Course 9

Deep Learning

Christophe Eloy

Outline

- Deep supervised learning
- PyTorch
- Example: Fully-connected network for classification
- CNNs
- Other architectures

Mini-batch gradient descent

- Batch

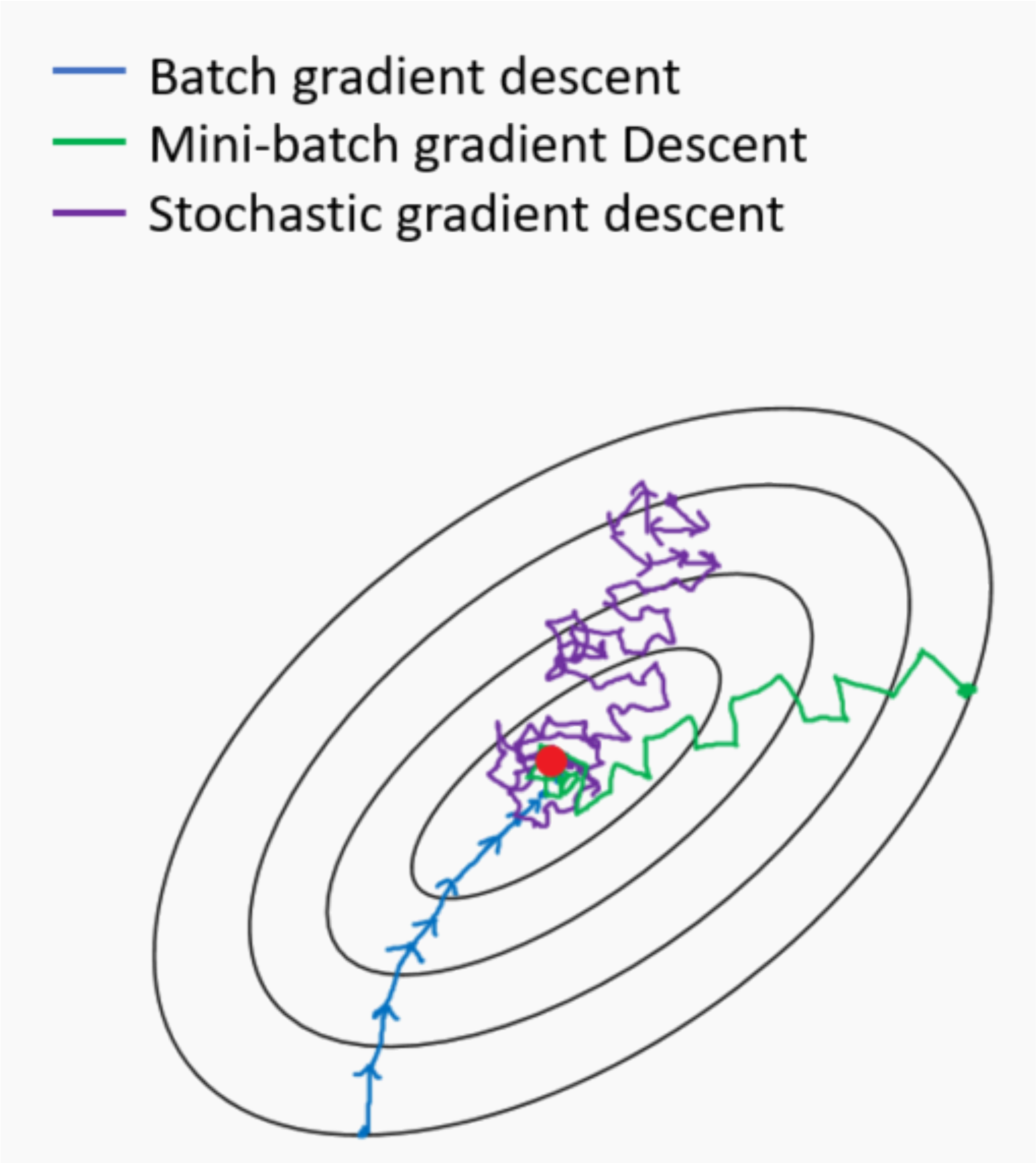
$$\theta_j := \theta_j - \frac{\alpha}{N}\sum_{i=1}^{N}\left(h_\theta(\boldsymbol{x}^{(i)}) - y^{(i)}\right)x_j^{(i)}$$

