Logistics:

- The Kaggle competition through this link (due 24/03 at 11:59)

https://www.kaggle.com/c/pistolas-vs-smartphones-con-deep-learning/

- Relevant Machine Learning notions/terminologies in:

https://sihamtabik.github.io/

- Assignment(optional): A DL classifier with data-augumentation for MNIST
- TFM proposals soon in PRADO

Today:

- RNNs and LSTM some theory
- Case study of RNNs
- Warming up examples with CNN and Tensorflow

Recurrent Neural Networks

Siham Tabik

siham@ugr.es

Outline

- Intro to RNNs
- How do RNNs work?
- Back propagation through time
- CNNs versus RNNs
- LSTMs
- LSTMs versus RNNs
- Case study

Recurrent Neural Network

RNNs work well in diverse applications:

- Predicting the next character in a word
- Predicting the next word in a sentence
- Language translation
- Speech recognition
- Action detection

https://www.youtube.com/watch?v=IIHKEs9m3WM

Recurrent Neural Network

The basic idea: Input are a sequence of elements, e.g.,

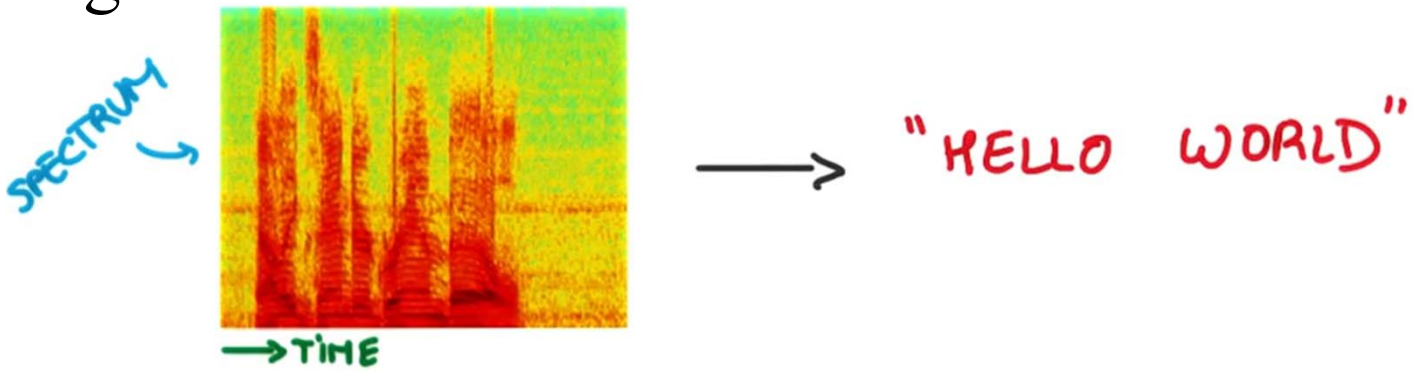
Sentence: predict the next word in a sentence

the	baby	is	?

Word: predict the next character in a word

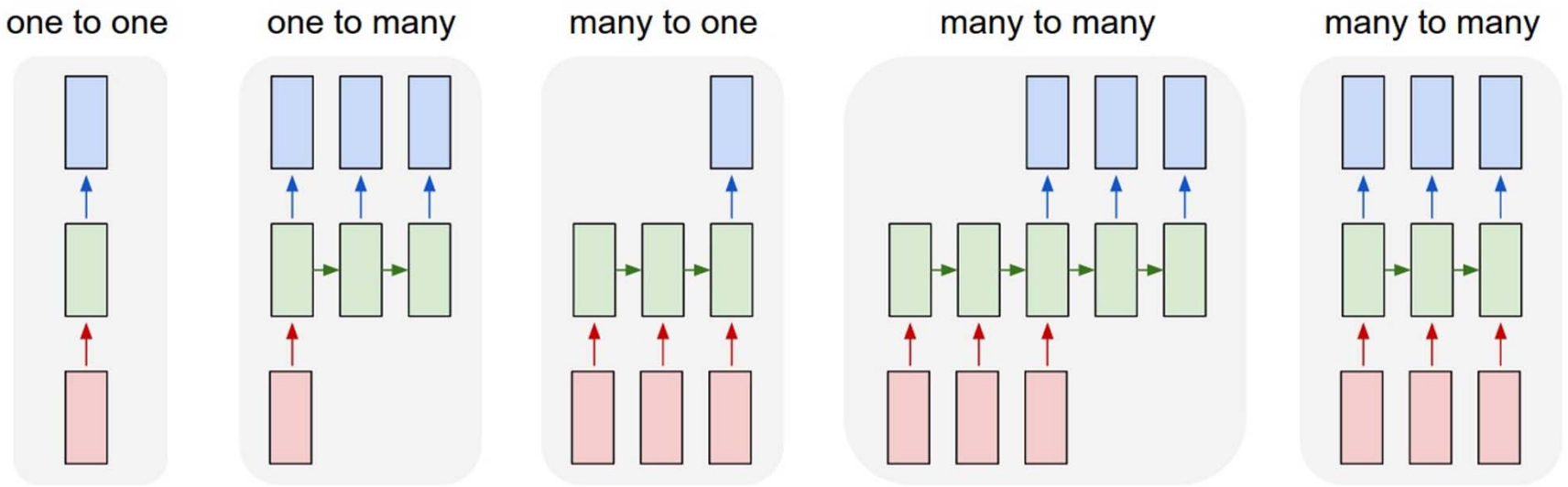
n	u	m	e	r	?	

