LSTM

Lecture 22-23

Long Short-Term Memory (LSTM)

- A type of RNN proposed by Hochreiter and Schmidhuber in 1997 as a solution to the vanishing gradients problem.

- On step *t*, there is a hidden state $\boldsymbol{h}^{(t)}$ *and* a cell state $\boldsymbol{c}^{(t)}$
  - Both are vectors length *n*
  - The cell stores long-term information
  - The LSTM can erase, write and read information from the cell

- The selection of which information is erased/written/read is controlled by three corresponding gates
  - The gates are also vectors length *n*
  - On each timestep, each element of the gates can be open (1), closed (0), or somewhere in-between.
  - The gates are dynamic: their value is computed based on the current context

Long Short-Term Memory (LSTM)

We have a sequence of inputs $\boldsymbol{x}^{(t)}$, and we will compute a sequence of hidden states $\boldsymbol{h}^{(t)}$ and cell states $\boldsymbol{c}^{(t)}$. On timestep t:

Sigmoid function: all gate values are between 0 and 1

Forget gate: controls what is kept vs forgotten, from previous cell state

Input gate: controls what parts of the new cell content are written to cell

Output gate: controls what parts of cell are output to hidden state

$$\boldsymbol{f}^{(t)} = \boxed{\sigma}\left(\boldsymbol{W}_f \boldsymbol{h}^{(t-1)} + \boldsymbol{U}_f \boldsymbol{x}^{(t)} + \boldsymbol{b}_f\right)$$

$$\boldsymbol{i}^{(t)} = \boxed{\sigma}\left(\boldsymbol{W}_i \boldsymbol{h}^{(t-1)} + \boldsymbol{U}_i \boldsymbol{x}^{(t)} + \boldsymbol{b}_i\right)$$

$$\boldsymbol{o}^{(t)} = \boxed{\sigma}\left(\boldsymbol{W}_o \boldsymbol{h}^{(t-1)} + \boldsymbol{U}_o \boldsymbol{x}^{(t)} + \boldsymbol{b}_o\right)$$

New cell content: this is the new content to be written to the cell

Cell state: erase ("forget") some content from last cell state, and write ("input") some new cell content

Hidden state: read ("output") some content from the cell

$$\tilde{\boldsymbol{c}}^{(t)} = \tanh\left(\boldsymbol{W}_c \boldsymbol{h}^{(t-1)} + \boldsymbol{U}_c \boldsymbol{x}^{(t)} + \boldsymbol{b}_c\right)$$