- Mini-batch

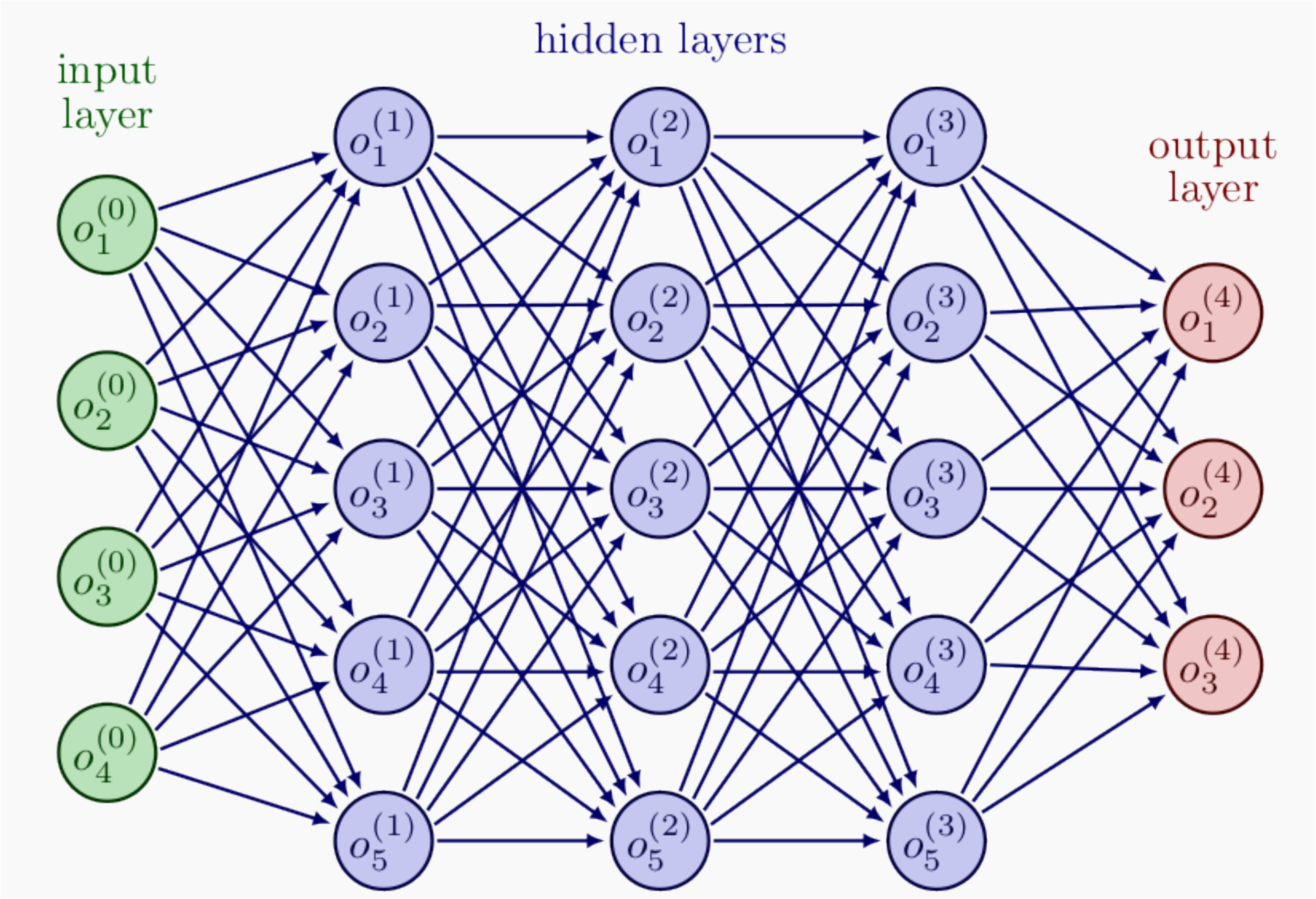
$$\theta_j := \theta_j - \frac{\alpha}{m}\sum_{i=1}^{m}\left(h_\theta(\boldsymbol{x}^{(i)}) - y^{(i)}\right)x_j^{(i)}$$

