Deep Learning
(Recurrent Neural Networks)

Sadegh Eskandari
Department of Computer Science, University of Guilan
eskandari@guilan.ac.ir

Today …

- Introduction
- Unfolding
- RNN Types
- RNN Training
- LSTM

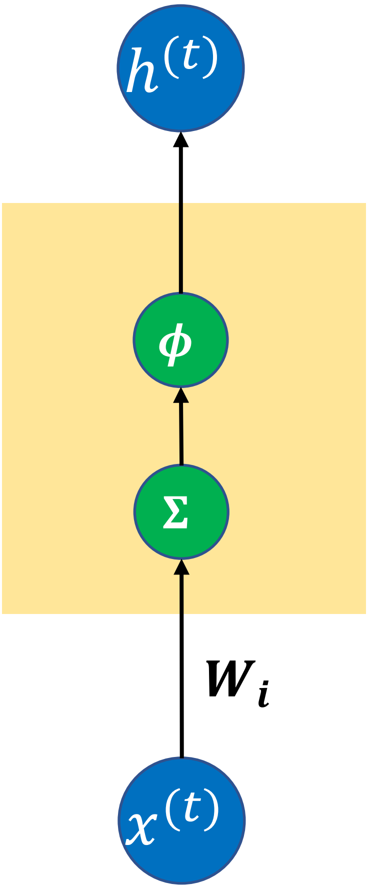
Introduction

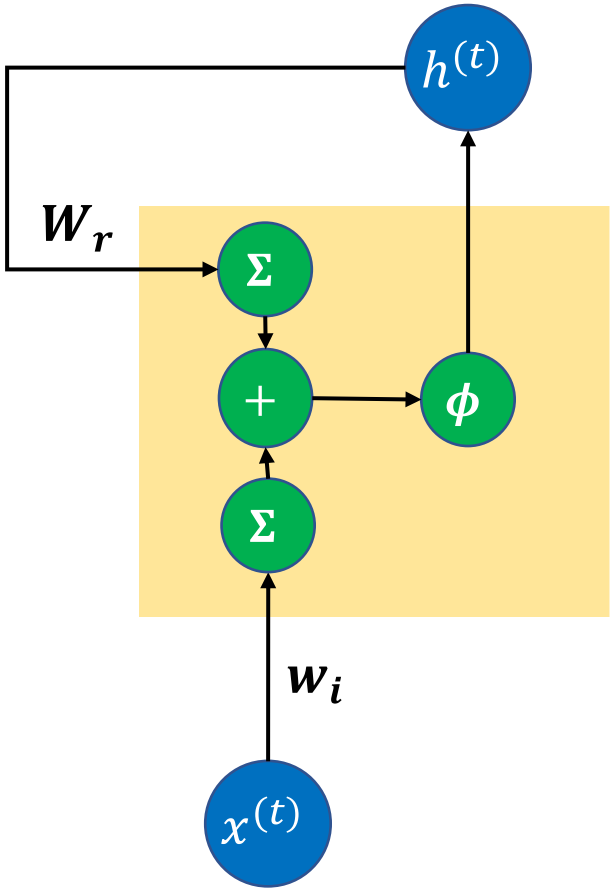
- **Recurrent neural networks (RNNs)**, are a family of neural networks for processing sequential data.
- **Examples:**
  - Speech-to-text and text-to-speech
  - Machine translation
  - Action recognition in video data
- We show the sequential data as $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \dots, \boldsymbol{x}^{(\tau)}$, where $\boldsymbol{x}^{(t)}, t \in \{1,2, \dots, \tau\}$, represents the input at time instance t

Introduction

- An MLP layer vs an RNN layer

MLP Layer





RNN Layer

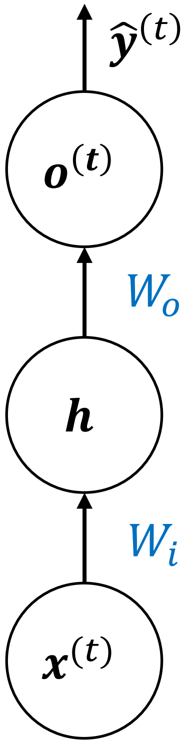
$$\boldsymbol{h}^{(t)} = \boldsymbol{\varphi}\left(\boldsymbol{w}_i \boldsymbol{x}^{(t)}\right)$$

$$\boldsymbol{h}^{(t)} = \boldsymbol{\varphi}\left(\boldsymbol{w}_i \boldsymbol{x}^{(t)} + \boldsymbol{w}_r \boldsymbol{h}^{(t-1)}\right)$$

Introduction

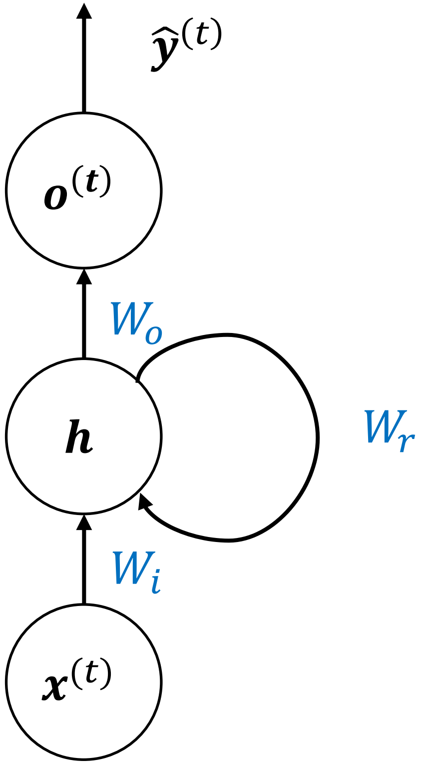
- We can think of an RNN as a neural network with hidden units which feed into themselves (self-loops).

MLP



$$\boldsymbol{h}^{(t)} = ReLU\left(\boldsymbol{W}_i \boldsymbol{x}^{(t)}\right)$$
$$\boldsymbol{o}^{(t)} = \boldsymbol{W}_o \boldsymbol{h}^{(t)}$$
$$\hat{\boldsymbol{y}}^{(t)} = \mathrm{softmax}\left(\boldsymbol{o}^{(t)}\right)$$

