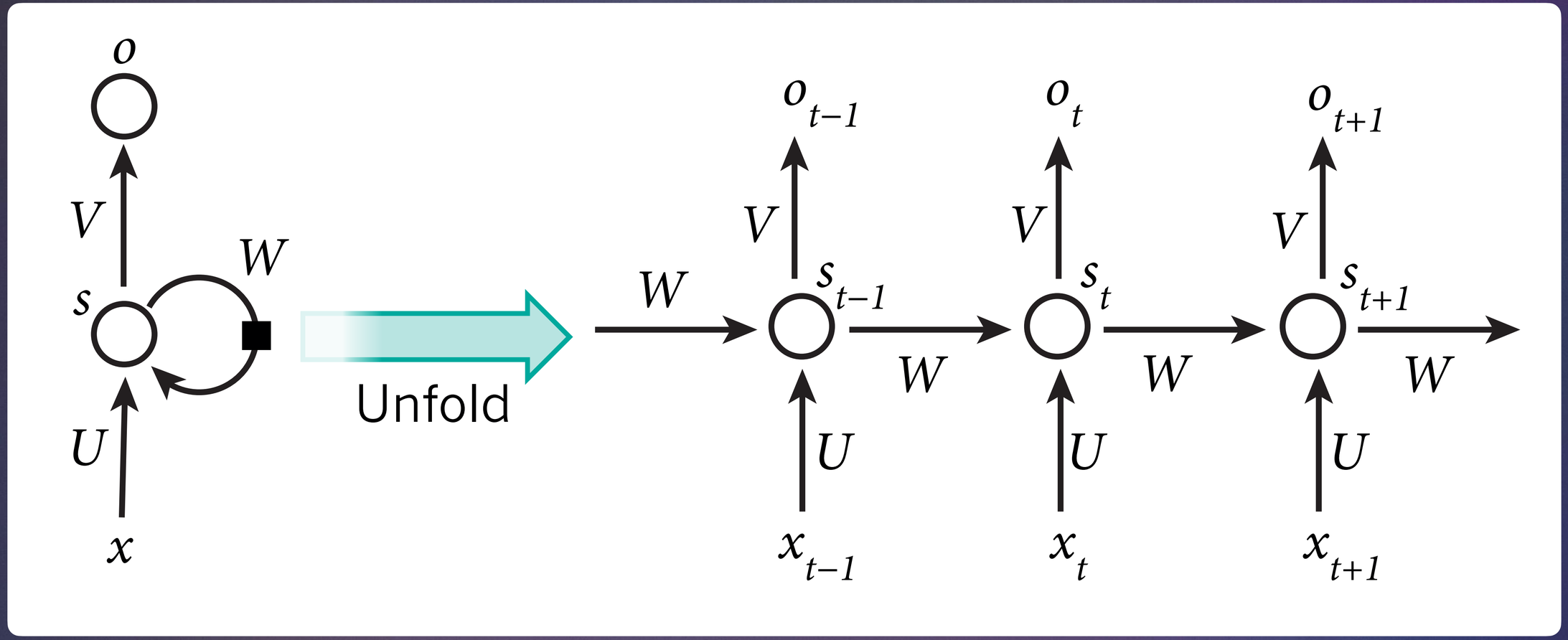
$$o_t = \tanh(\mathbf{V}s_t).$$

- O_t after through multiple hidden layers, it can get the final output of the sequence at time t.



LECUN, Yann; BENGIO, Yoshua; HINTON, Geoffrey. Deep learning. nature, v. 521, n. 7553, p. 436-444, 2015.