- Stochastic

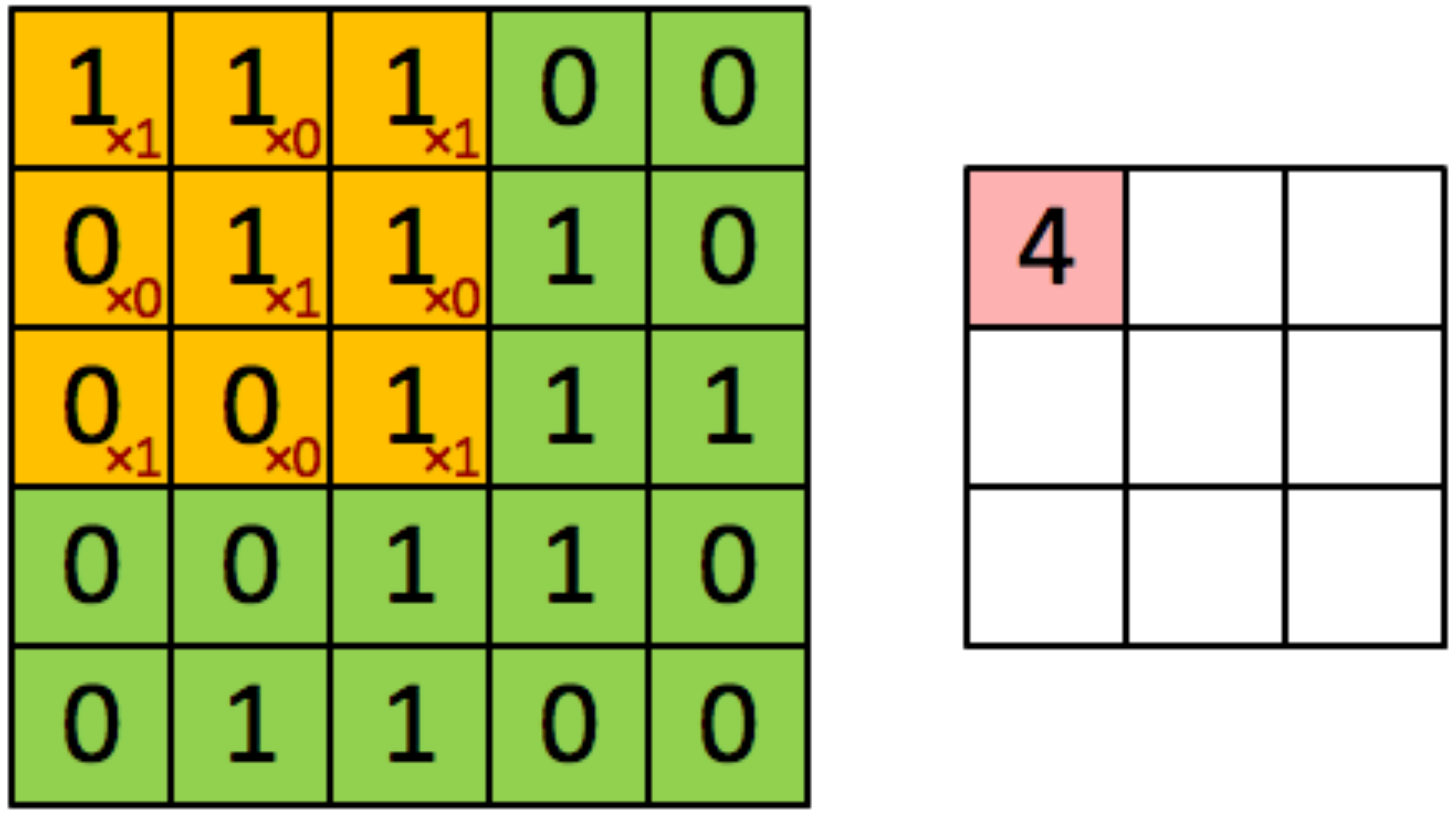
$$\theta_j := \theta_j - \alpha\left(h_\theta(\boldsymbol{x}^{(i)}) - y^{(i)}\right)x_j^{(i)}$$



