

LSTM (LONG-SHORT TERM MEMORY) RNNs



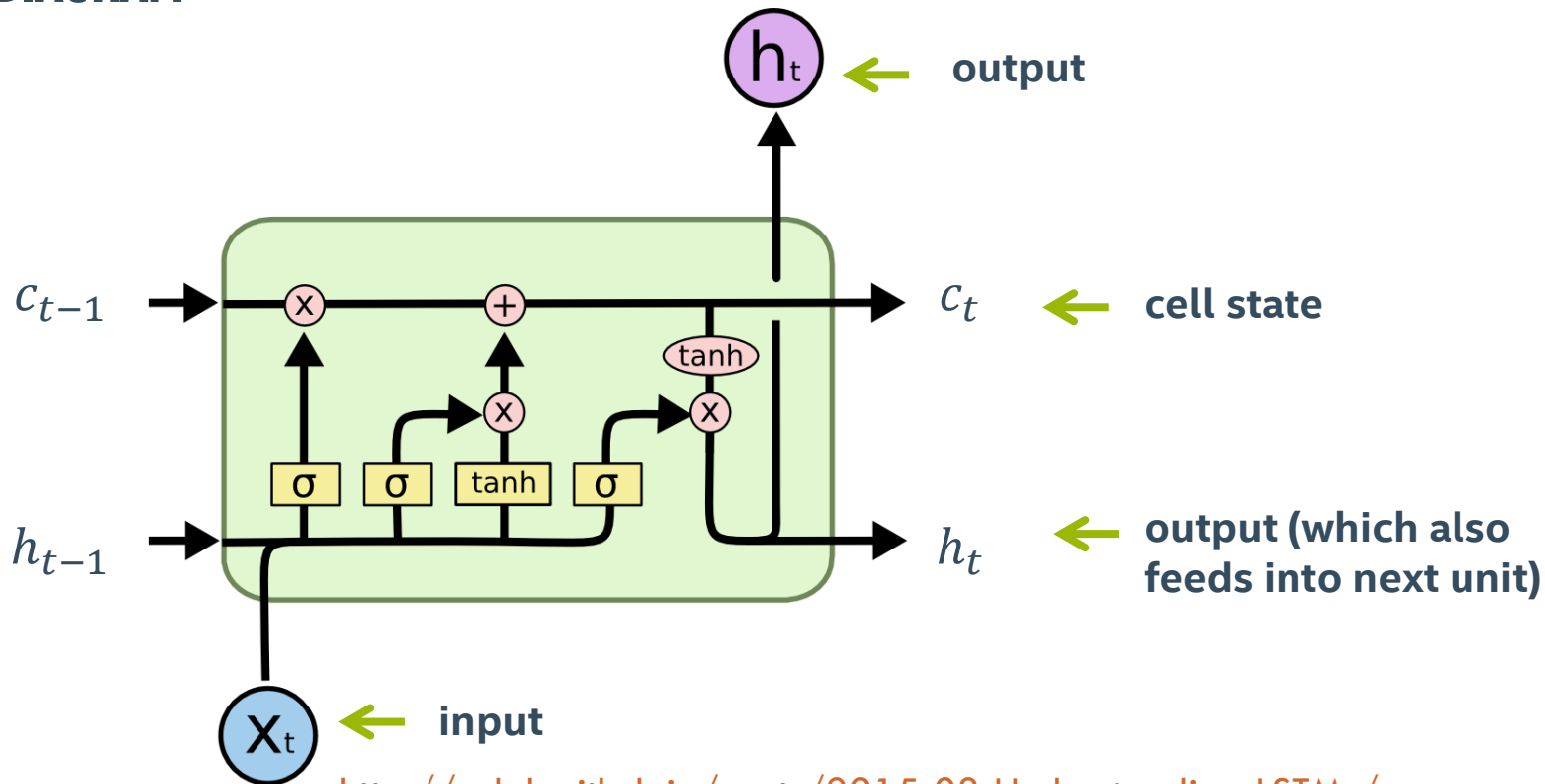
ISSUE: STANDARD RNNs HAVE POOR MEMORY

- Transition Matrix necessarily weakens signal
- Need a structure that can leave some dimensions unchanged over many steps
- This is the problem addressed by so-called Long-Short Term Memory RNNs (LSTM)

IDEA: MAKE “REMEMBERING” EASY

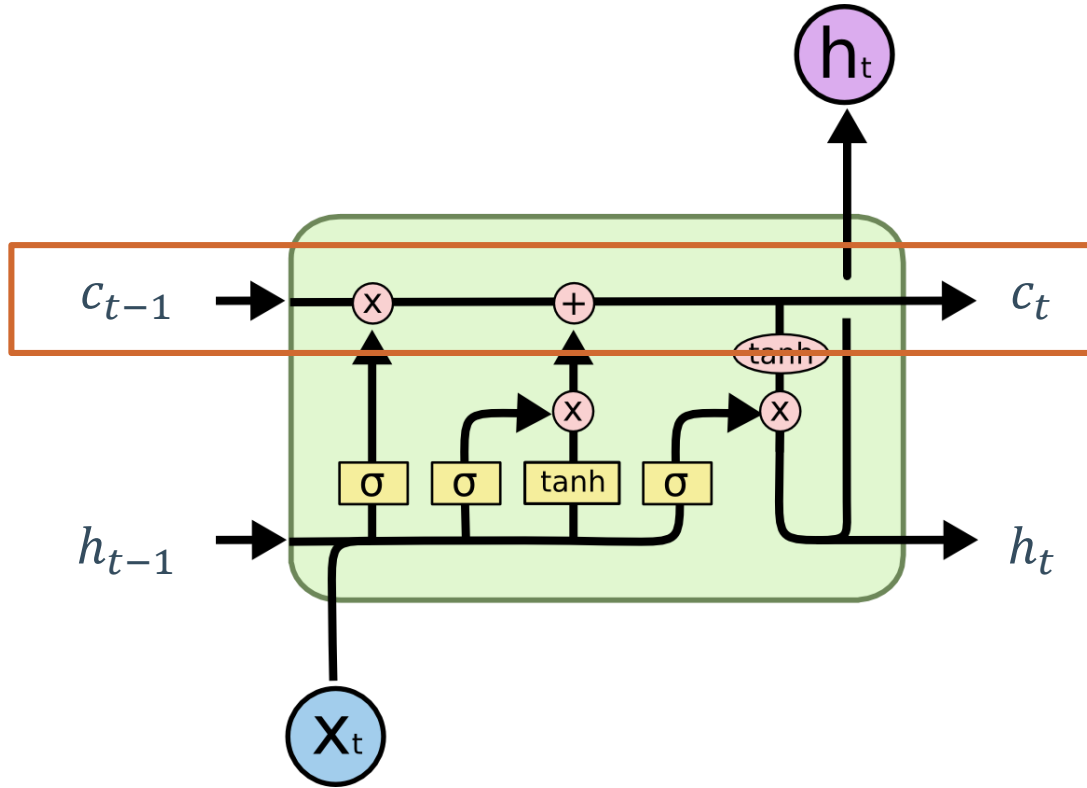
- Define a more complicated update mechanism for the changing of the internal state
- By default, LSTMs remember the information from the last step
- Items are overwritten as an active choice