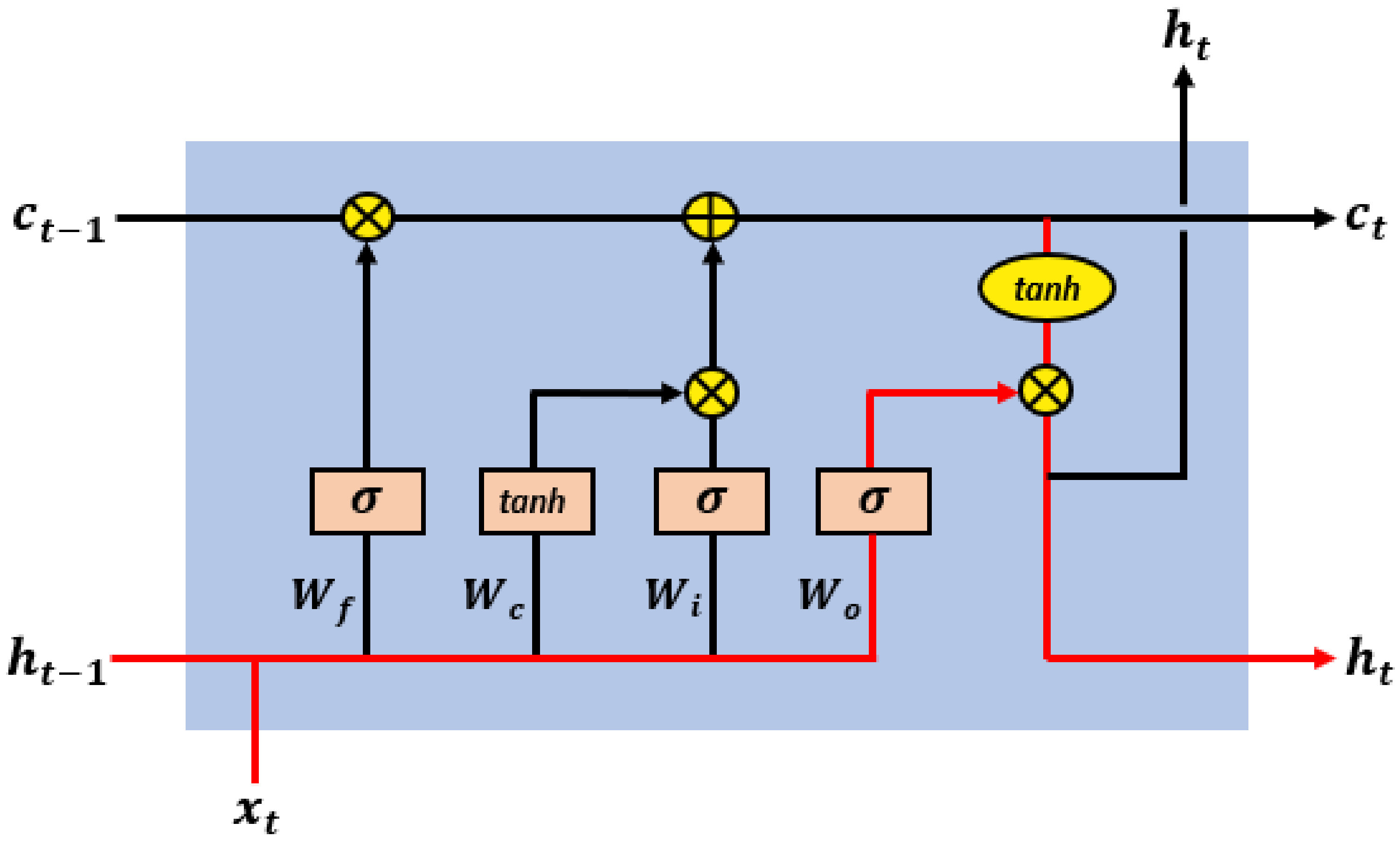
$$\boldsymbol{c}^{(t)} = \boldsymbol{f}^{(t)} \circ \boldsymbol{c}^{(t-1)} + \boldsymbol{i}^{(t)} \circ \tilde{\boldsymbol{c}}^{(t)}$$