Fully-connected neural networks



Convolutional neural networks

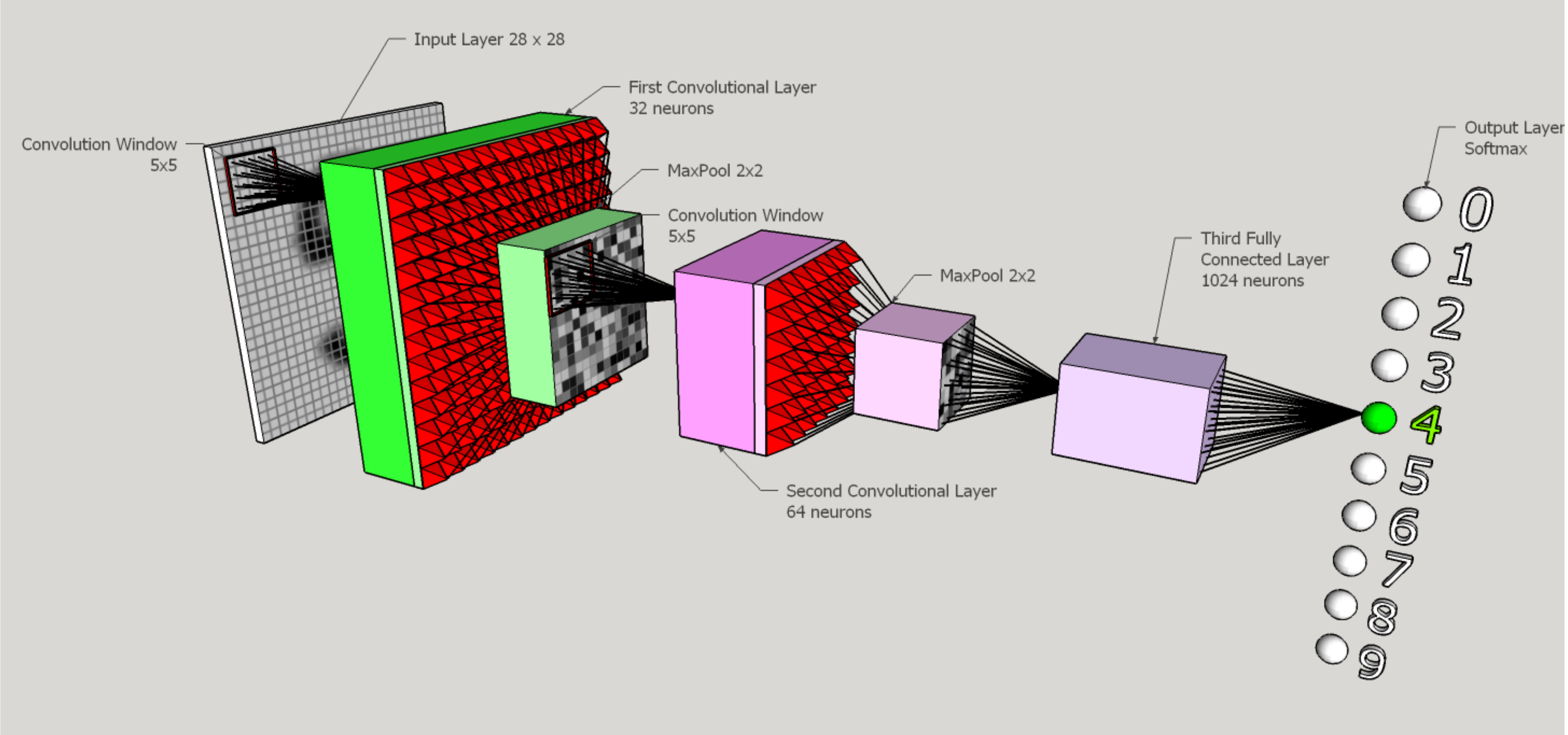


Image