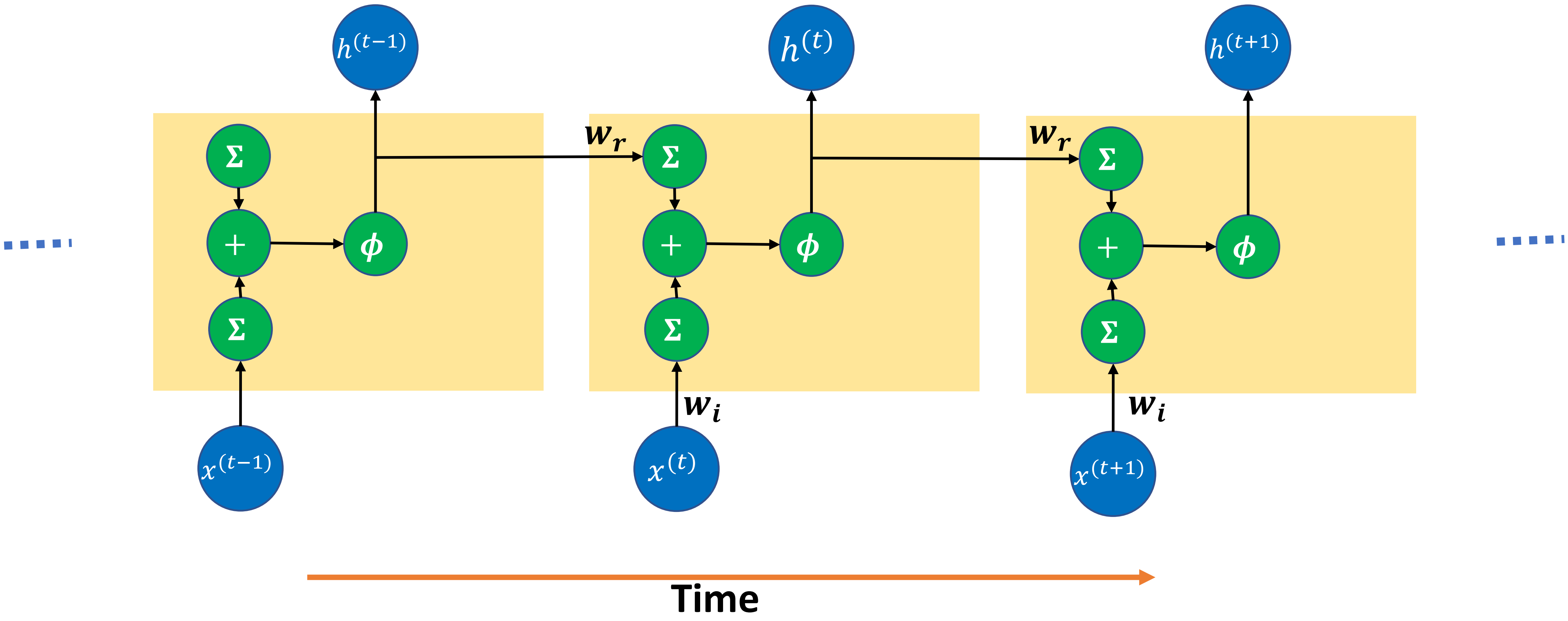
RNN



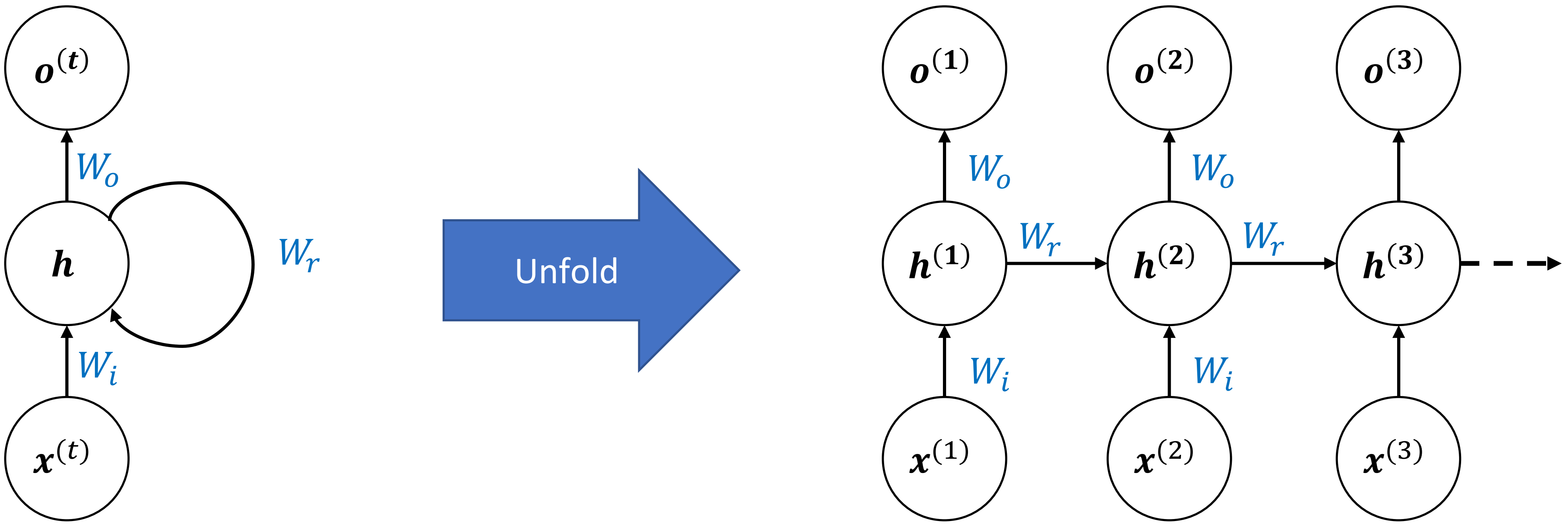
$$\boldsymbol{h}^{(t)} = ReLU\left(\boldsymbol{W}_r \boldsymbol{h}^{(t-1)} + \boldsymbol{W}_i \boldsymbol{x}^{(t)}\right)$$
$$\boldsymbol{o}^{(t)} = \boldsymbol{W}_o \boldsymbol{h}^{(t)}$$
$$\hat{\boldsymbol{y}}^{(t)} = \mathrm{softmax}\left(\boldsymbol{o}^{(t)}\right)$$

Unfolding

- We can unfold the RNN's graph by explicitly representing the units at all time steps.
- The weights are shared between all time steps