LSTM DIAGRAM



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

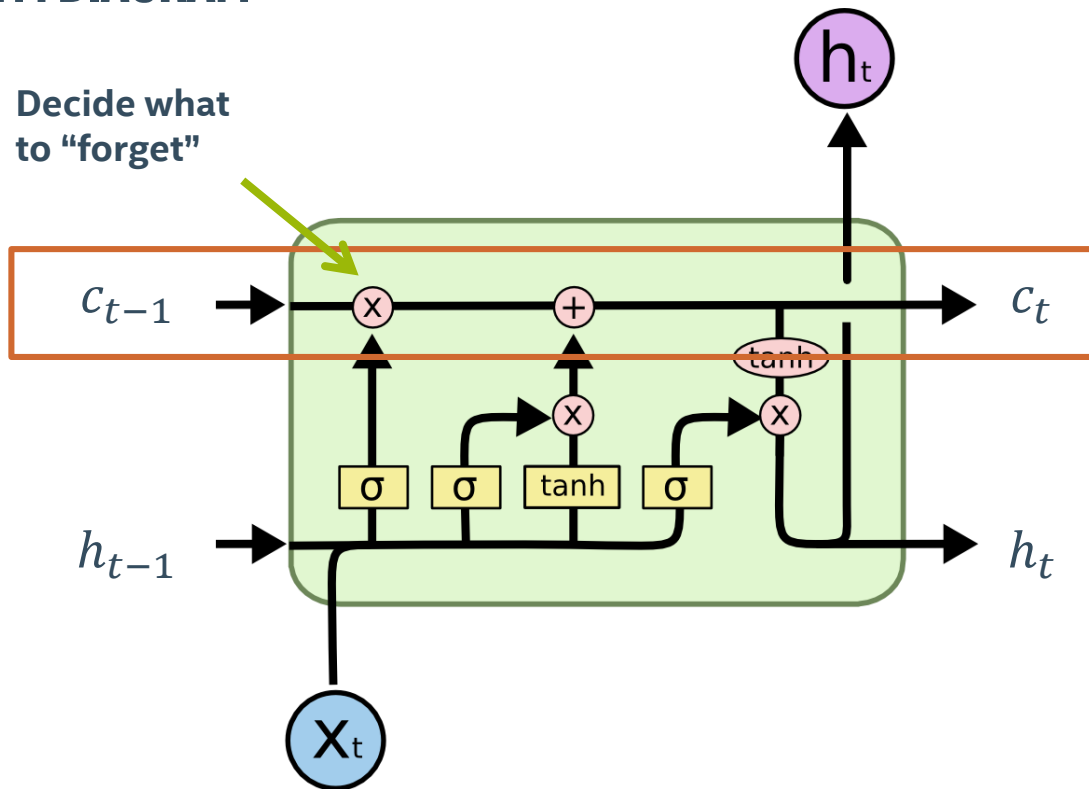
LSTM DIAGRAM



cell state gets updated
in two stages

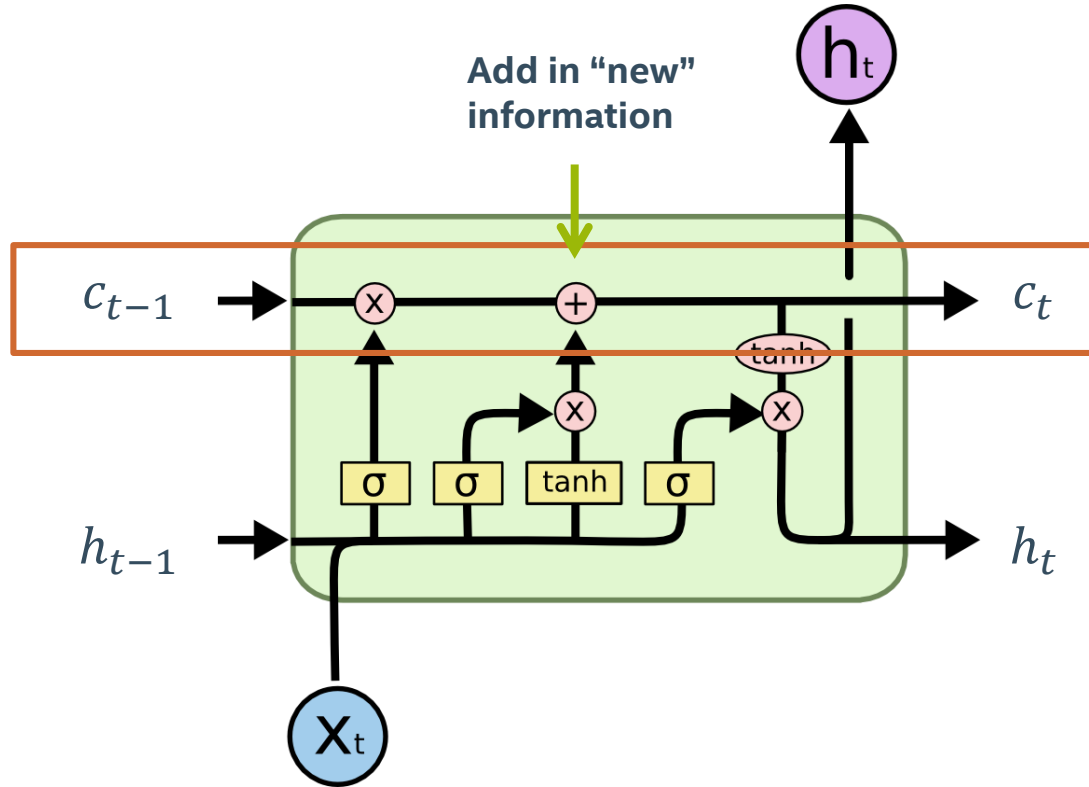
LSTM DIAGRAM

Decide what
to "forget"