Convolved Feature

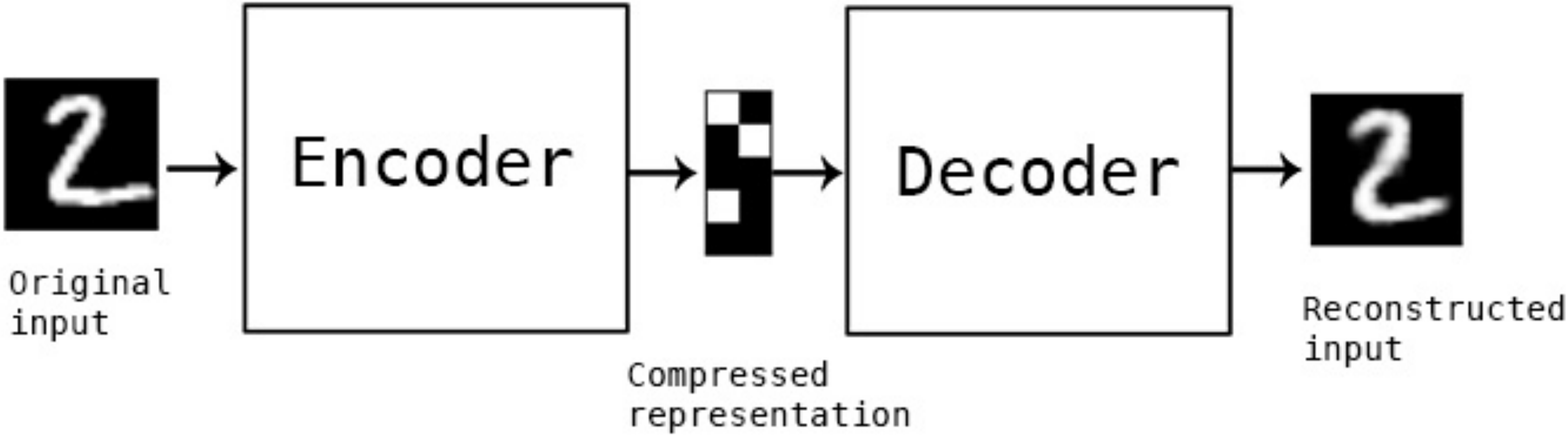
https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