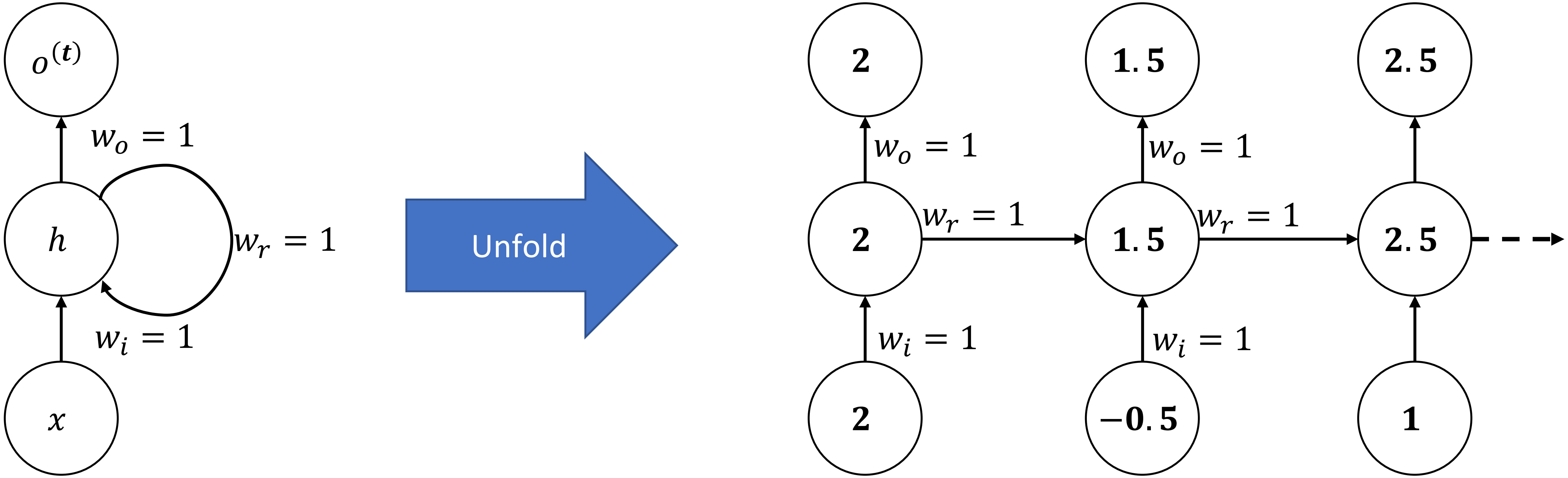


Unfolding



Unfolding

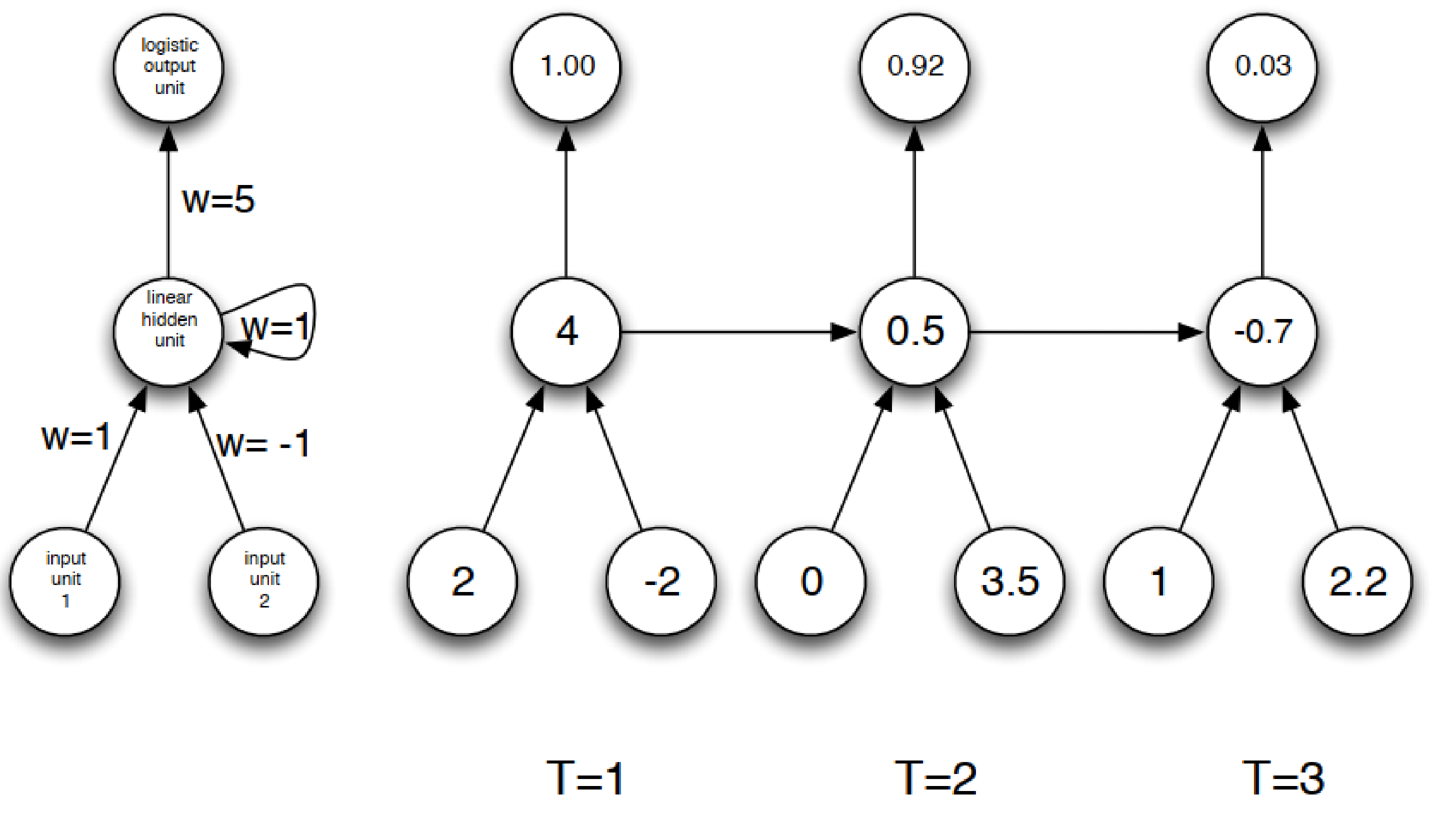
- A simple example: an RNN to sum its inputs



Example from Neural Networks and Deep Learning course by Jimmy Ba, 2020, University of Toronto: https://csc413-2020.github.io/

Unfolding

- An other example: an RNN that determines if the total values of the first or second input are larger



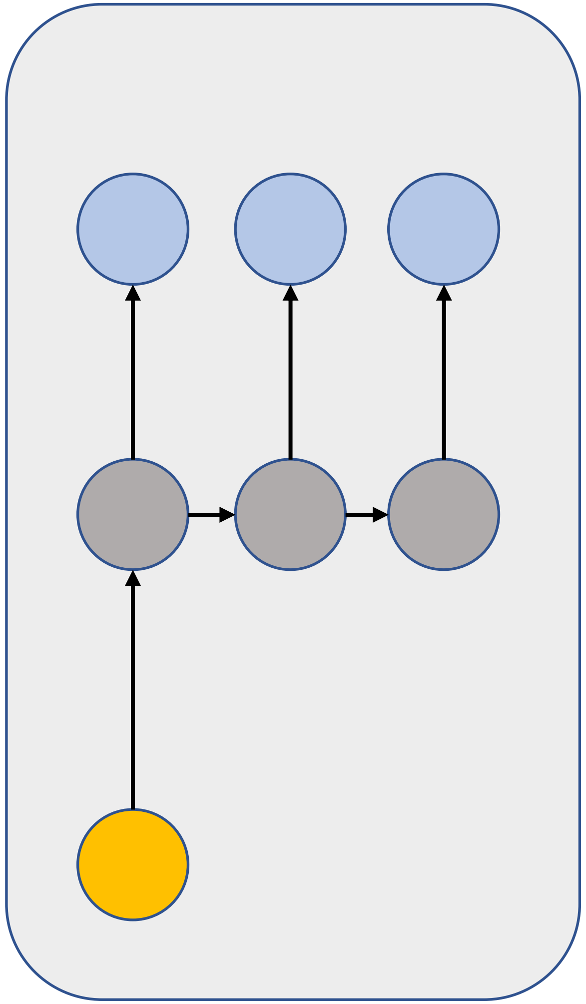
Logistic Function:

$$L(z) = \frac{1}{1 + e^{-z}}$$

Example from Neural Networks and Deep Learning course by Jimmy Ba, 2020, University of Toronto: https://csc413-2020.github.io/

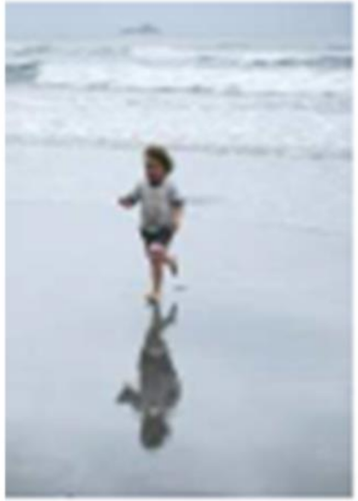
RNN: Sequence Processing Types

One to Many



Example: Image captioning

Image => sequence of words



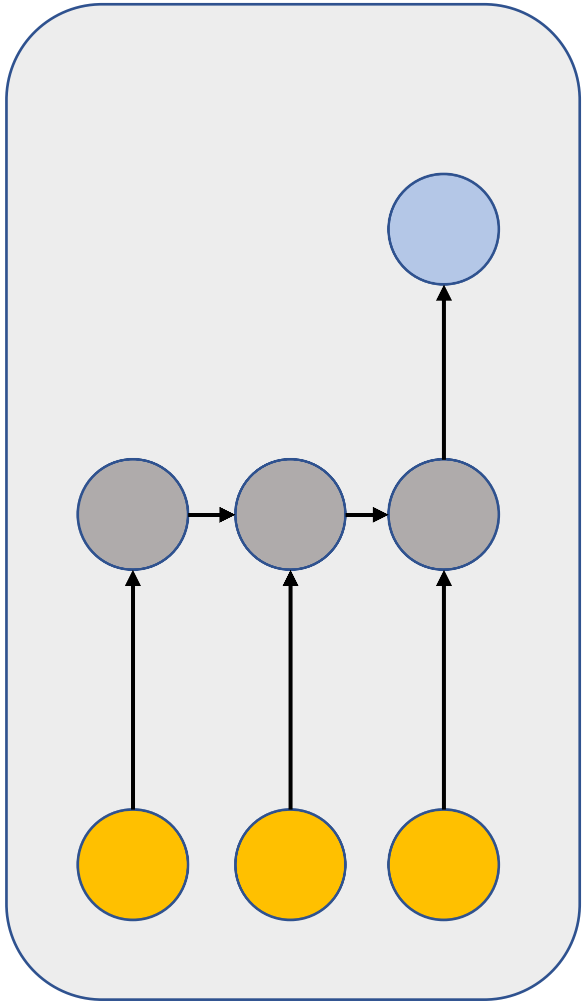
Ground Truth Caption: A little boy runs away from the approaching waves of the ocean.