$$\boldsymbol{h}^{(t)} = \boldsymbol{o}^{(t)} \circ \tanh \boldsymbol{c}^{(t)}$$

All these are vectors of same length n

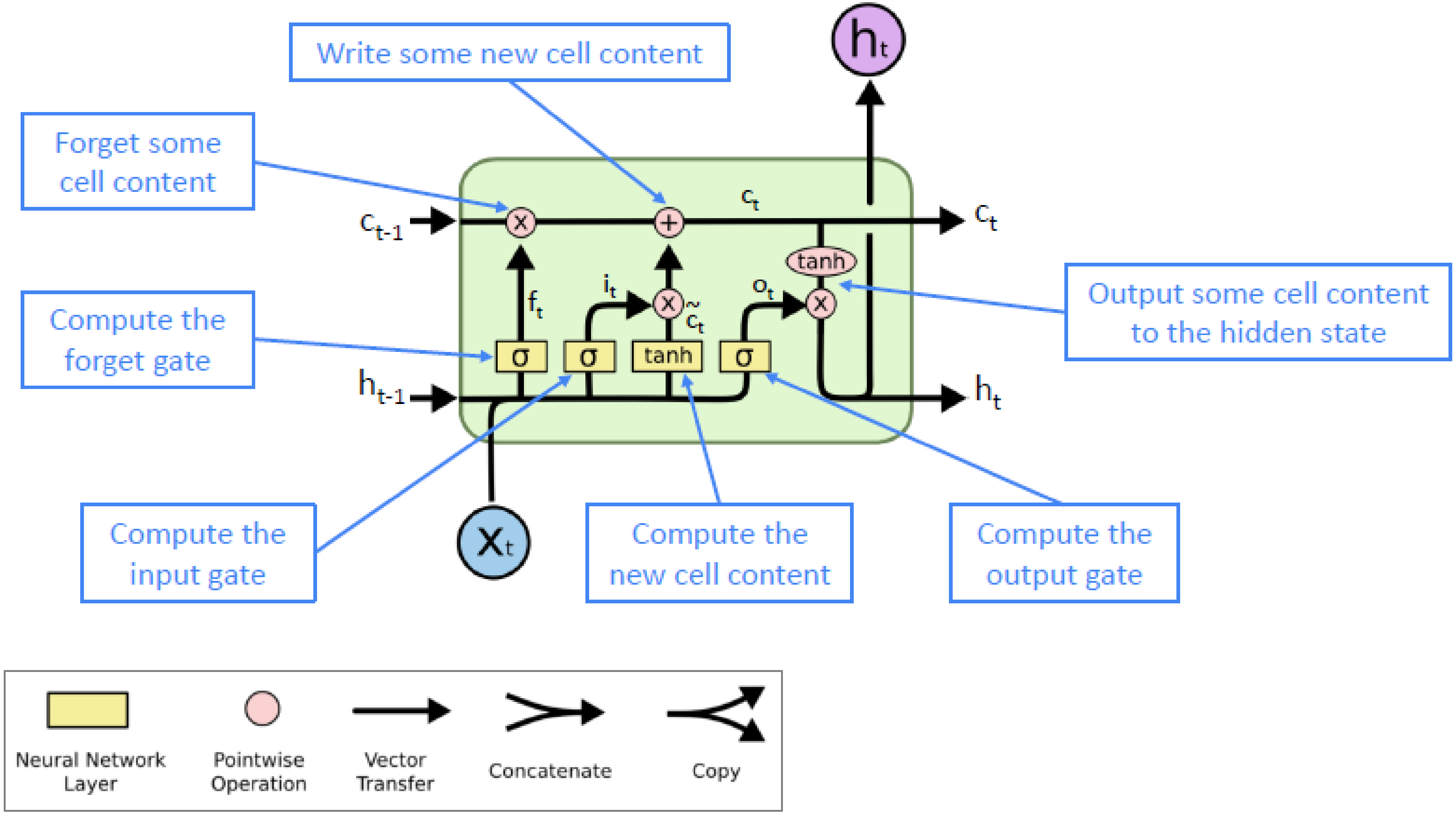
Gates are applied using element-wise product

The LSTM output gate's action on the cell state



Long Short-Term Memory (LSTM)

You can think of the LSTM equations visually like this:

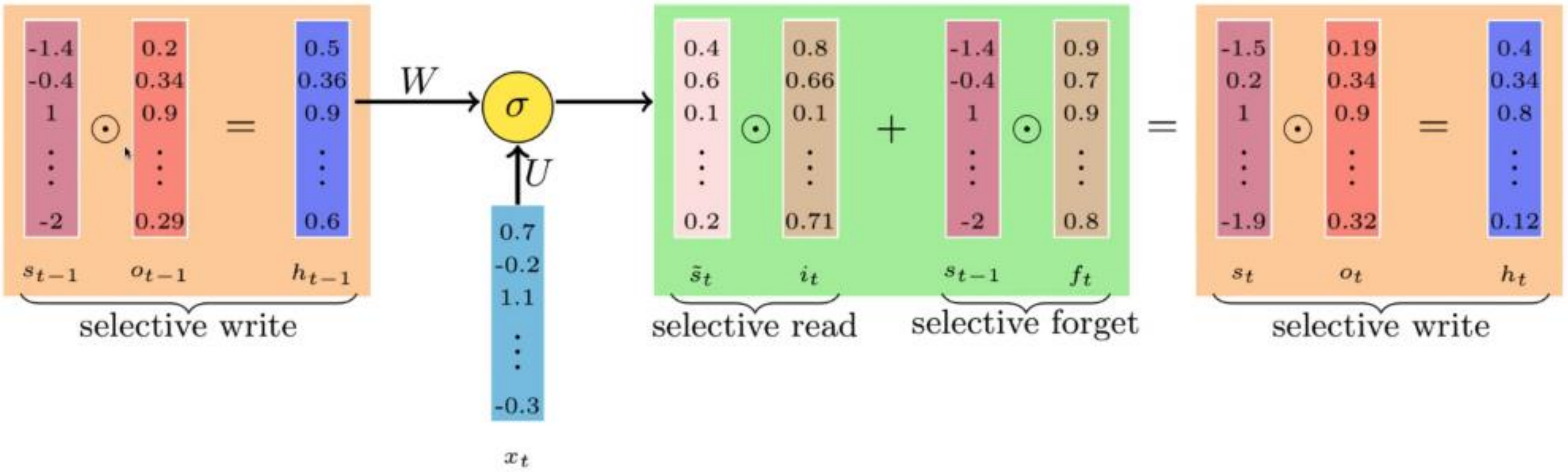


Gates:

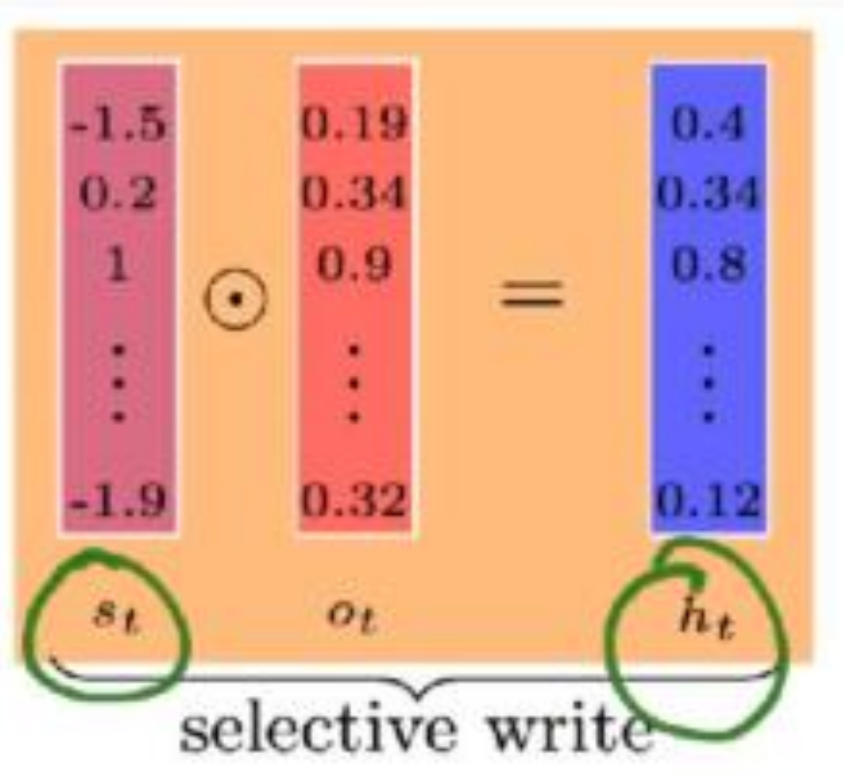
$$o_t = \sigma(W_o h_{t-1} + U_o x_t + b_o)$$
$$i_t = \sigma(W_i h_{t-1} + U_i x_t + b_i)$$
$$f_t = \sigma(W_f h_{t-1} + U_f x_t + b_f)$$

