Long short-term Memory (LSTM)

Mathematical formulation

- Suppose that there are h hidden units, a batch size n, and d inputs
- $\boldsymbol{X}_t \in \mathbb{R}^{n \times d}$ and $\boldsymbol{H}_{t-1} \in \mathbb{R}^{n \times h}$
- We get the following gate values

$$\begin{aligned}
\boldsymbol{I}_t &= \sigma(\boldsymbol{X}_t \boldsymbol{W}_{xi} + \boldsymbol{H}_{t-1} \boldsymbol{W}_{hi} + \boldsymbol{b}_i) \\
\boldsymbol{F}_t &= \sigma(\boldsymbol{X}_t \boldsymbol{W}_{xf} + \boldsymbol{H}_{t-1} \boldsymbol{W}_{hf} + \boldsymbol{b}_f) \\
\boldsymbol{O}_t &= \sigma(\boldsymbol{X}_t \boldsymbol{W}_{xo} + \boldsymbol{H}_{t-1} \boldsymbol{W}_{ho} + \boldsymbol{b}_o)
\end{aligned}$$

- $\boldsymbol{W}_{\boldsymbol{x}.} \in \mathbb{R}^{d \times h}$
- $\boldsymbol{W}_{\boldsymbol{h}.} \in \mathbb{R}^{h \times h}$
- $\boldsymbol{I}_t, \boldsymbol{F}_t, \boldsymbol{O}_t \in \mathbb{R}^{n \times h}$

$\boldsymbol{\sigma}$ = Sigmoid

$\odot$ = element-wise product

Memory cell state

- Let us define the memory cell state $\boldsymbol{C}_t \in \mathbb{R}^{n \times h}$ for timestep t

$$\boldsymbol{C}_t = \boldsymbol{F}_t \odot \boldsymbol{C}_{t-1} + \boldsymbol{I}_t \odot \tilde{\boldsymbol{C}}_t$$

- $\boldsymbol{F}_t$ addresses how much of the old cell internal state $\boldsymbol{C}_{t-1}$ we retain
- $\boldsymbol{I}_t$ governs how much we take new data into account via $\tilde{\boldsymbol{C}}_t$
- If $\boldsymbol{F}_t = \boldsymbol{1}$ and $\boldsymbol{I}_t = \boldsymbol{0}$ the memory cell remains constant ($\boldsymbol{C}_t = \boldsymbol{C}_{t-1}$)

Input node

- $\tilde{\boldsymbol{C}}_t \in \mathbb{R}^{n \times h}$
- Its computation is equivalent to the one of three gates but with a $\tanh$ activation

$$\tilde{\boldsymbol{C}}_t = \tanh(\boldsymbol{X}_t \boldsymbol{W}_{xc} + \boldsymbol{H}_{t-1} \boldsymbol{W}_{hc} + \boldsymbol{b}_c)$$

Link with the input gate

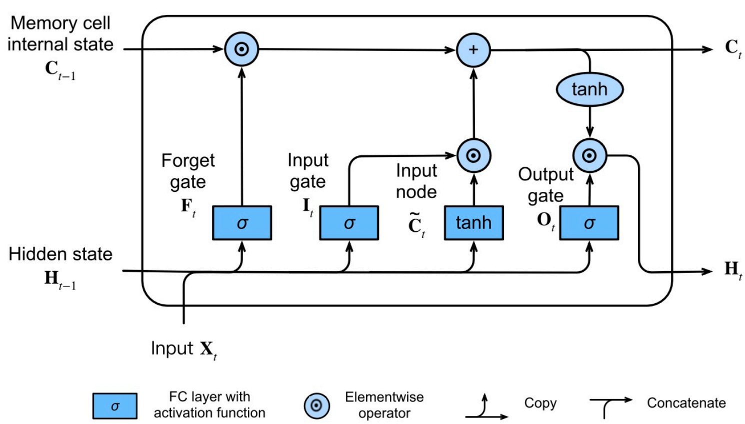
- The value of the input node interacts with the input gate to decide what should be added to the current **internal state**

Output gate and internal state

- Finally, we have to define the output $\boldsymbol{H}_t$ of the memory cell using both $\boldsymbol{O}_t$ and $\boldsymbol{C}_t$

$$\boldsymbol{H}_t = \boldsymbol{O}_t \odot \tanh(\boldsymbol{C}_t)$$

- When $\boldsymbol{O}_t$ is close to 0, current memory does not impact the subsequent layer of the network
- When $\boldsymbol{O}_t$ is close to 1, current memory adds information to the next layer



Gated Recurrent Units (GRU)

$$\boldsymbol{R}_t = \sigma(\boldsymbol{X}_t \boldsymbol{W}_{xr} + \boldsymbol{H}_{t-1} \boldsymbol{W}_{hr} + \boldsymbol{b}_r)$$
$$\boldsymbol{Z}_t = \sigma(\boldsymbol{X}_t \boldsymbol{W}_{xz} + \boldsymbol{H}_{t-1} \boldsymbol{W}_{hz} + \boldsymbol{b}_z)$$