Generated Caption: A young boy is running on the beach.

Source: Hossain, M.Z., Sohel, F., Shiratuddin, M.F. and Laga, H., 2019. A comprehensive survey of deep learning for image captioning. ACM Computing Surveys (CsUR), 51(6), pp.1-36.

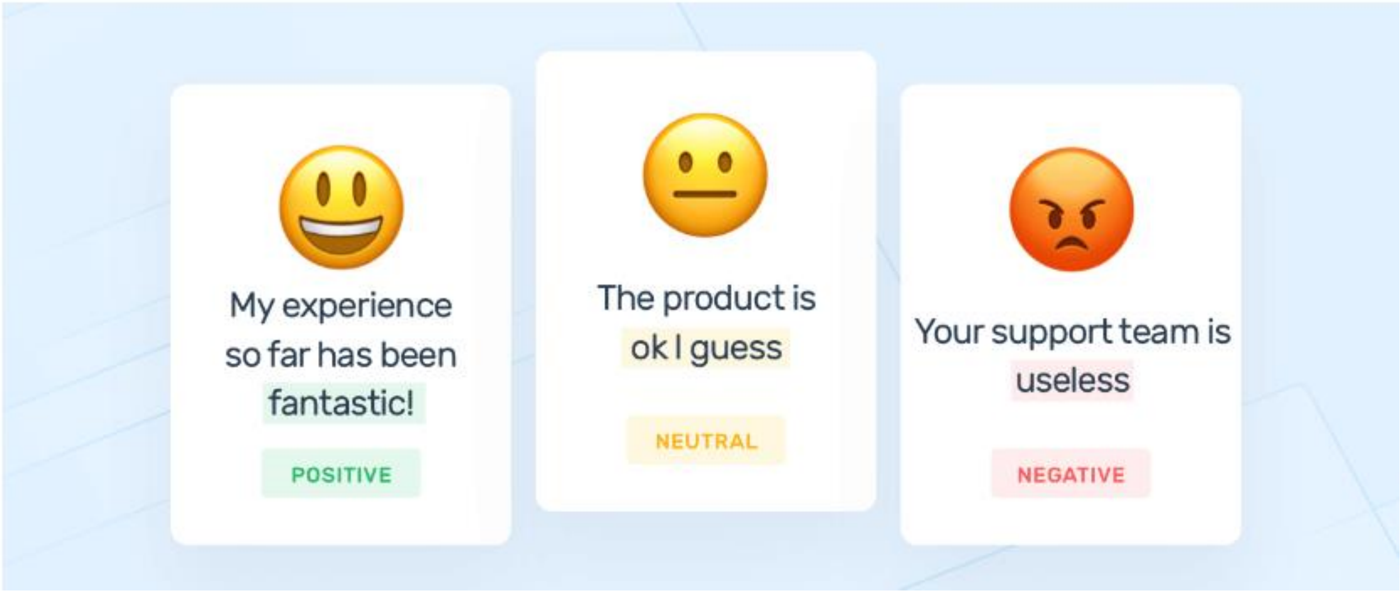
RNN: Sequence Processing Types

Many to One



Example: Sentiment Classification

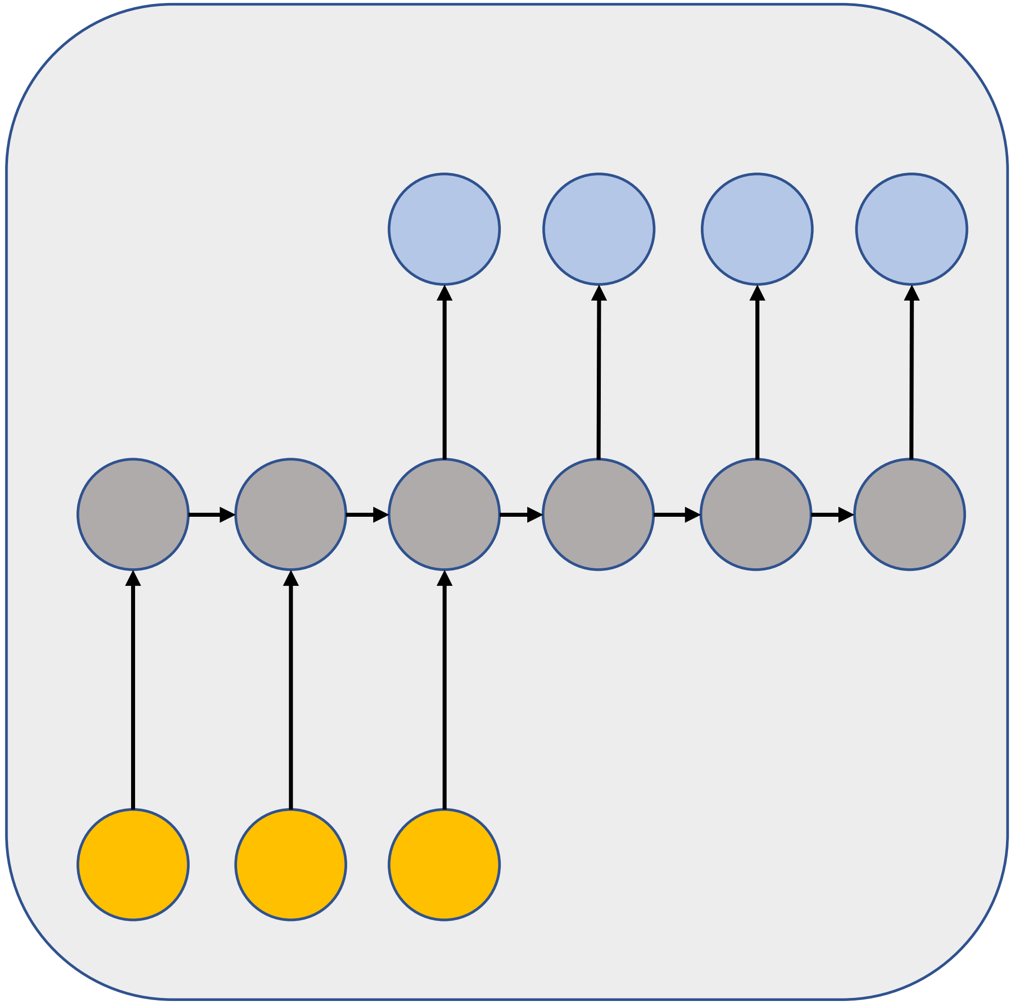
Sequence of words => Sentiment



Source: https://monkeylearn.com/sentiment-analysis/