Speech recognition



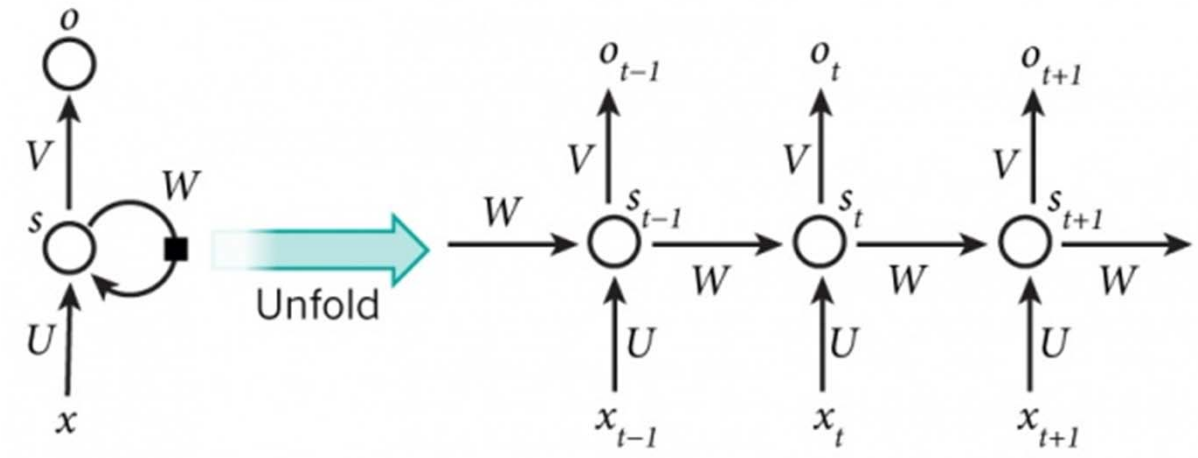
Recurrent Neural Network

RNNs are **versatile**, they allow us to operate over a sequence of elements and one or a sequence of output



How does RNN work?