cell state gets updated
in two stages

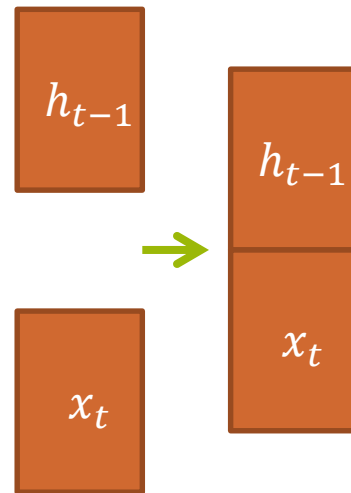
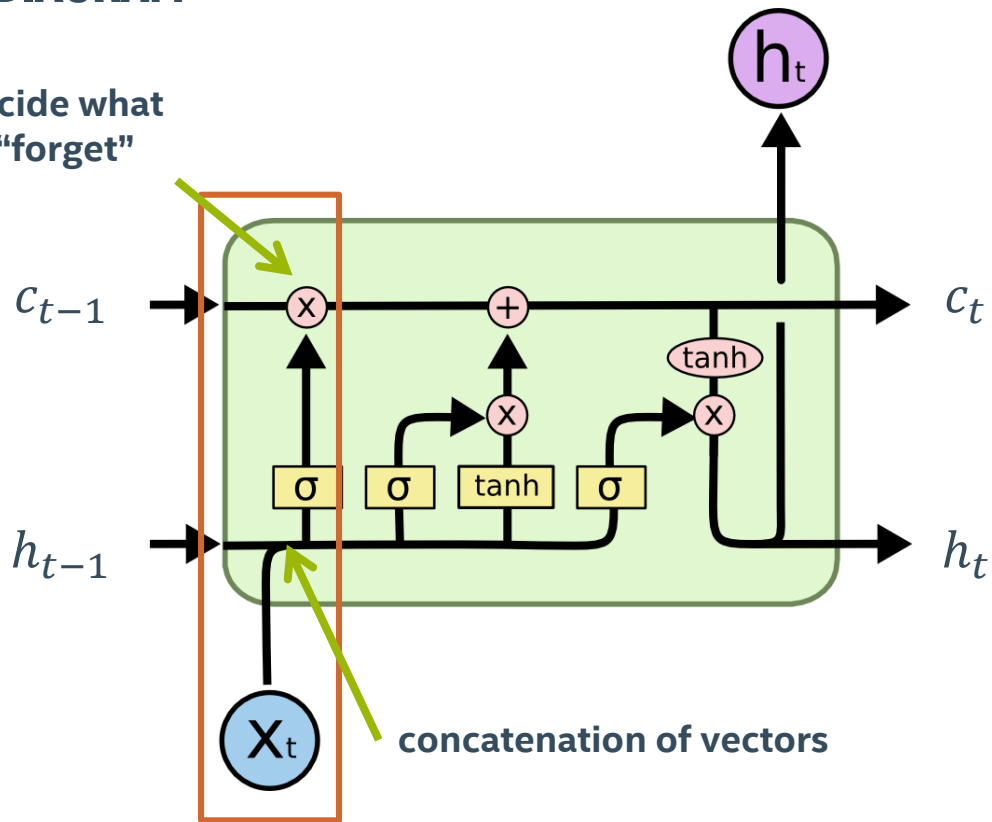
LSTM DIAGRAM



cell state gets
updated in two stages

LSTM DIAGRAM

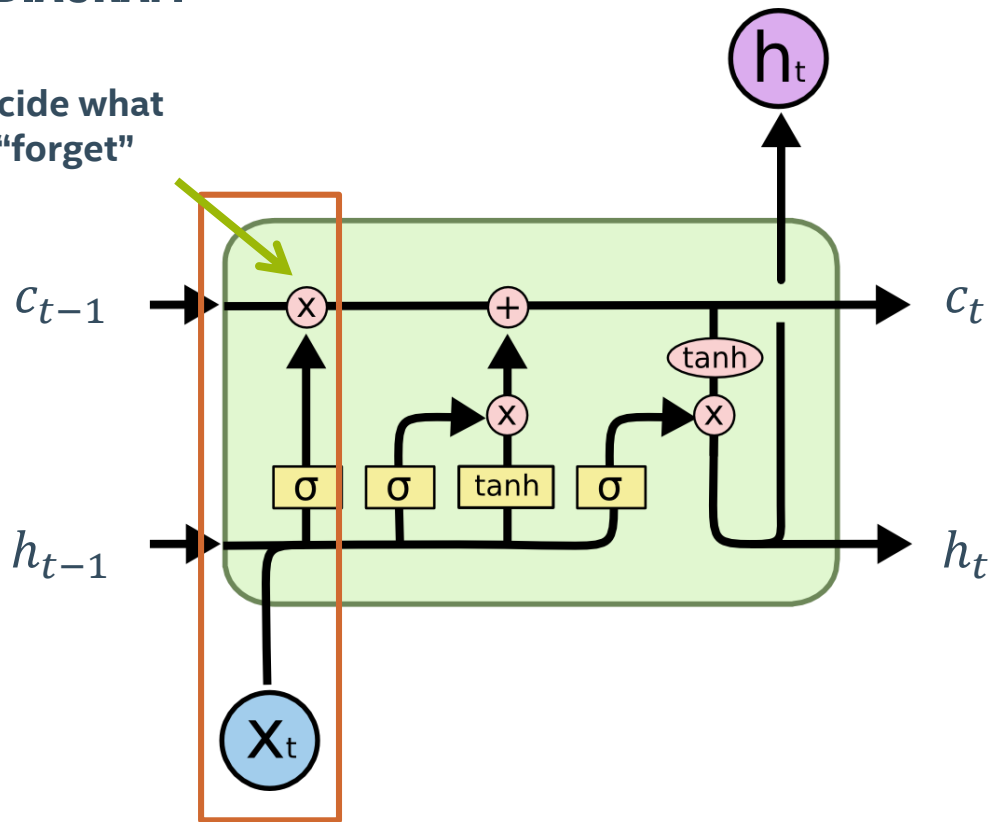
Decide what
to "forget"



$$[h_{t-1}, x_t]$$

LSTM DIAGRAM

Decide what
to "forget"