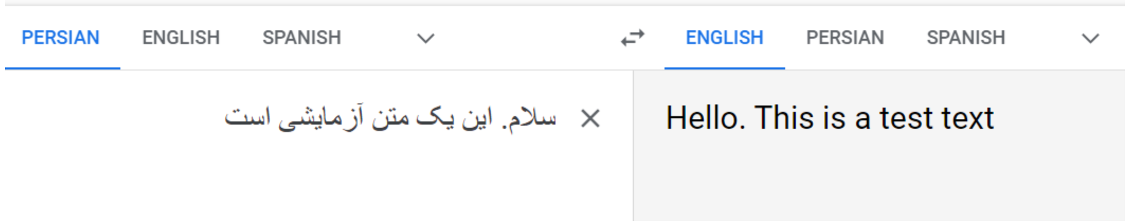
RNN: Sequence Processing Types

Many to Many

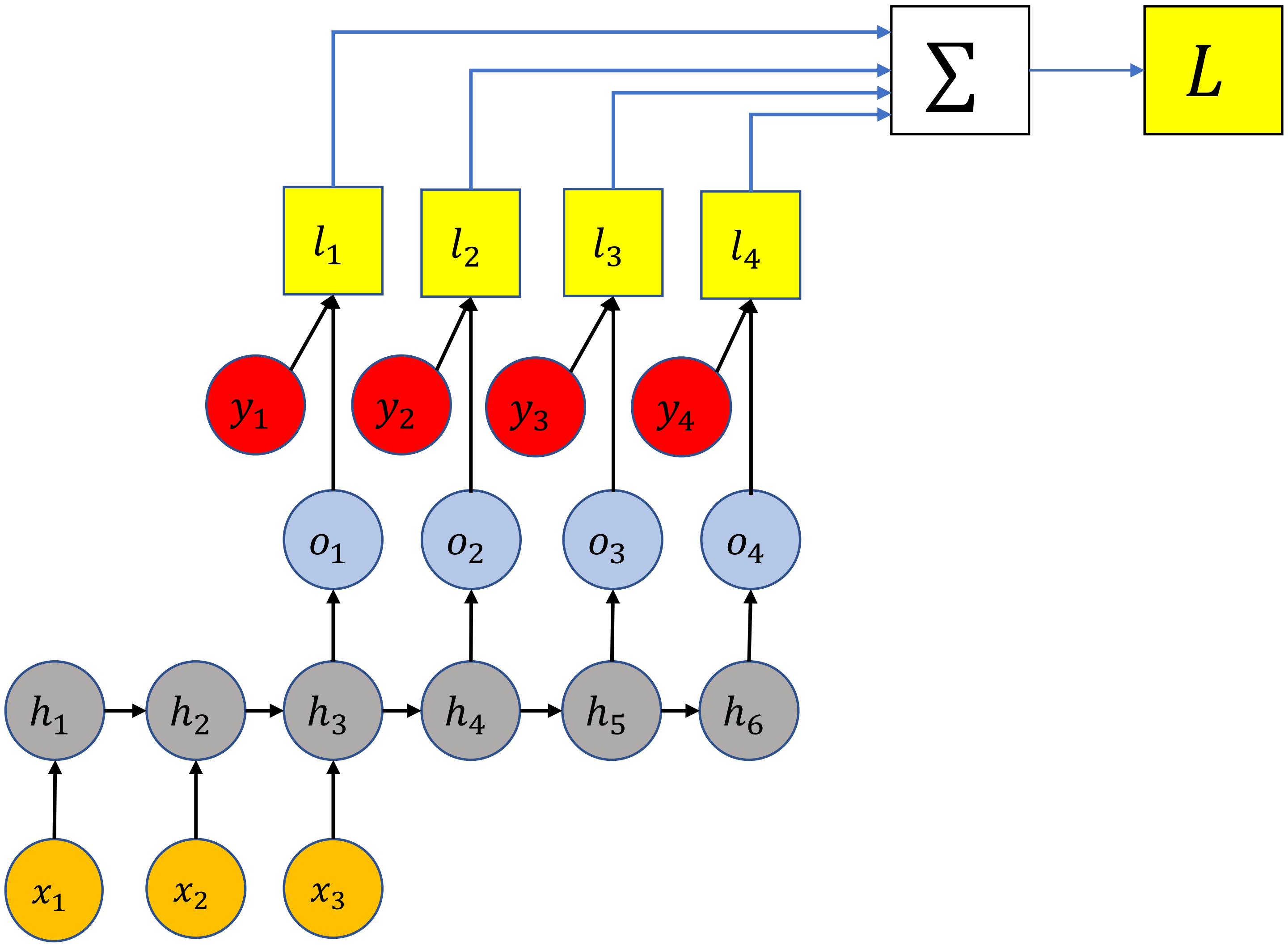


Example: Machine Translation, Language Model

Sequence of words => Sequence of words

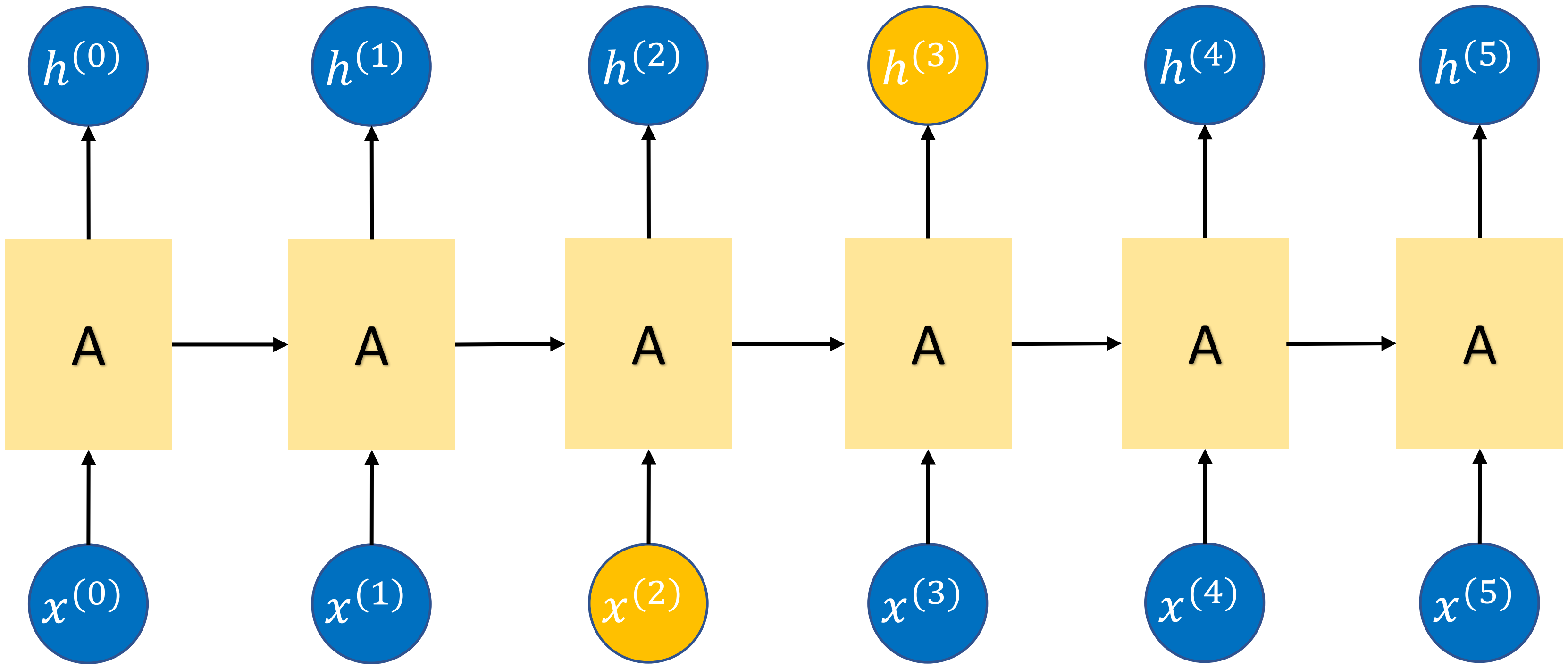


RNN: Training



RNN: The Problem of Long-Term Dependencies

- As stated, RNNs are able to connect previous information to the present task
- Simple RNNs work well when the gap between the relevant information is small