Convolutional neural networks

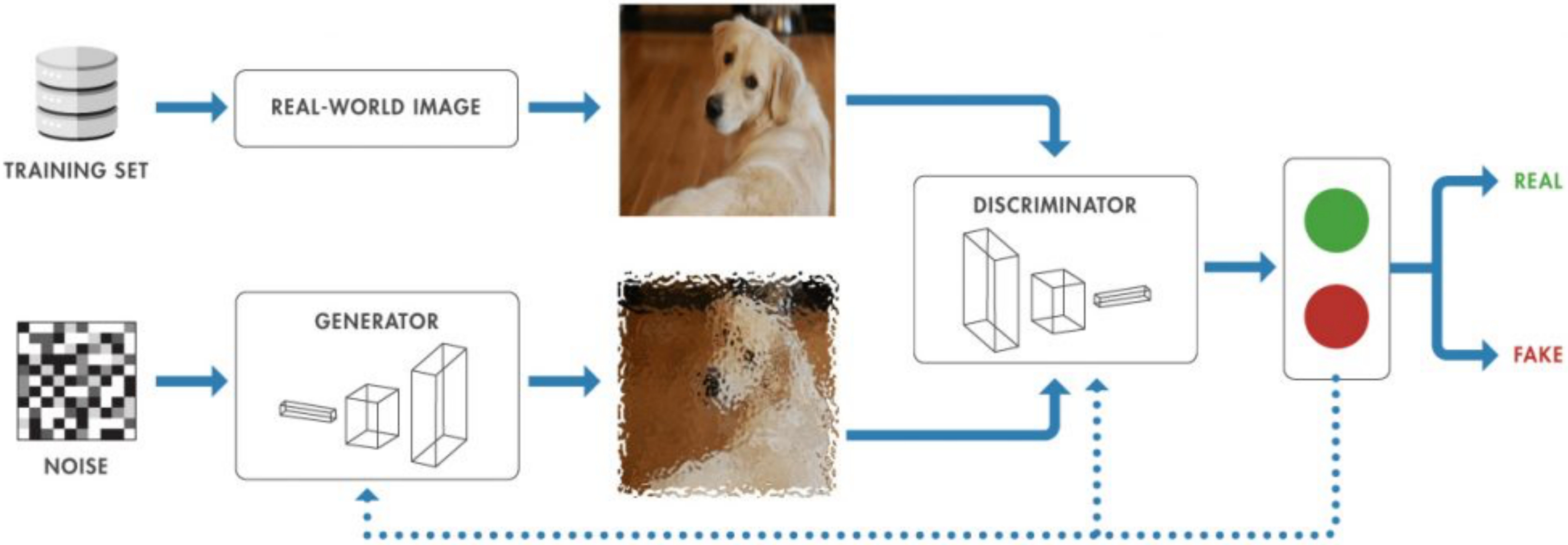


http://zderadicka.eu/revival-of-neural-networks/