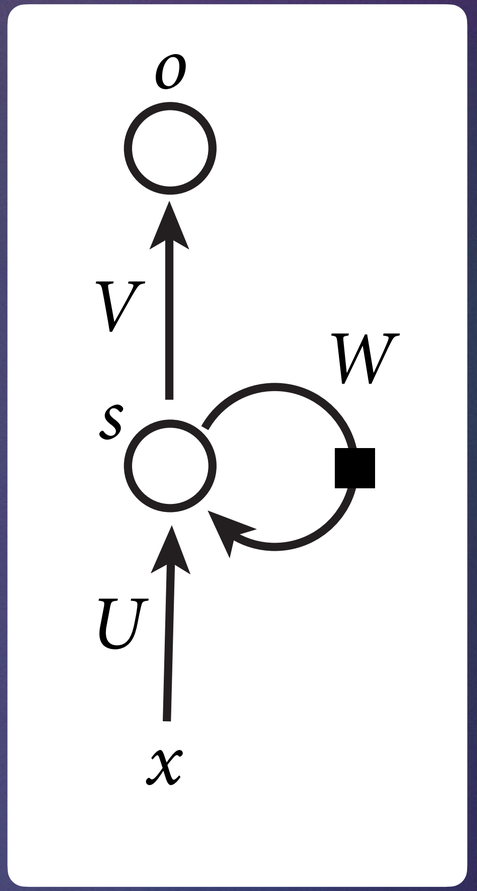
Recurrent Neural Network Architecture (2)



LECUN, Yann; BENGIO, Yoshua; HINTON, Geoffrey. Deep learning. nature, v. 521, n. 7553, p. 436-444, 2015.

Recurrent Neural Network Architecture (3)

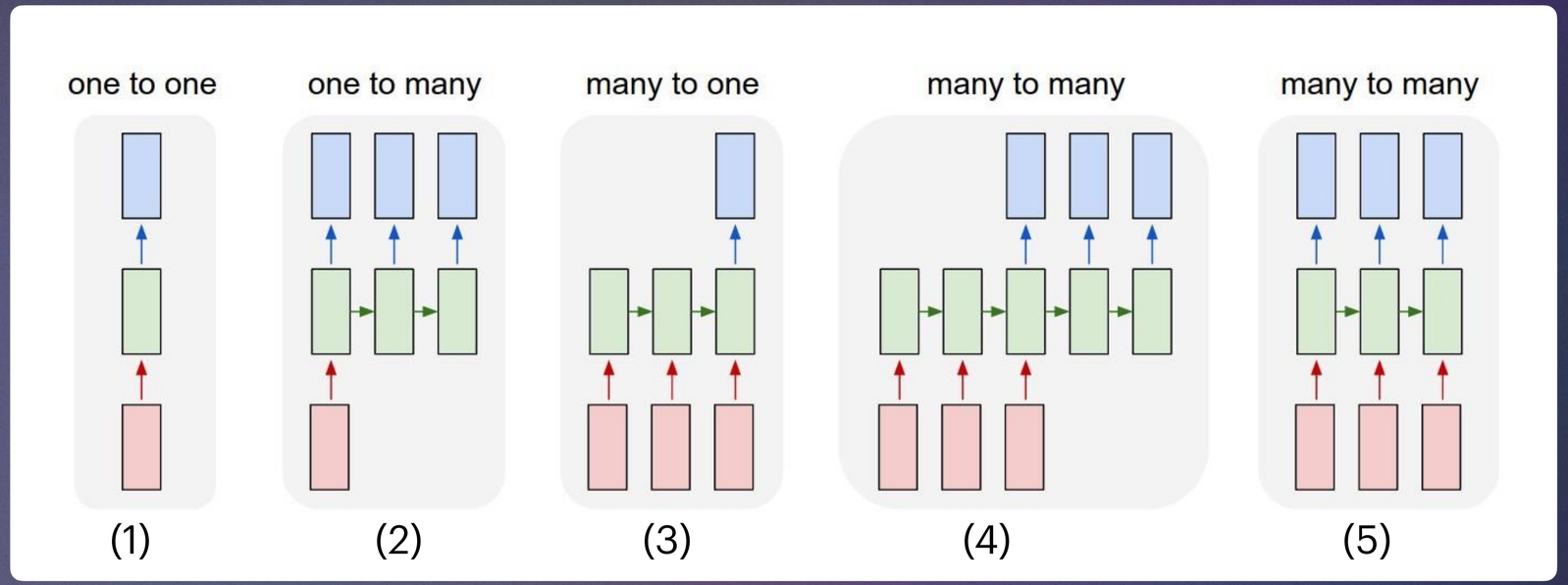
- $s_t = \sigma(\mathbf{U}x_t + \mathbf{W}s_{t-1})$ is s extended on the time sequence.
- $$s_t = \sigma\left(\mathbf{U}x_t + \mathbf{W}\left(\sigma\left(\mathbf{U}x_{t-1} + \mathbf{W}\left(\sigma\left(\mathbf{U}x_{t-2} + \mathbf{W}(\cdots)\right)\right)\right)\right)\right).$$
- Despite that the standard RNN structure solves the problem of information memory, the information attenuates during long-term memory.
- Information needs to be saved long time in many tasks. For example, a hint at the beginning of a speculative fiction may not be answered until the end.
- The RNN may not be able to save information for long due to the limited memory unit capacity.
- We expect that memory units can remember key information.