- Given input $x(x_0, x_1, x_2, \dots)$ and output $o(o_1, o_2, o_3, \dots)$

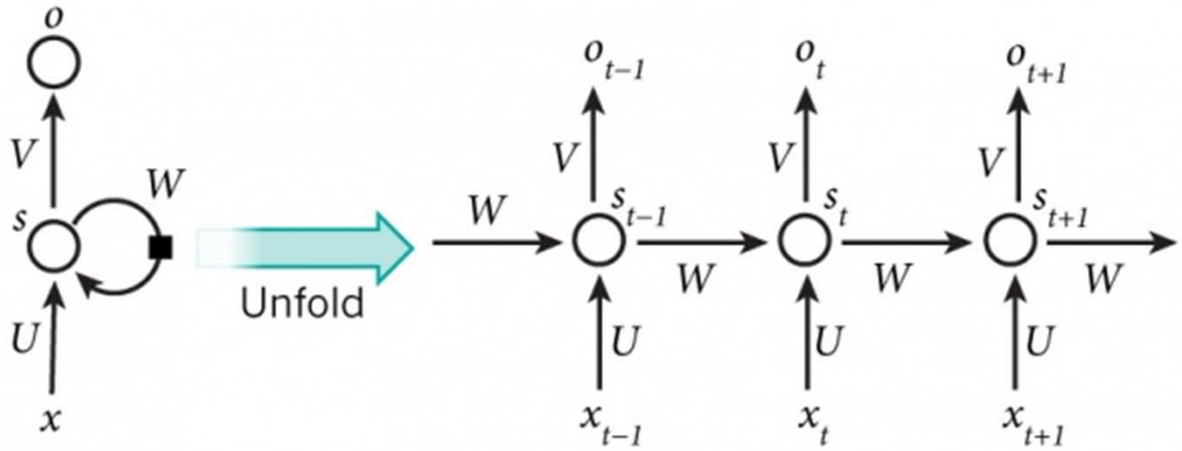


$$S_t = f(U \cdot x_t + W \cdot S_{t\_1})$$
$$O_t = V \cdot S_t$$

- f *is nonlinear function such as ReLU*
- S_t is the network's state vector
- U, W and V are matrices parameters

Back propagation through time

- The entire input sequence is considered as a single element of the training set
- The total error gradient is the sum of the error gradients at each instant of time



Problem: When the sequence is too long→ the vanishing or exploding gradient problem

CNNs models versus RNNs models

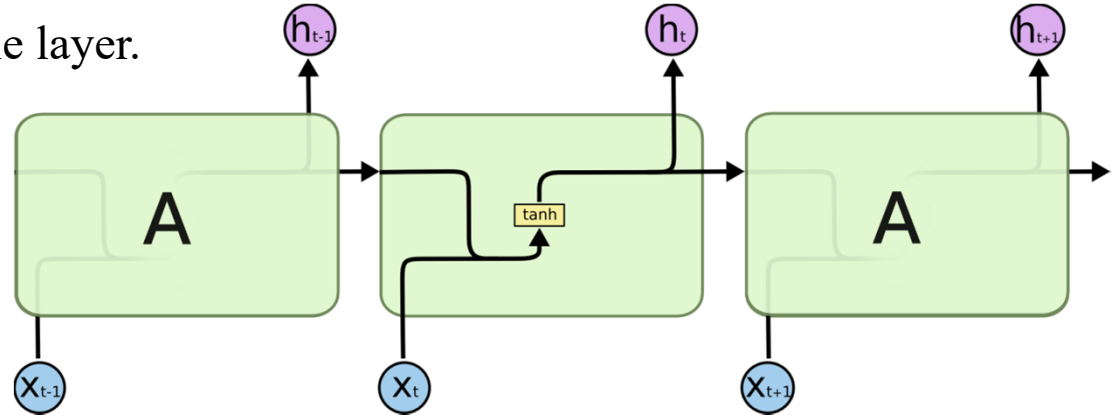
CNNs	RNNs
Example of one layer CNN $$y = f(W \cdot x + b)$$ f *is a nonlinear funtion*	$$S_t = f(U \cdot xt + W \cdot St_{-1})$$ $$O_t = V \cdot St$$ • f *is nonlinear function such as ReLU* • St is the network's state vector • U, W and V are matrices parameters
• learn to recognize patterns across space • looks for the same patterns on all the different subfields of the image • Learns different parameters in each level	• learn to recognize patterns across time • Do not look for the same patterns over the previous hidden layers • Use the same parameters for each instance of the sequence

Long Term Short Term Memory(LSTM)