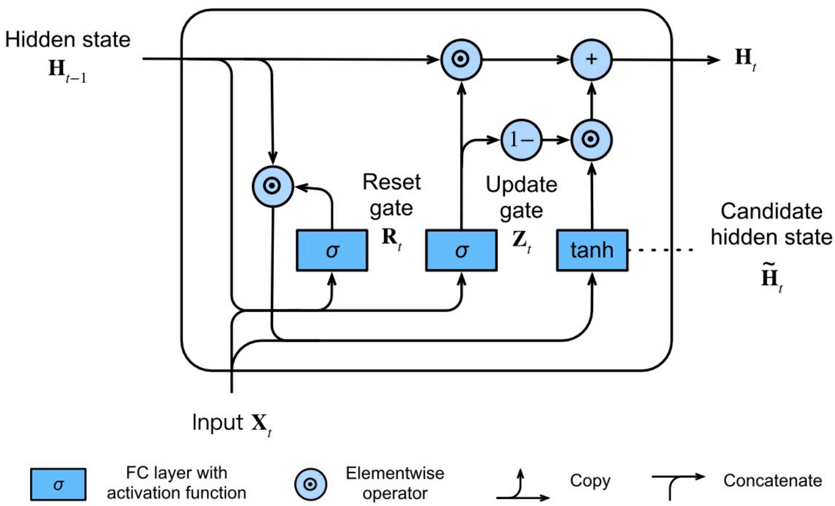
Candidate Hidden State

$$\tilde{\boldsymbol{H}}_t = \tanh(\boldsymbol{X}_t \boldsymbol{W}_{xh} + (\boldsymbol{R}_t \odot \boldsymbol{H}_{t-1}) \boldsymbol{W}_{hh} + \boldsymbol{b}_h)$$

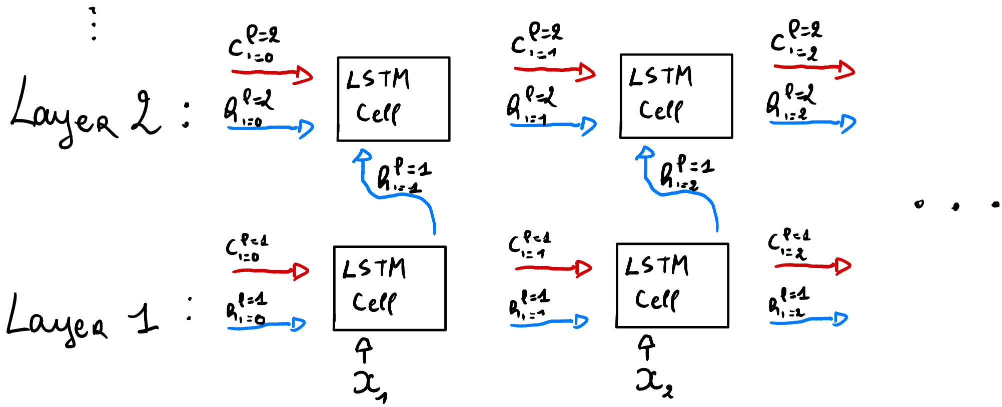
Hidden State

- Finally, we incorporate the effect of the update gate $\boldsymbol{Z}_t$ on the hidden state $\boldsymbol{H}_t$

$$\boldsymbol{H}_t = \boldsymbol{Z}_t \odot \boldsymbol{H}_{t-1} + (1 - \boldsymbol{Z}_t) \odot \tilde{\boldsymbol{H}}_t$$

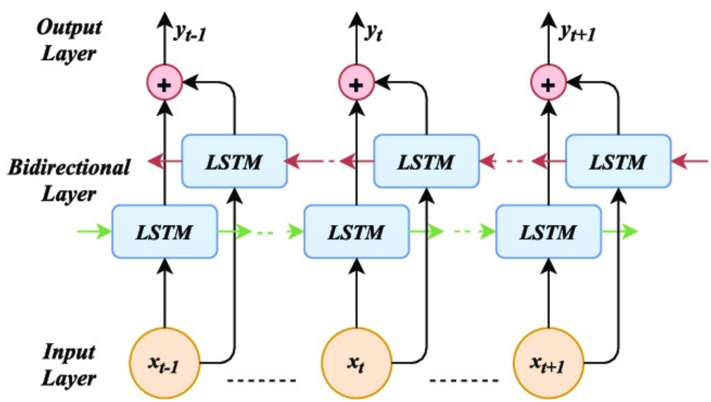


Stacking recurrent cells



(Même principe pour GRU)

Bi-directional RNN :



Convolutional RNN :

All matrix products are replaced by convolultional operations.
Inputs, hiddens states, memory states are not of shape n x d but n x channels x height x width .