LECUN, Yann; BENGIO, Yoshua; HINTON, Geoffrey. Deep learning. nature, v. 521, n. 7553, p. 436-444, 2015.

Types of Recurrent Neural Networks

Input vectors are in red, output vectors are in blue and green vectors hold the RNN's state:



(1) Vanilla mode of processing without RNN, from fixed-sized input to fixed-sized output (e.g. image classification).

(2) Sequence output (e.g. image captioning takes an image and outputs a sentence of words).

(3) Sequence input (e.g. sentiment analysis where a given sentence is classified as expressing positive or negative sentiment).

(4) Sequence input and sequence output (e.g. Machine Translation: an RNN reads a sentence in English and then outputs a sentence in French).

(5) Synced sequence input and output (e.g. video classification where we wish to label each frame of the video).

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

02

Backpropagation Through Time

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Ceará

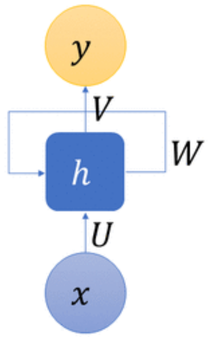
HUAWEI

Backpropagation Through Time

BPTT:

1. Traditional backpropagation is the extension on the time sequence.
2. There are two sources of errors in the sequence at time of memory unit: first is from the hidden layer output error at t time sequence; the second is the error from the memory cell at the next time sequence t + 1.
3. The longer the time sequence, the more likely the loss of the last time sequence to the gradient of w in the first time sequence causes the vanishing gradient or exploding gradient problem.
4. The total gradient of weight w is the accumulation of the gradient of the weight at all time sequence.

https://discuss.pytorch.org/t/implementing-backpropagation-through-time/69765



Three steps of BPTT:

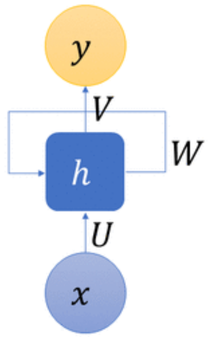
1. Computing the output value of each neuron through forward propagation.
2. Computing the error value of each neuron through backpropagation δ_j.
3. Computing the gradient of each weight.

Backpropagation Through Time

BPTT:

1. Traditional backpropagation is the extension on the time sequence.
2. There are two sources of errors in the sequence at time of memory unit: first is from the hidden layer output error at t time sequence; the second is the error from the memory cell at the next time sequence t + 1.
3. The longer the time sequence, the more likely the loss of the last time sequence to the gradient of w in the first time sequence causes the vanishing gradient or exploding gradient problem.
4. The total gradient of weight w is the accumulation of the gradient of the weight at all time sequence.