States:

$$\tilde{s}_t = \sigma(W h_{t-1} + U x_t + b)$$
$$s_t = f_t \odot s_{t-1} + i_t \odot \tilde{s}_t$$
$$h_t = o_t \odot \sigma(s_t)$$

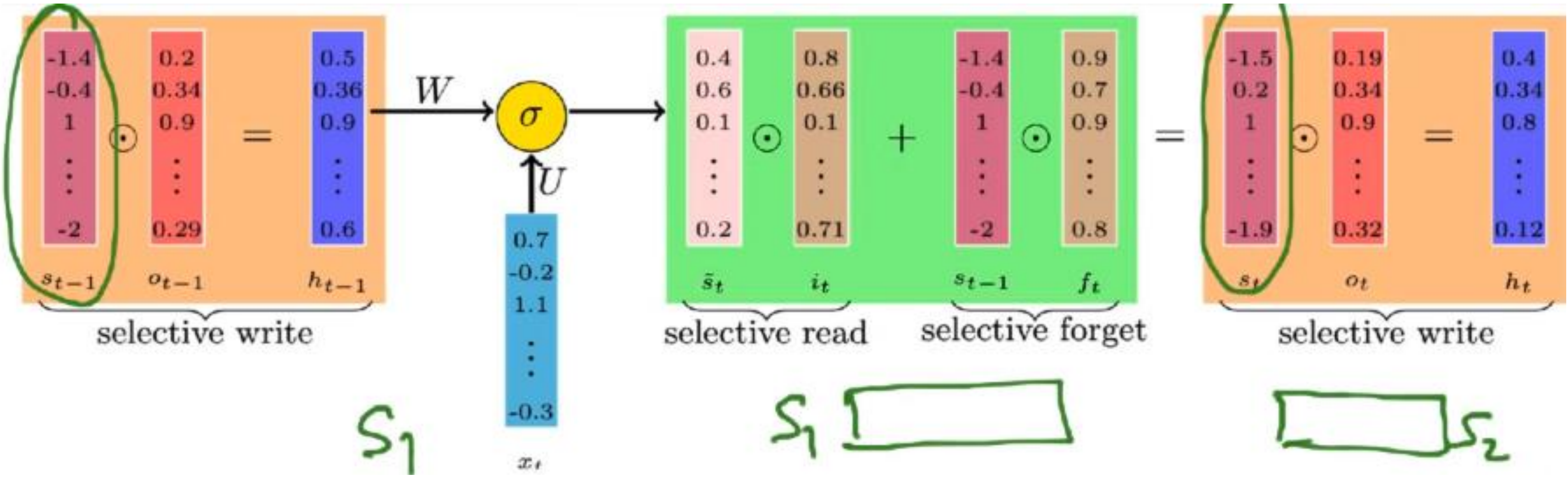


https://medium.com/@prvnk10/how-lstms-solve-the-problem-of-vanishing-gradients-ea88f08c78ca