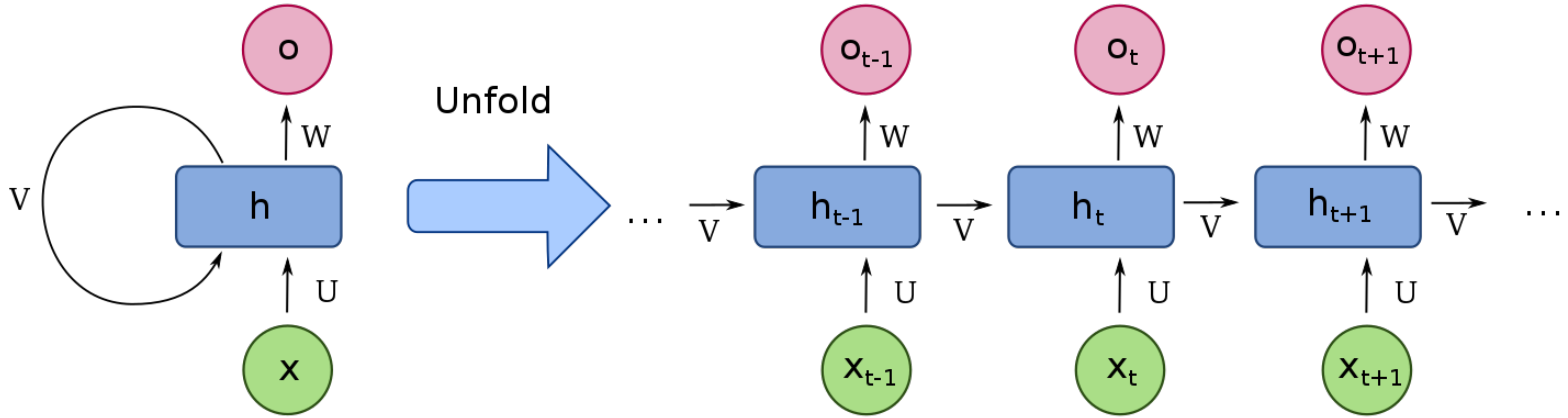
Autoencoders



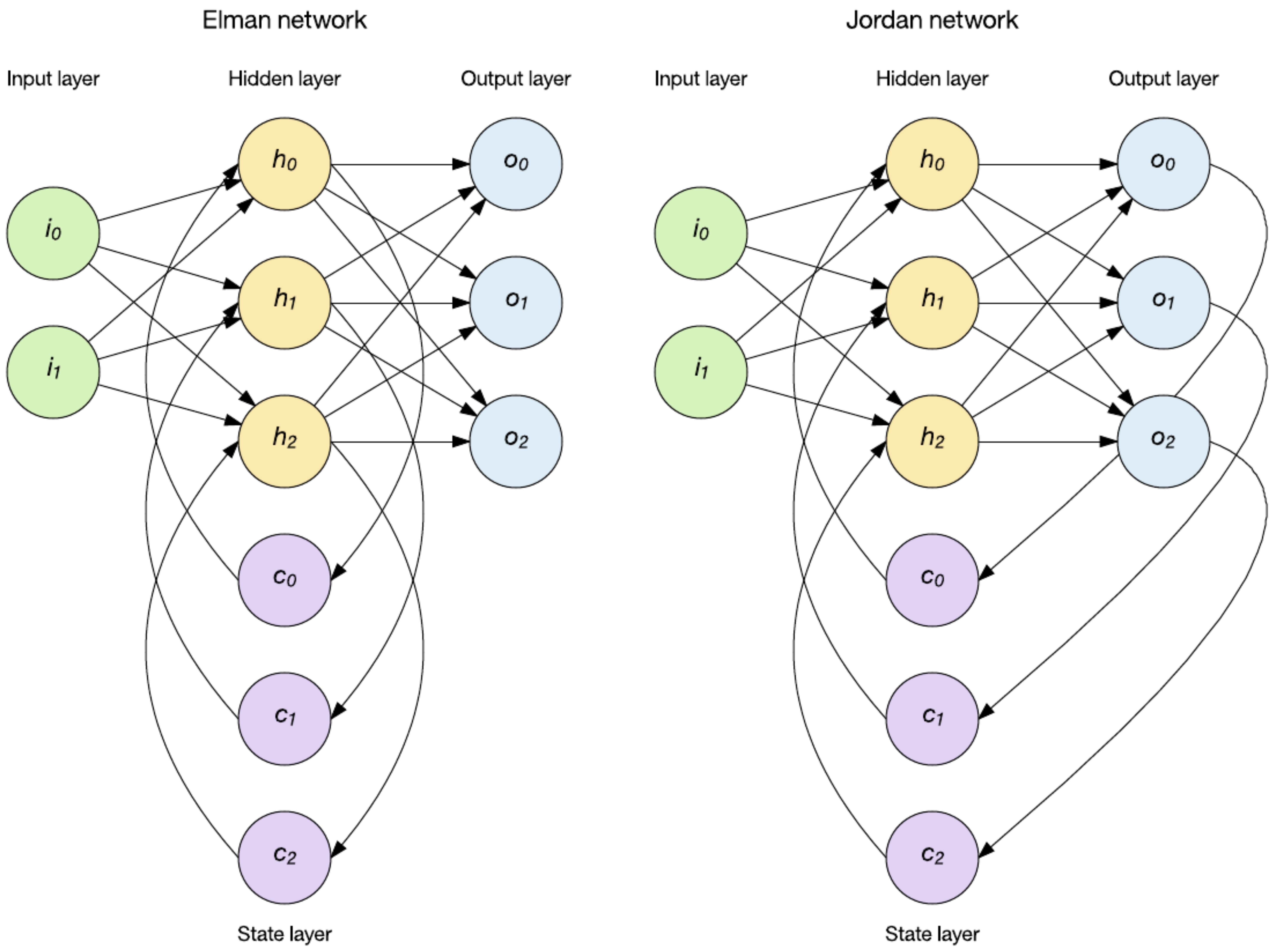