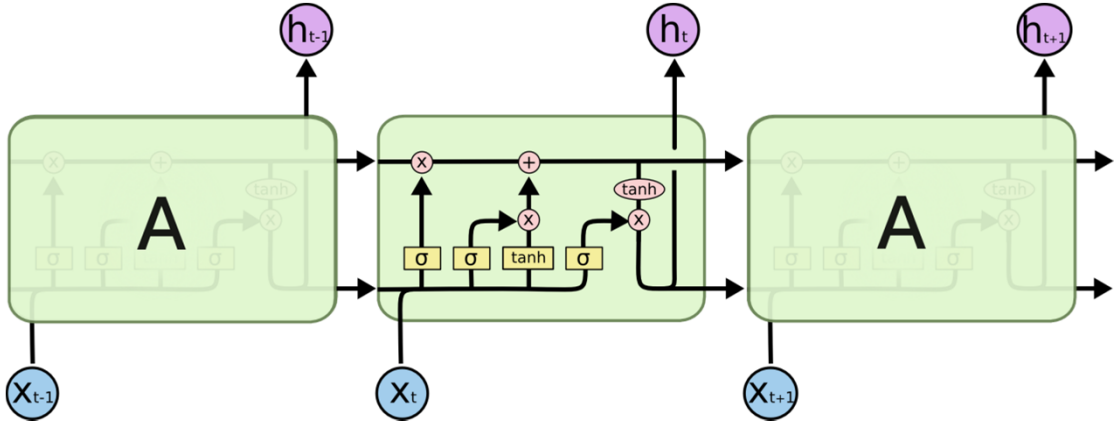
- LSTM are a special RNN architecture, originally conceived:
  - Long short term memory Sepp Hochreiter, Jurgen Schmidhuber Neural Computation, 9(8):1735-1780, 1997.
- Free from the problem of vanishing gradient and offers excellent results and performance
- Ideal for prediction and classification on long temporal sequences
- LSTM has the ability to forget irrelevant information and remember relevant information

RNNs versus LSTMs

The RNN cell is one single layer.