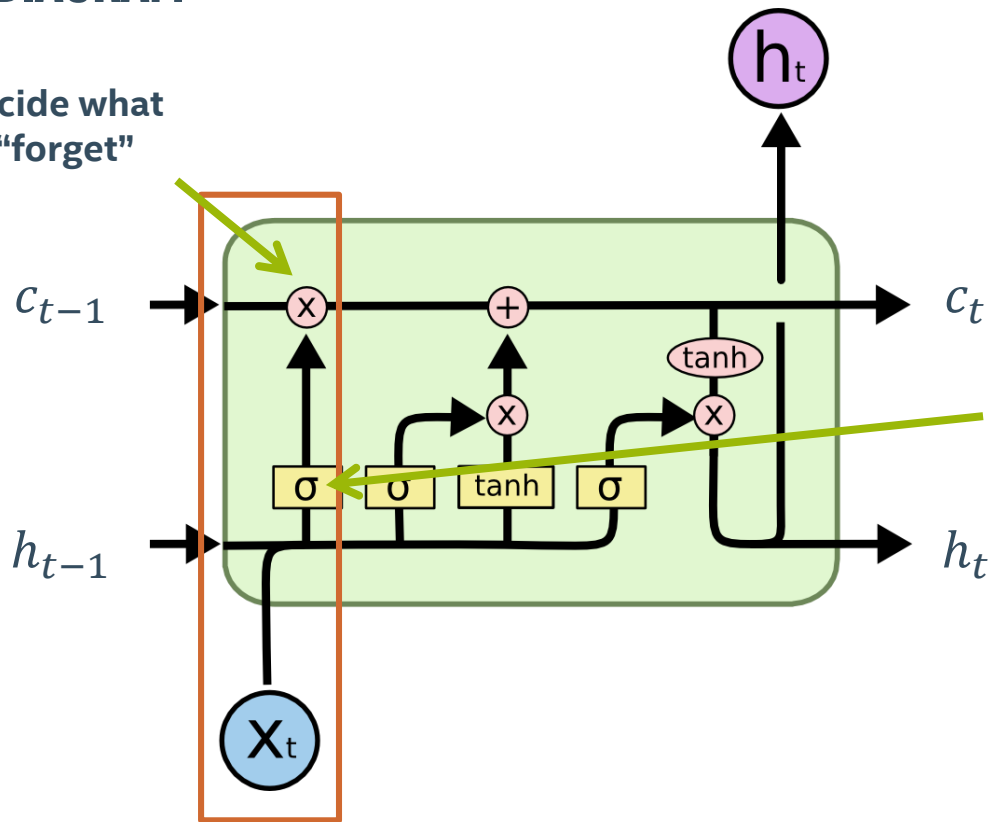


$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$$

based on previous output
and current input

LSTM DIAGRAM

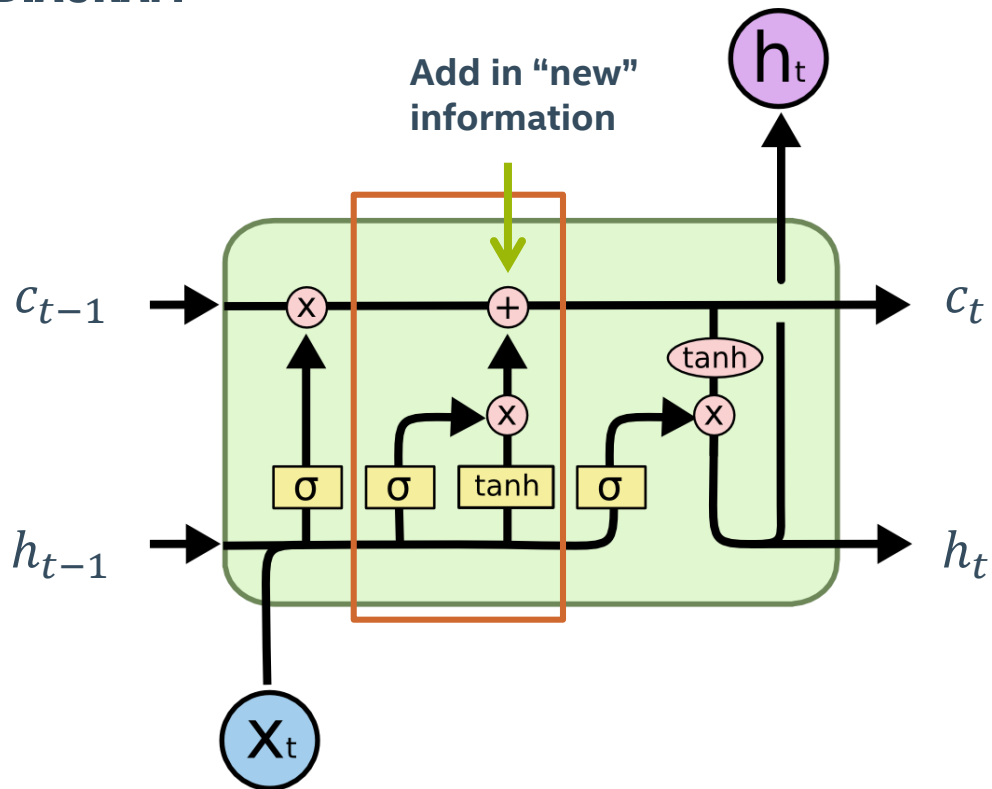
Decide what
to "forget"