RNN: The Problem of Long-Term Dependencies

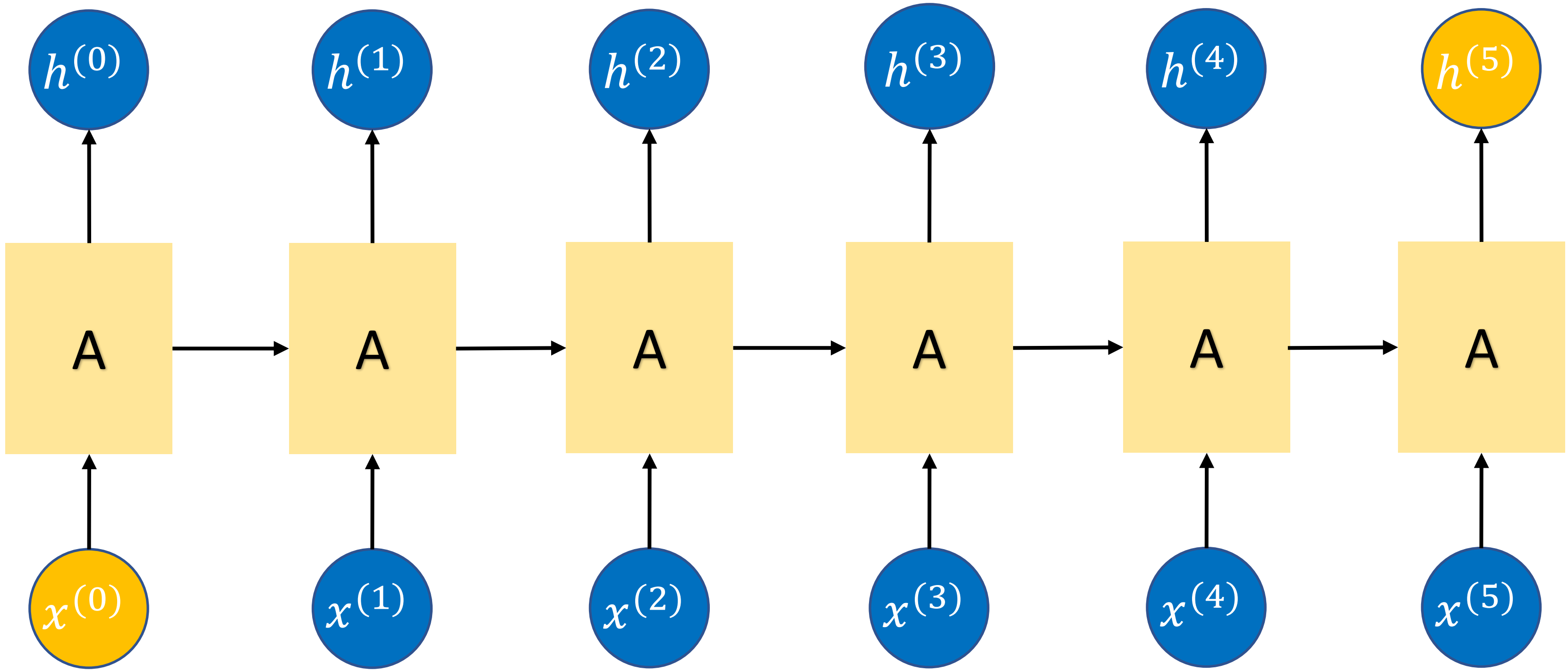
- As stated, RNNs are able to connect previous information to the present task
- Simple RNNs work well when the gap between the relevant information is small
- Example:

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**.

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

RNN: The Problem of Long-Term Dependencies

- Unfortunately, as that gap grows, RNNs become unable to learn to connect the information.



RNN: The Problem of Long-Term Dependencies

- Unfortunately, as that gap grows, RNNs become unable to learn to connect the information.

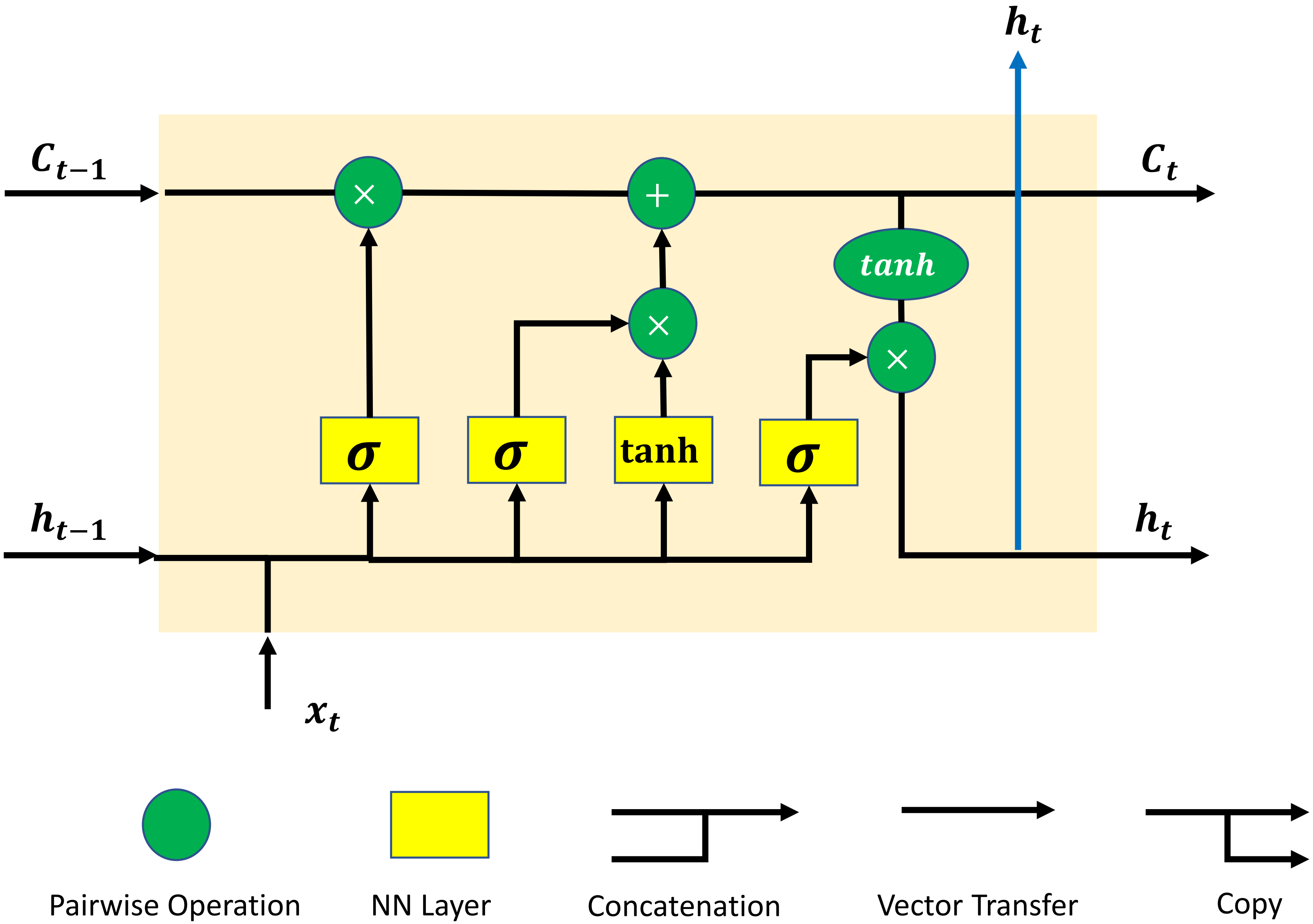
- Example:

  If we are trying to predict the last word in **“I live in Iran, a beautiful country in the middle east. My country has a population of eighty millions and its official language is …,”** 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 Iran, from further back.