The LSTM cell is a combination of four layers, three sigmoid and one tanh layers

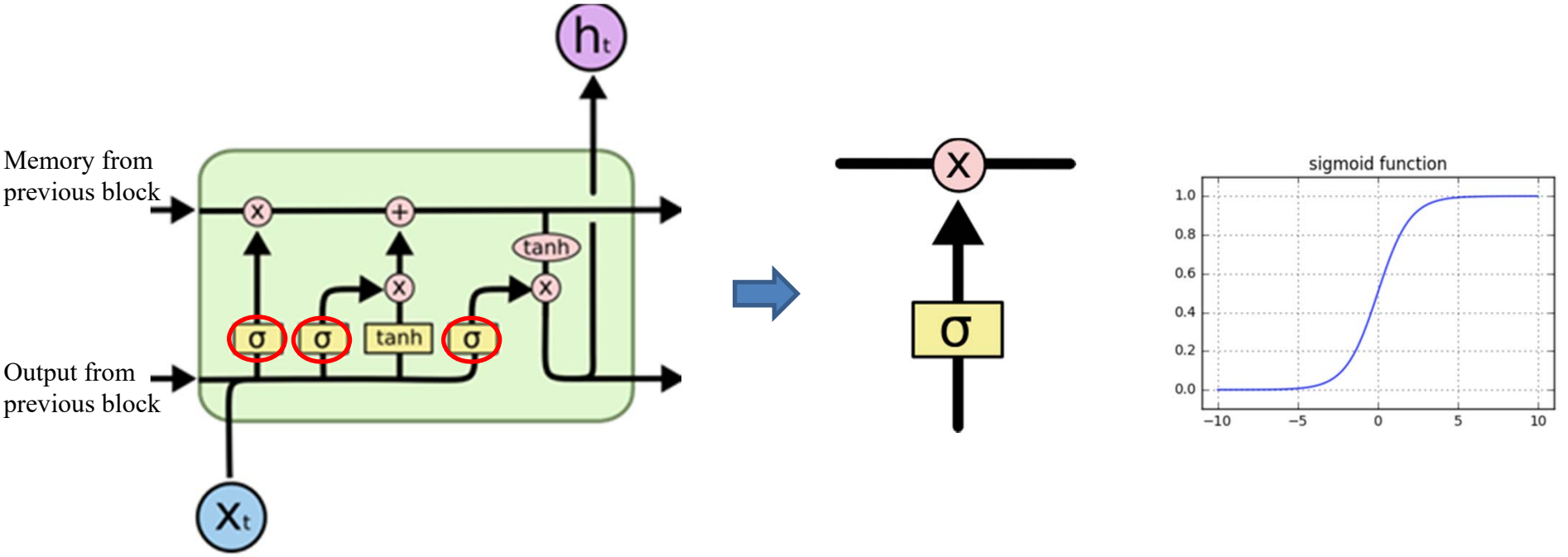


Source: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM Cell

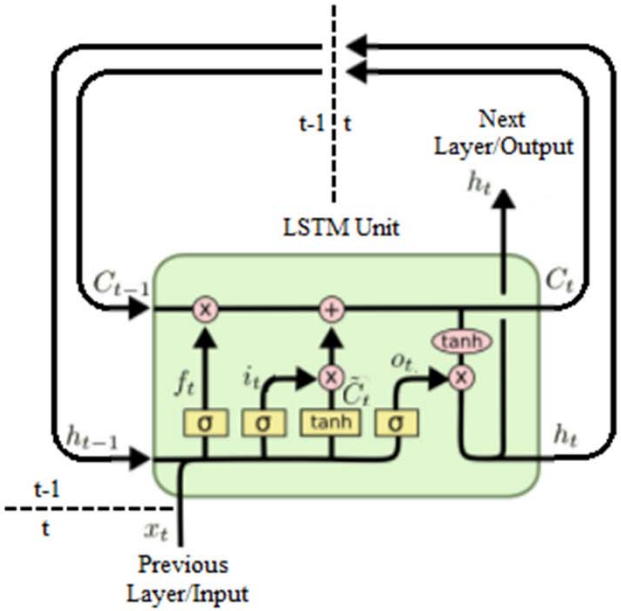
- LSTM has the ability to decide whether to remember or discard the information obtained from a given element of the sequence
- Three gates are governed by sigmoid units, outputs a value in [0,1],

To control the in and out information

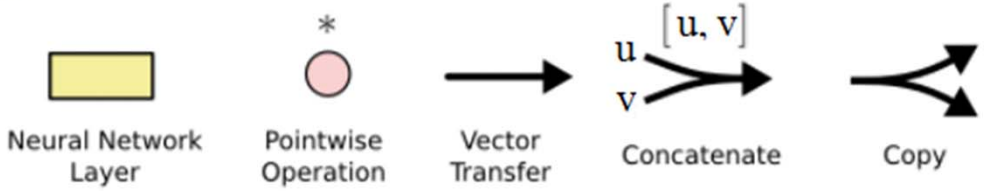


Source: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

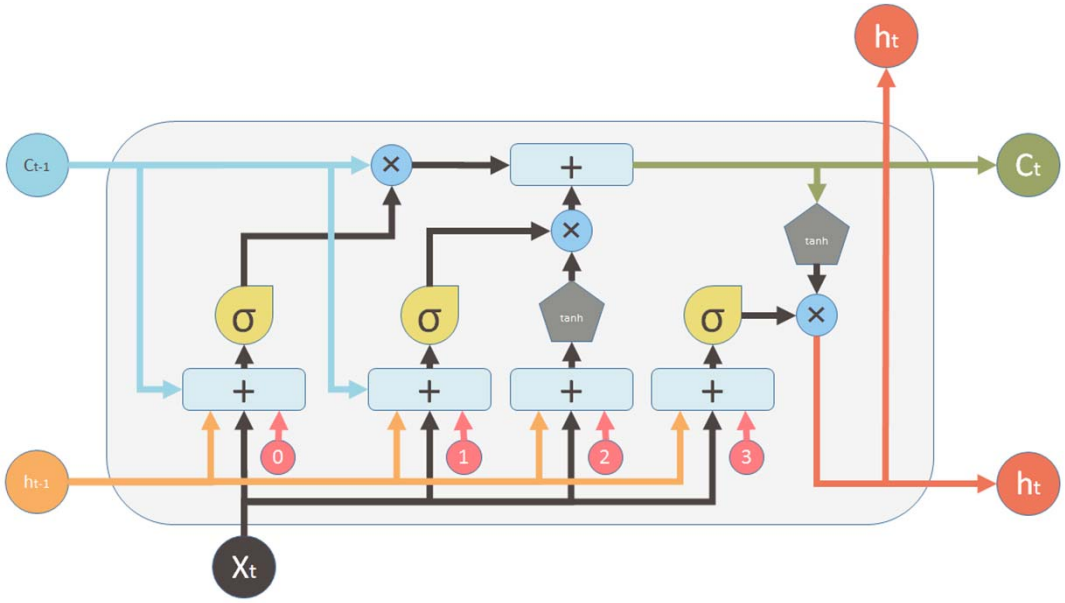
Understanding LSTM Networks



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$
$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$



LSTM Cell



Inputs:

- X_t Input vector
- C_{t-1} Memory from previous block
- h_{t-1} Output of previous block

outputs:

- C_t Memory from current block
- h_t Output of current block

Nonlinearities:

- σ Sigmoid
- tanh Hyperbolic tangent

Bias: 0

Vector operations:

- × Element-wise multiplication
- + Element-wise Summation / Concatenation