https://discuss.pytorch.org/t/implementing-backpropagation-through-time/69765



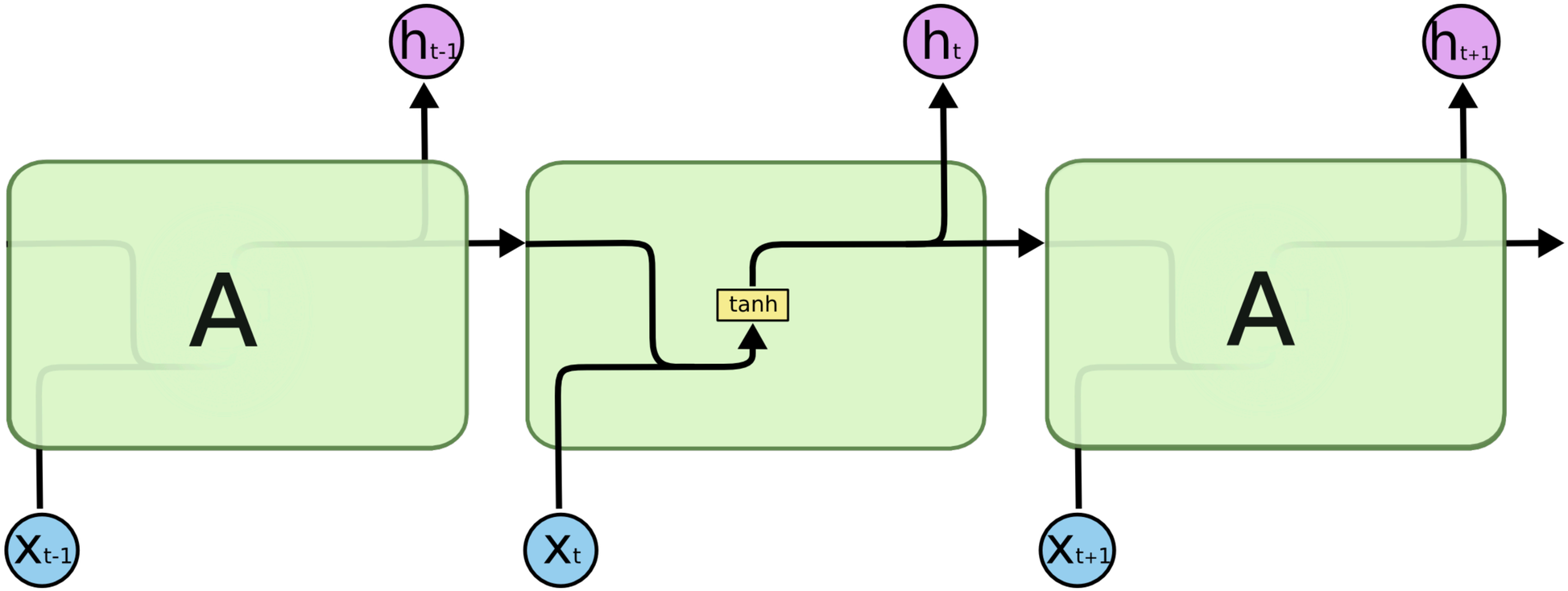
Three steps of BPTT:

1. Computing the output value of each neuron through forward propagation.
2. Computing the error value of each neuron through backpropagation δ_j.
3. Computing the gradient of each weight.