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

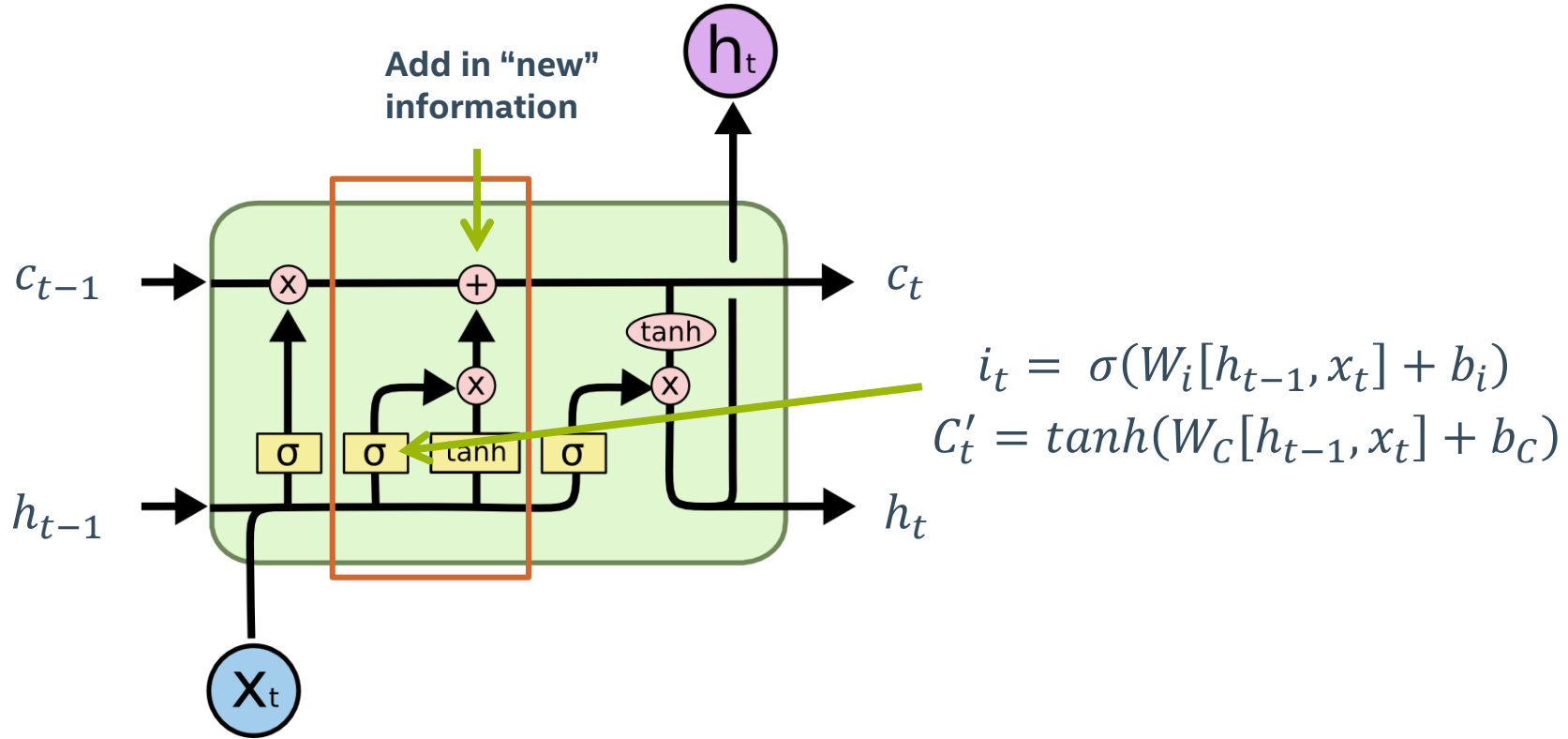
based on previous output
and current input

LSTM DIAGRAM

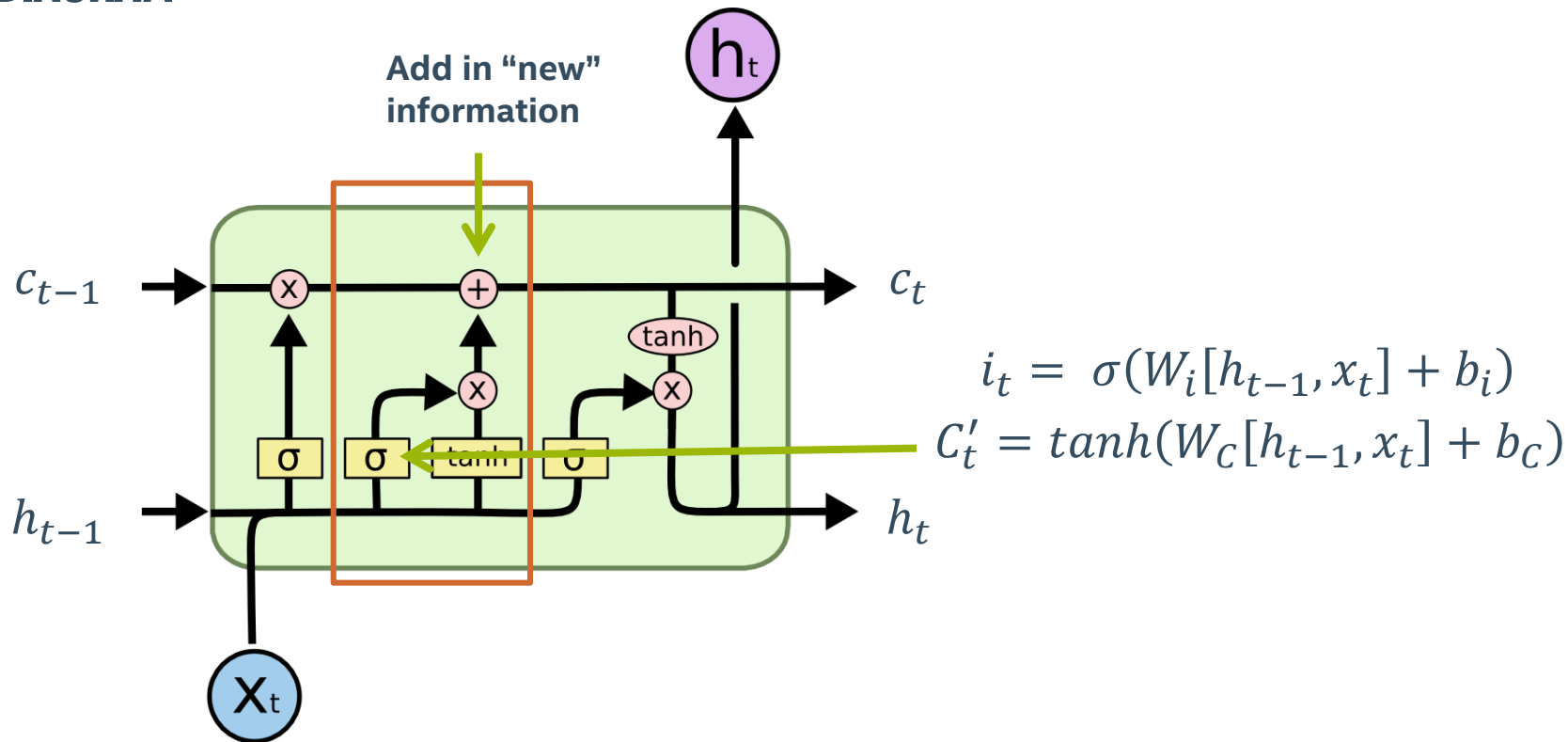


$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$
$$C'_t = \tanh(W_C[h_{t-1}, x_t] + b_C)$$