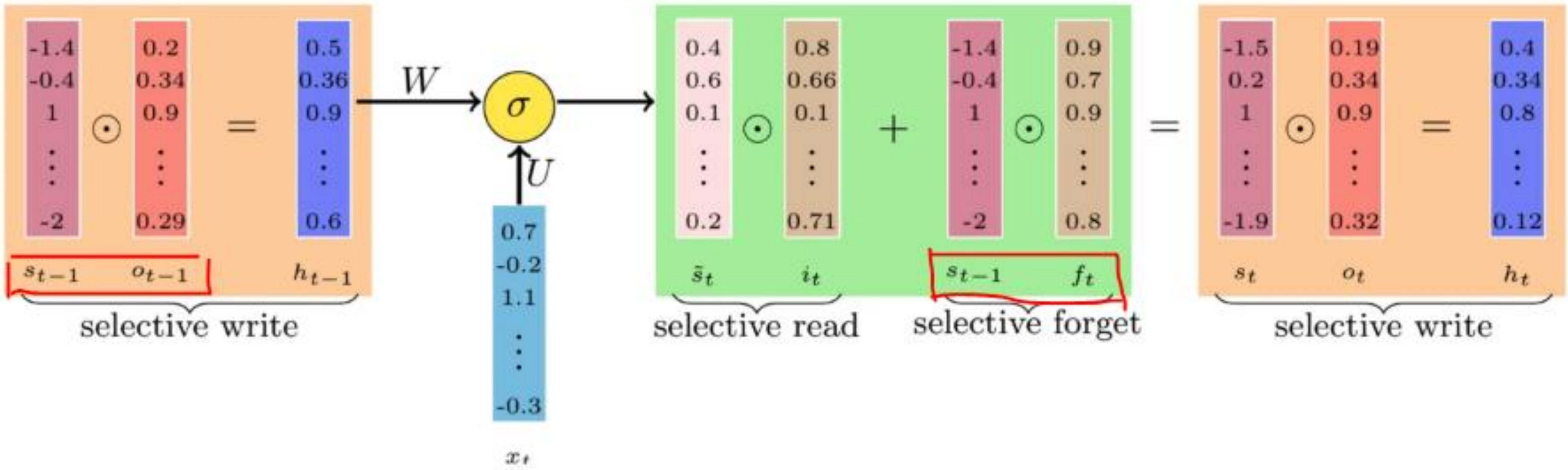


$$h_t = (s_t)*(o_t)$$

https://medium.com/@prvnk10/how-lstms-solve-the-problem-of-vanishing-gradients-ea88f08c78ca



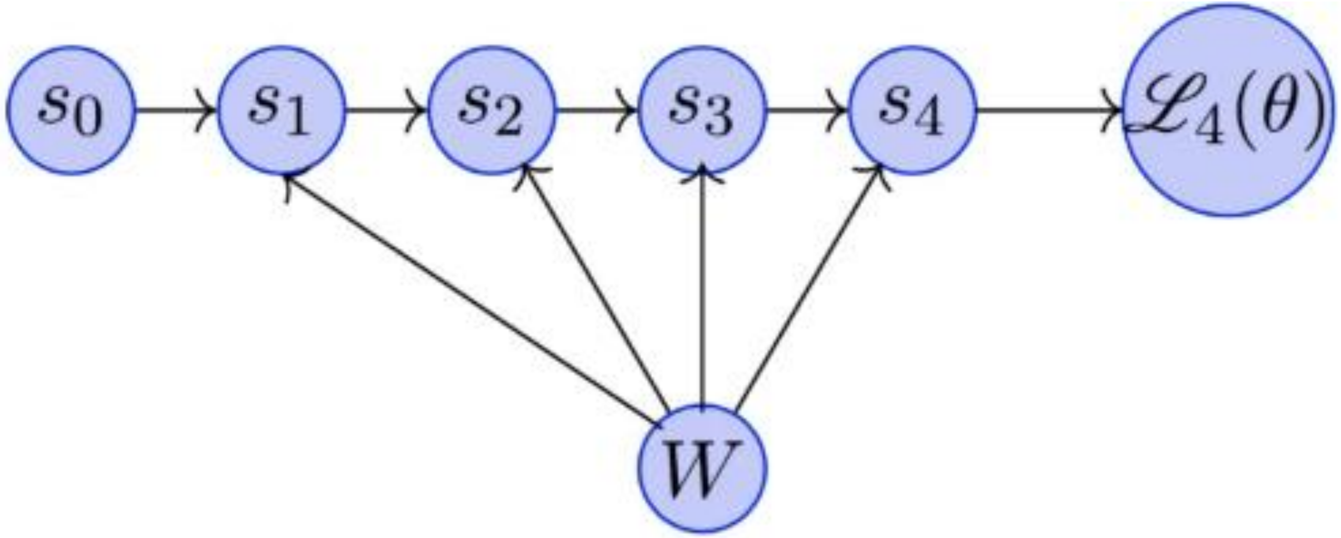
https://medium.com/@prvnk10/how-lstms-solve-the-problem-of-vanishing-gradients-ea88f08c78ca