03

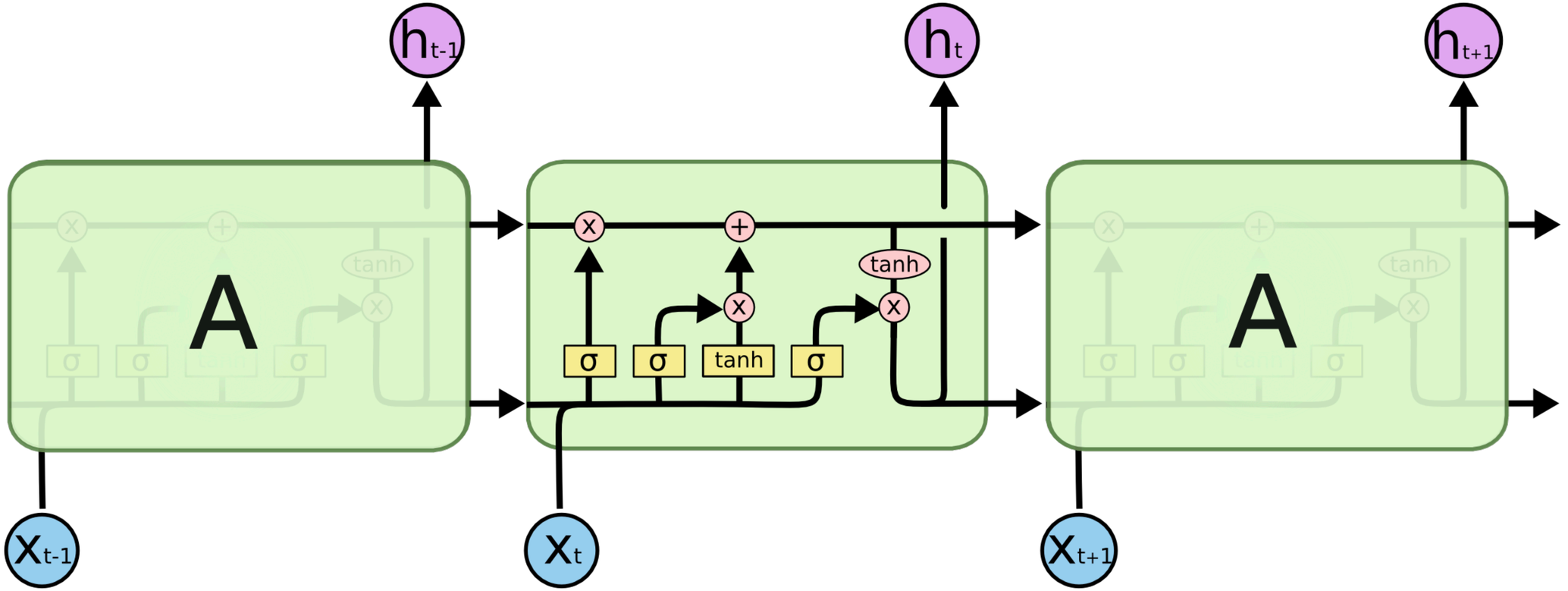
Long Short-term Memory Network

Standard RNNs



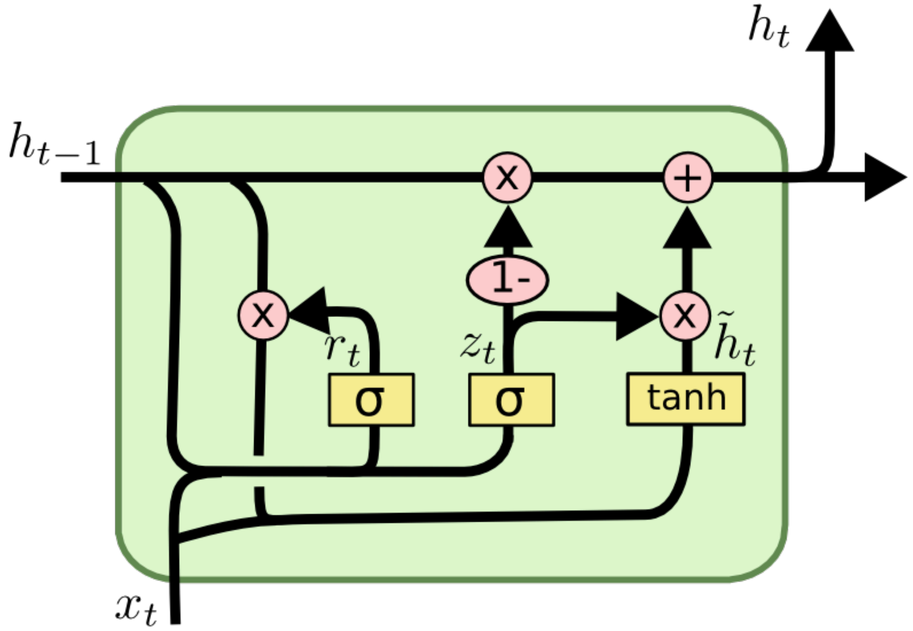
https://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTMs