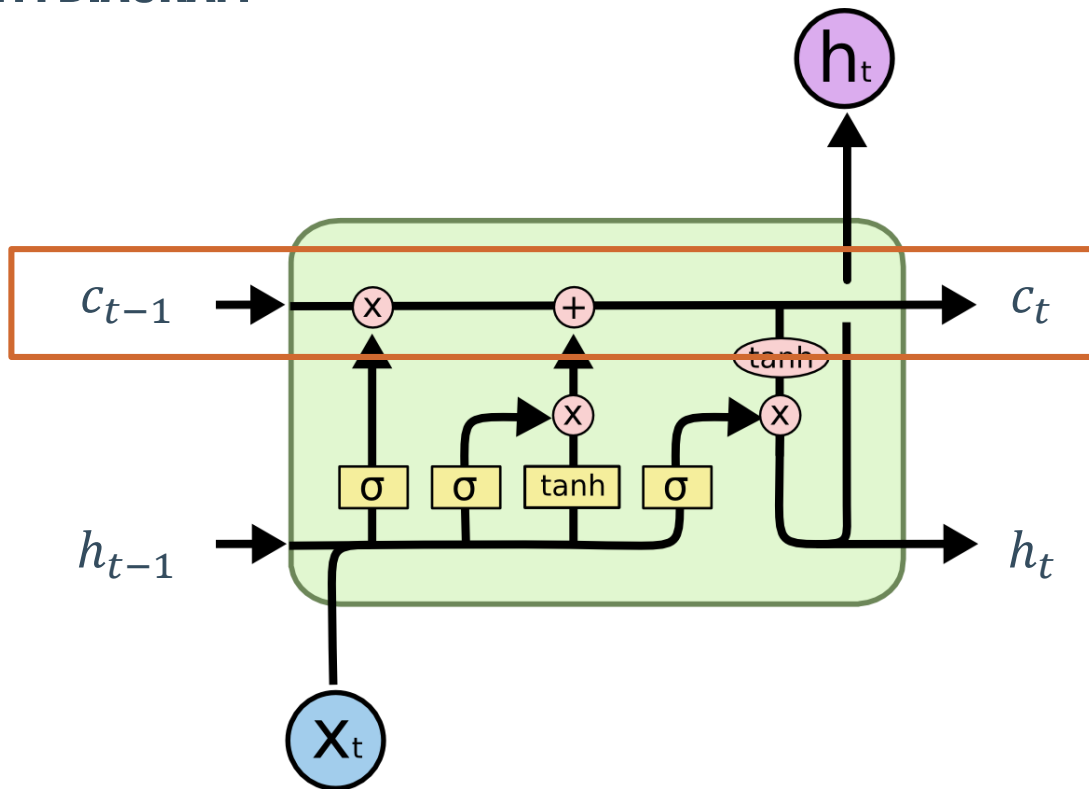
LSTM DIAGRAM



LSTM DIAGRAM



LSTM DIAGRAM



Note: '*' represents element-wise multiplication

$$C_t = f_i * C_{t-1} + i_t * C'_t$$

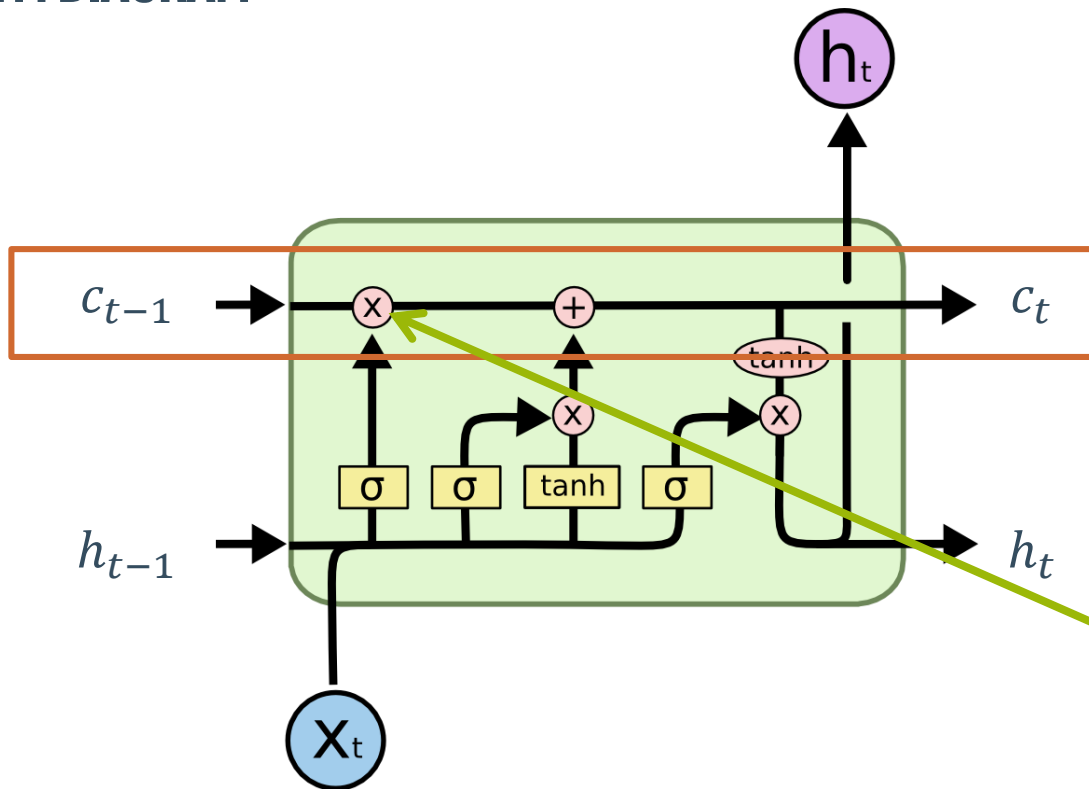


forget
the old
(or not)



add
the new
(or not)