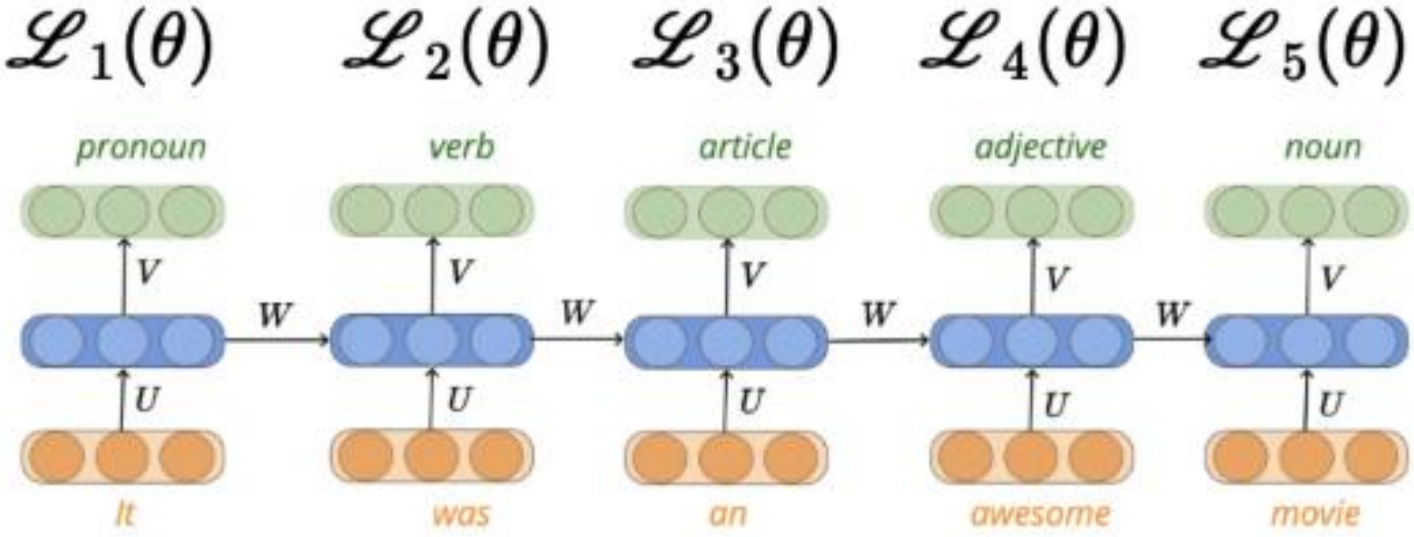
https://medium.com/@prvnk10/how-lstms-solve-the-problem-of-vanishing-gradients-ea88f08c78ca

If the state at time $t-1$ did not contribute much to the state at time t (i.e., if $\|f_t\| \rightarrow 0$ and $\|o_{t-1}\| \rightarrow 0$) then during backpropagation the gradients flowing into s_{t-1} will vanish

https://medium.com/@prvnk10/how-lstms-solve-the-problem-of-vanishing-gradients-ea88f08c78ca



s_0
s_1
s_2
s_3
s_4
$\mathscr{L}_4(\theta)$
W



$\mathscr{L}_1(\theta)$
$\mathscr{L}_2(\theta)$
$\mathscr{L}_3(\theta)$
$\mathscr{L}_4(\theta)$
$\mathscr{L}_5(\theta)$
pronoun
verb
article
adjective
noun
V
W
U
It
was
an
awesome
movie

In general, the gradient of $\mathscr{L}_t(\theta)$ w.r.t. θ_i vanishes when the gradients flowing through **each and every path** from $L_t(\theta)$ to θ_i vanish.

On the other hand, the gradient of $\mathscr{L}_t(\theta)$ w.r.t. θ_i explodes when the gradient flowing through **at least one path** explodes.