Long Short Term Memory

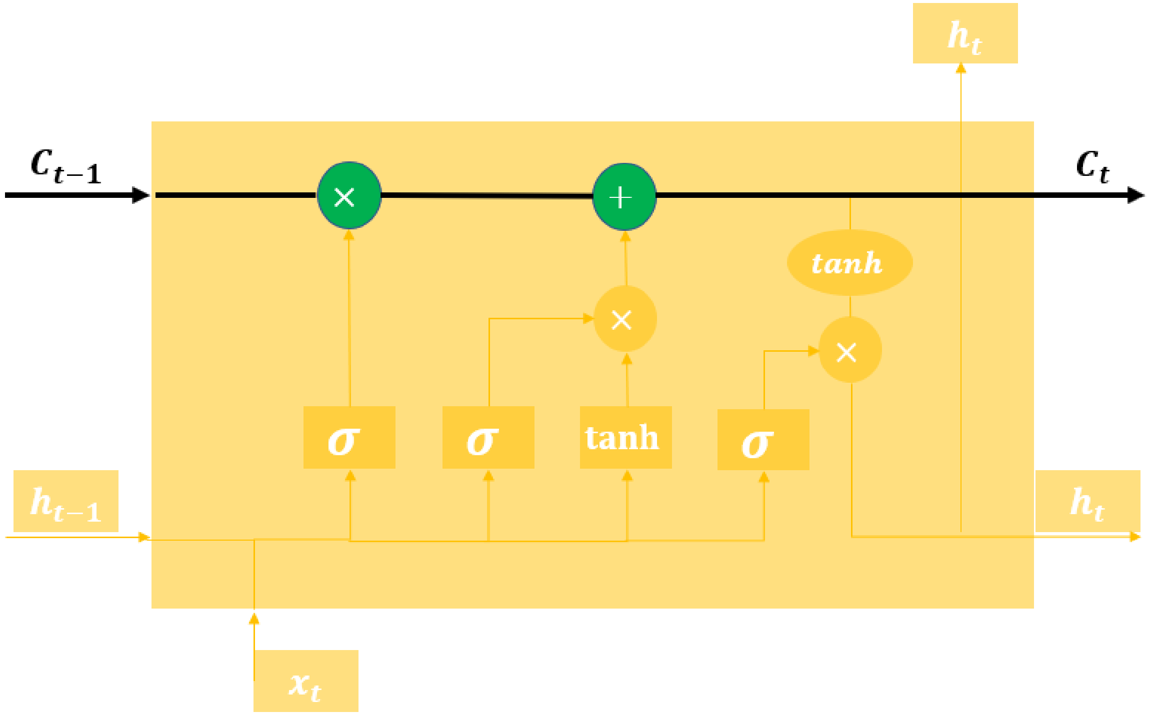
- **Long Short Term Memory (LSTM) n**etworks are a special kind of RNN, capable of learning long-term dependencies.

Long Short Term Memory



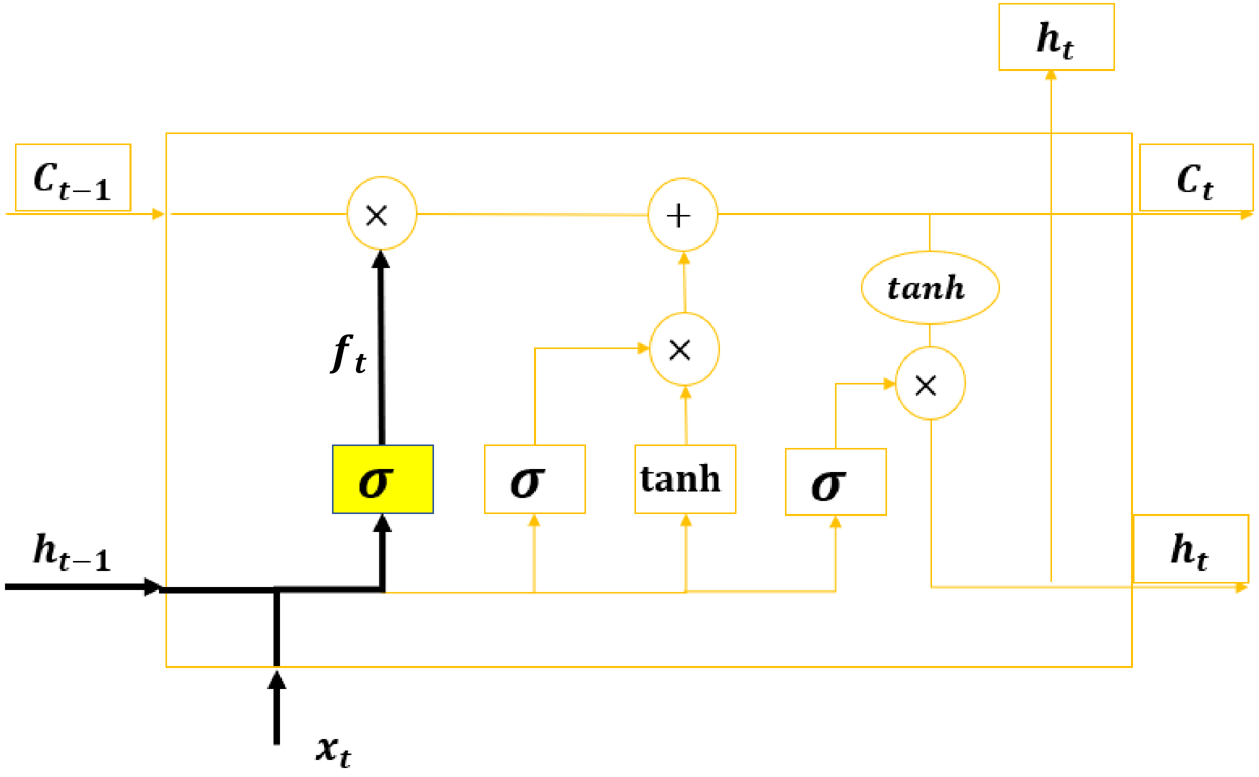
Long Short Term Memory

- The key to LSTMs is the cell state (C_t)