LSTM DIAGRAM



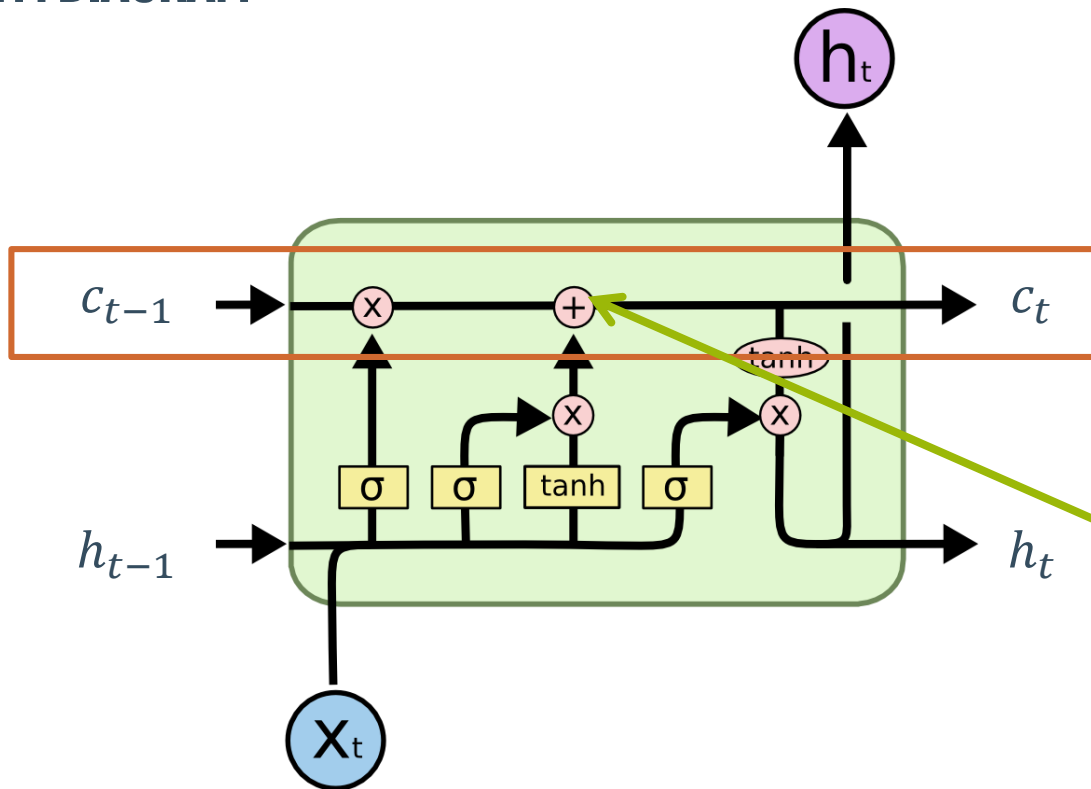
Note: '*' represents element-wise multiplication

$$C_t = f_i * C_{t-1} + i_t * C'_t$$

forget
the old
(or not)

add
the new
(or not)

LSTM DIAGRAM



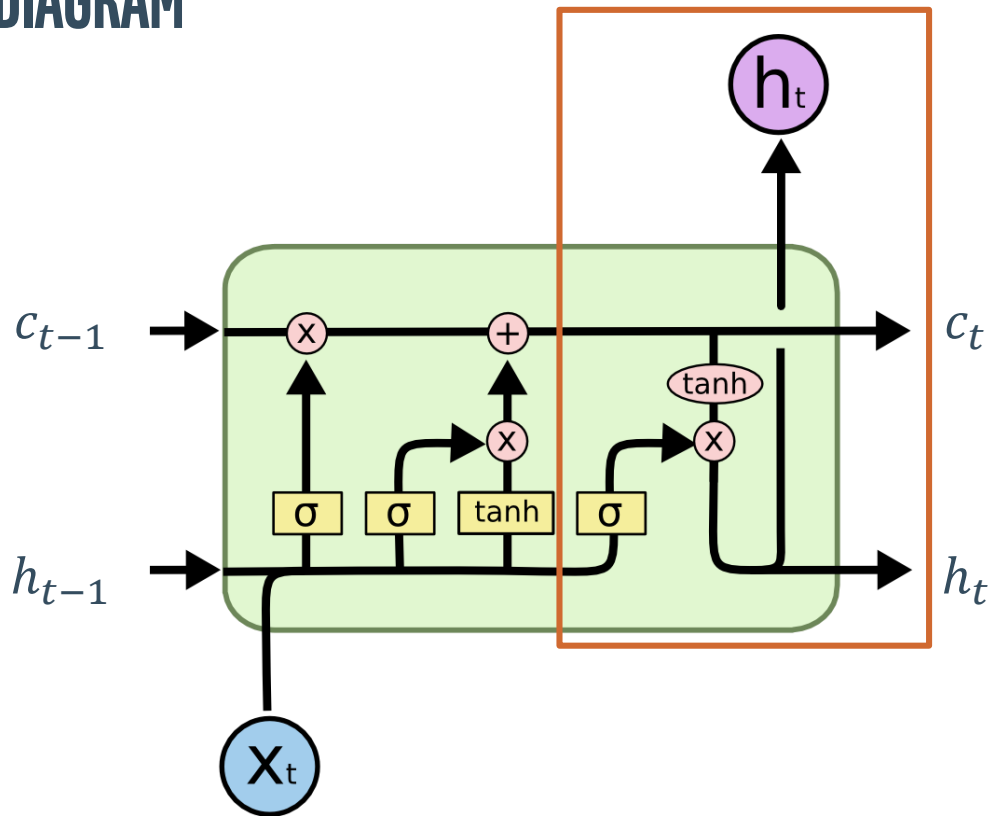
Note: '*' represents element-wise multiplication

$$C_t = f_i * C_{t-1} + i_t * C'_t$$

forget
the old
(or not)

add
the new
(or not)