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

Gated Recurrent Unit (GRU)



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

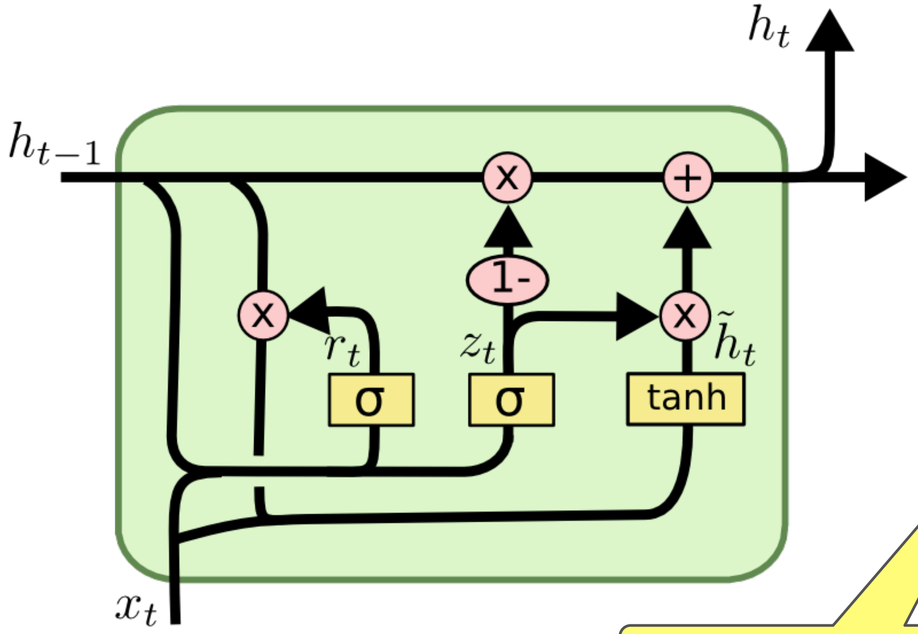
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be

Gated Recurrent Unit (GRU)



Update gate

$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

Reset gate

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Current memory content

Final memory at current time step

https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be

References

- Google. ImageCollection Visualization. https://developers.google.com/earth-engine/guides/ic_visualization. 2020, Accessed on Mar 2021.
- Aashish Chaubey. Sequential Data — and the Neural Network Conundrum! https://medium.com/analytics-vidhya/sequential-data-and-the-neural-network-conundrum-b2c005f8f865. 2020, Accessed on Mar 2021.
- HUAWEI. Deep Learning Overview. 2020, Accessed on Mar 2021.
- Christopher Olah. Understanding LSTM Networks. https://colah.github.io/posts/2015-08-Understanding-LSTMs/?source=post_page-----79e5eb8049c9----------. 2015, Accessed on Mar 2021.
- LECUN, Yann; BENGIO, Yoshua; HINTON, Geoffrey. Deep learning. nature, v. 521, n. 7553, p. 436-444, 2015.
- Andrej Karpathy. The Unreasonable Effectiveness of Recurrent Neural Networks. http://karpathy.github.io/2015/05/21/rnn-effectiveness/. 2015, Accessed on Mar 2021.
- Frank Yang. Implementing Backpropagation Through Time. https://discuss.pytorch.org/t/implementing-backpropagation-through-time/69765. 2020, Accessed on Mar 2021.
- Simeon Kostadinov. Understanding GRU Networks. https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be. 2017, Accessed on Mar 2021.

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Ceará

HUAWEI

Thank you for your attention!

🙏☺️

Prof. Me. Saulo A. F. Oliveira
saulo.oliveira@ifce.edu.br