Long Short Term Memory

- **Forget Gate Layer**

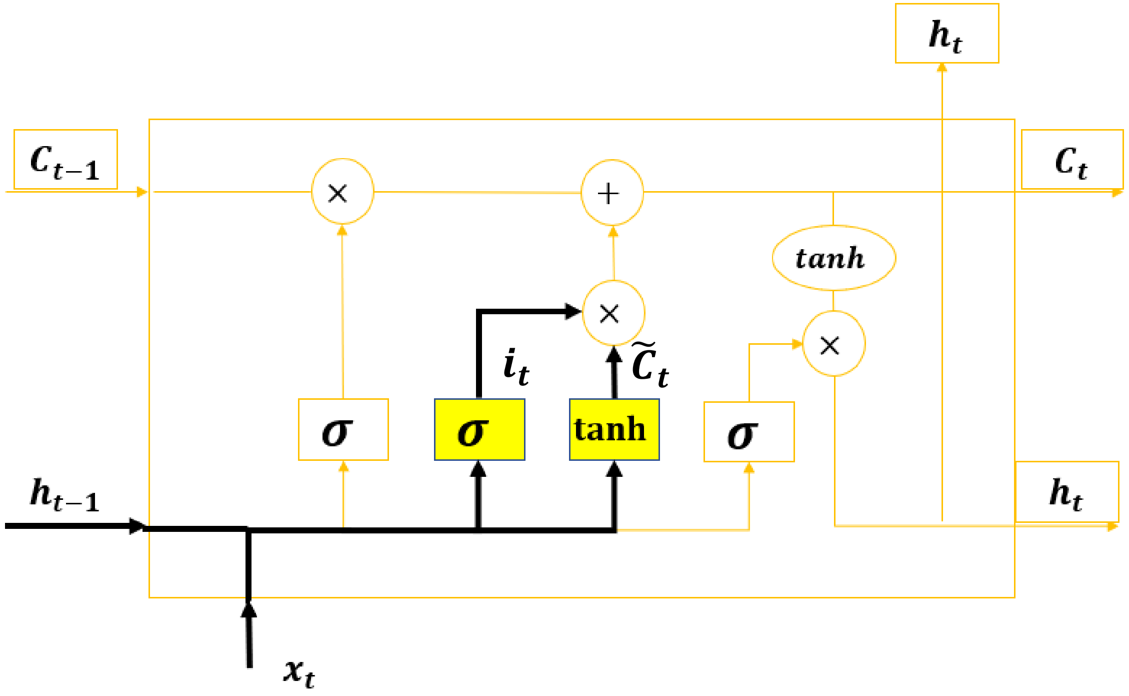


$$f_t = \sigma\left(W_f\ [h_{t-1}, x_t]\ \right)$$

$f_t = \bar{1}$ represents "completely keep C_{t-1}" while a $f_t = \bar{0}$ represents "completely forget C_{t-1}."

Long Short Term Memory

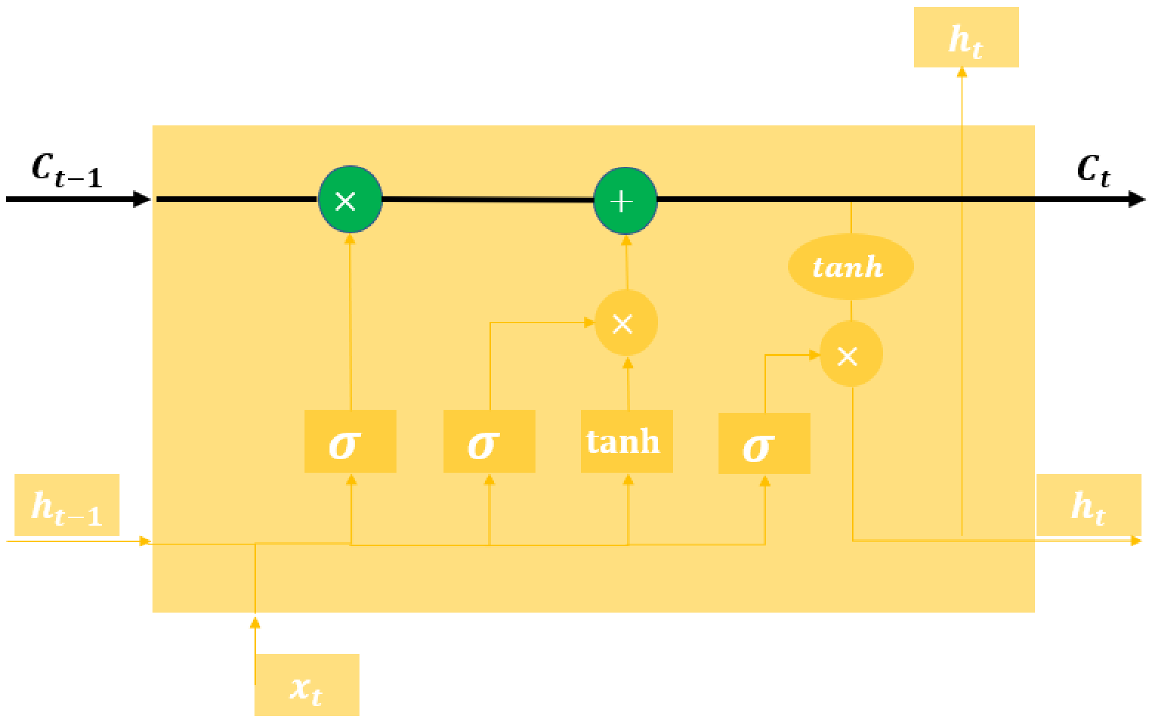
- **Input Gate Layer**: what new information we're going to store in the cell state



$$i_t = \sigma(W_i\,[h_{t-1}, x_t])$$

$$\widetilde{C}_t = \sigma(W_C\,[h_{t-1}, x_t])$$

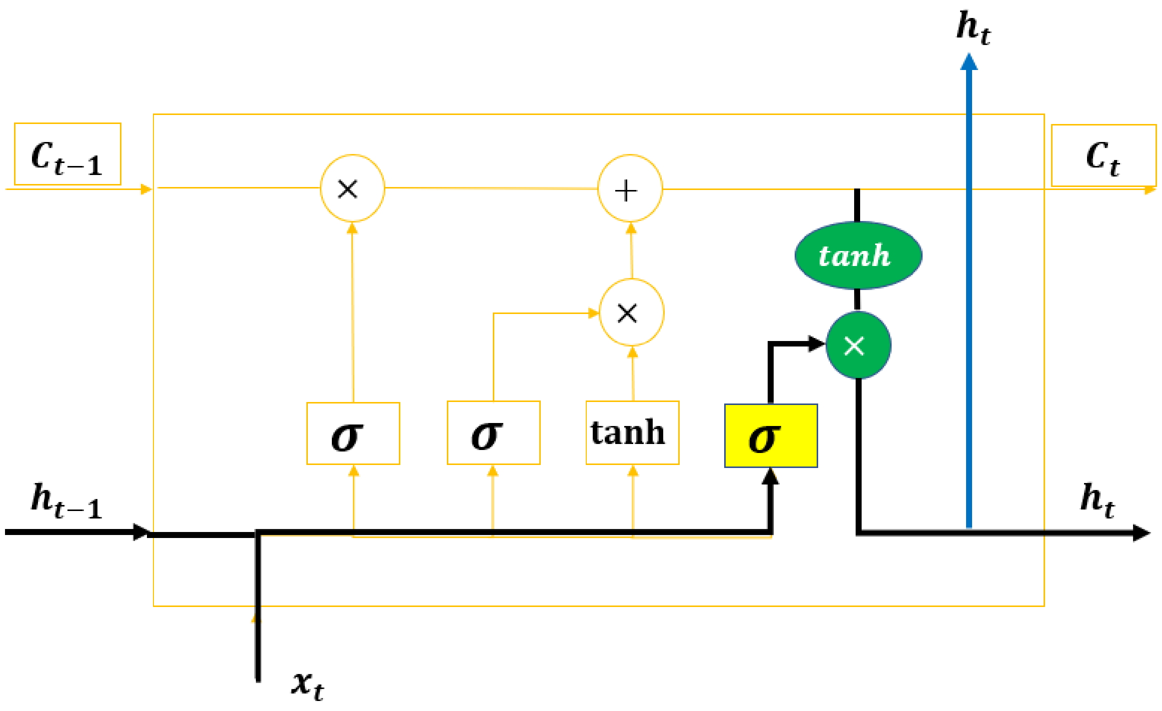
Long Short Term Memory



$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t$$

Long Short Term Memory

- **Output Gate Layer**



$$h_t = \sigma(W_o\ [h_{t-1}, x_t]\ ) * \tanh(C_t)$$