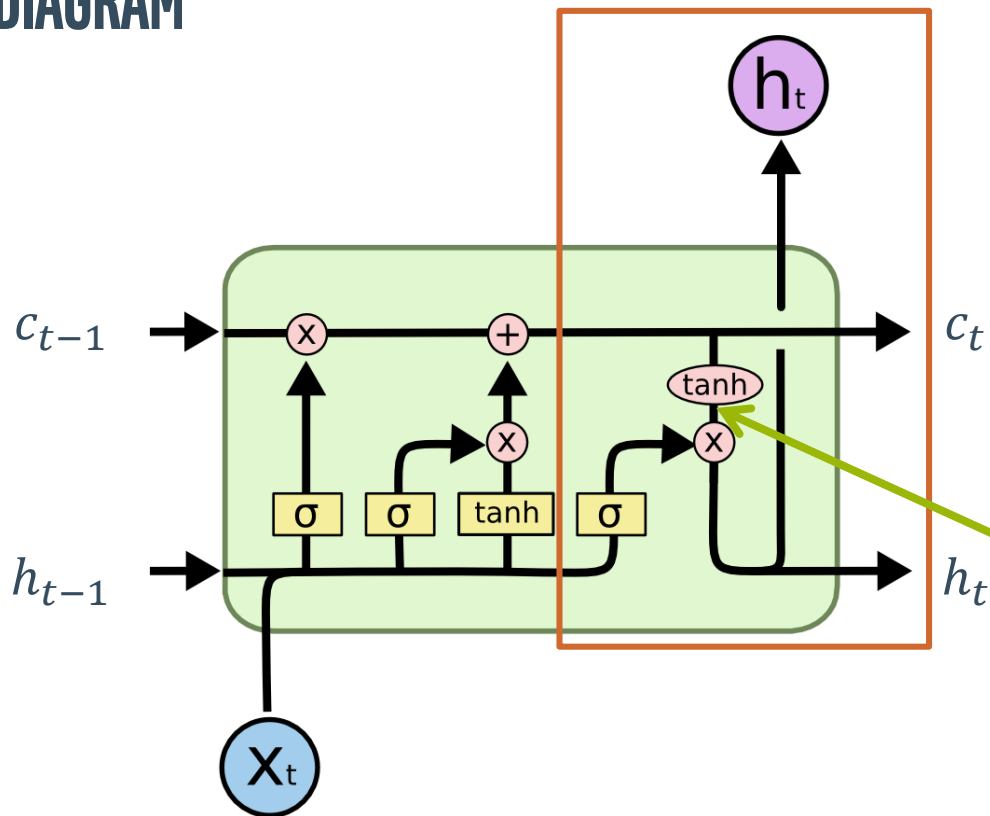
LSTM DIAGRAM



Final stage computes the output

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

LSTM DIAGRAM

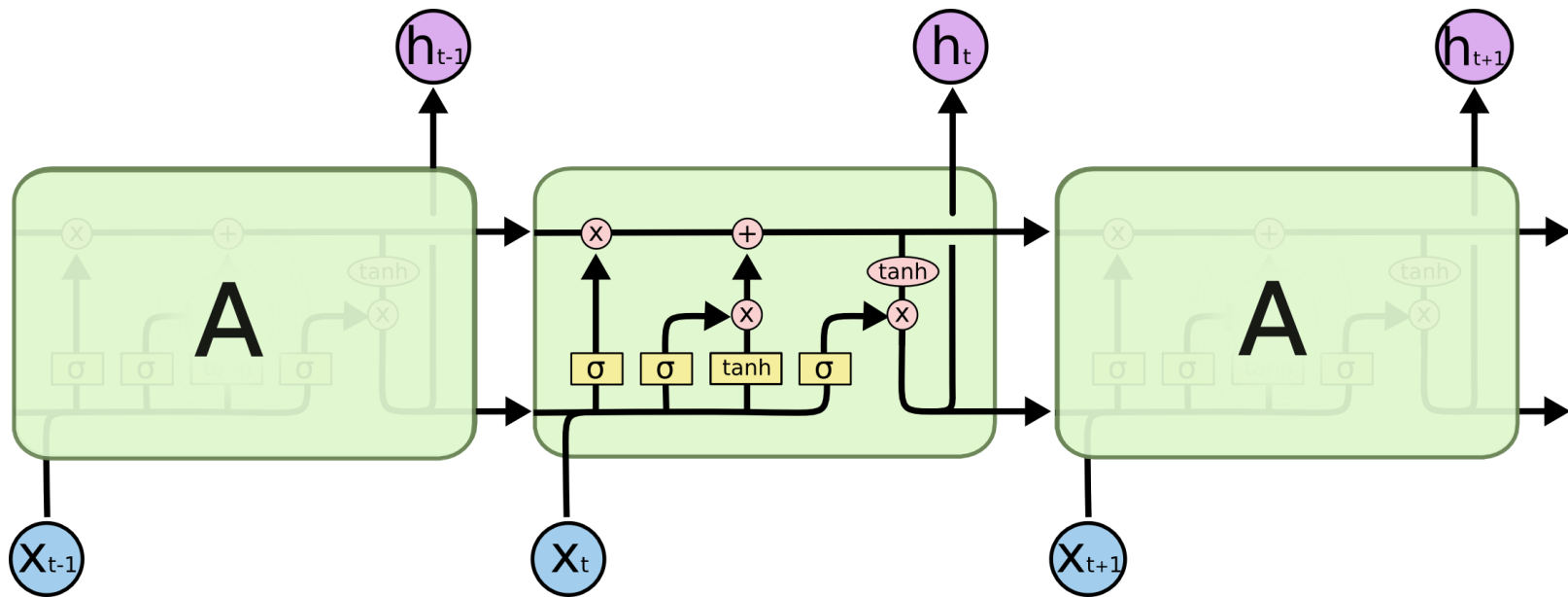


Final stage computes the output

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

Note: No weights here

LSTM UNROLLED



FINAL POINTS

- This is the most common version of LSTM, but there are many different “flavors”
 - Gated Recurrent Unit (GRU)
 - Depth-Gated RNN
- LSTMs have considerably more parameters than plain RNNs
- Most of the big performance improvements in NLP have come from LSTMs, not plain RNN