Generative Adversarial Networks



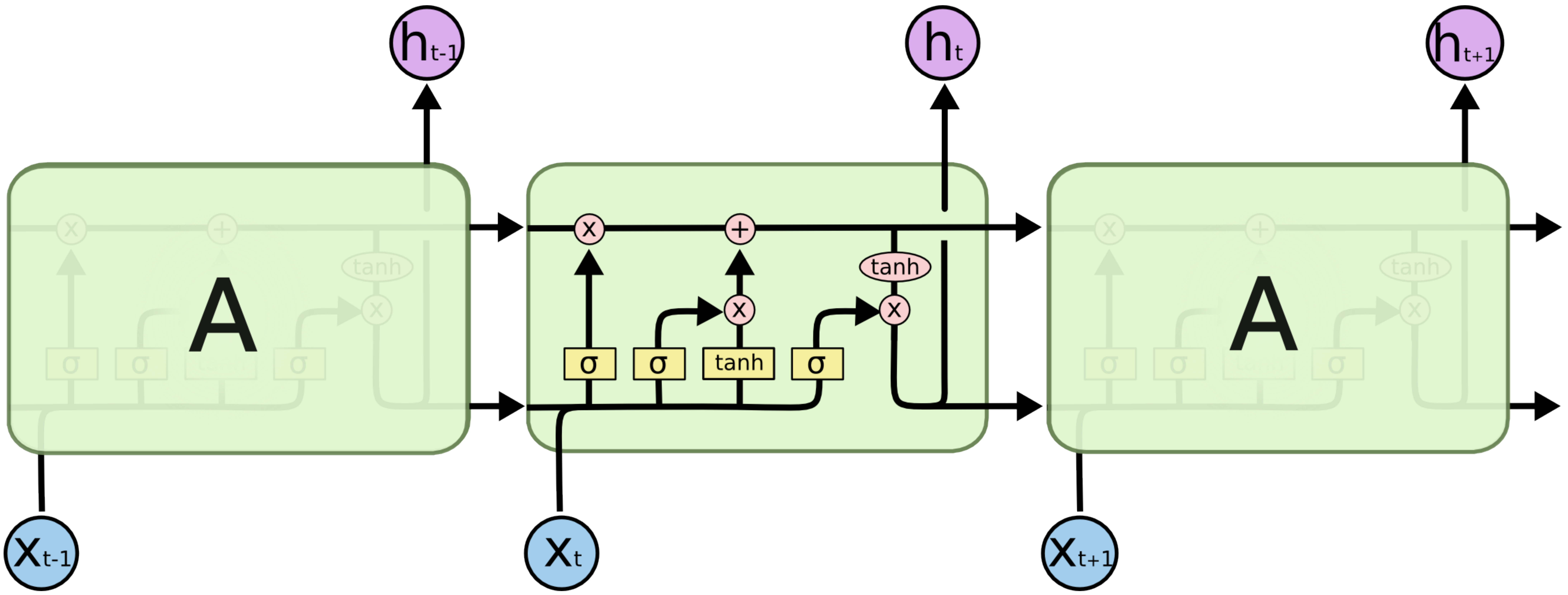
Recurrent neural networks



Recurrent neural networks



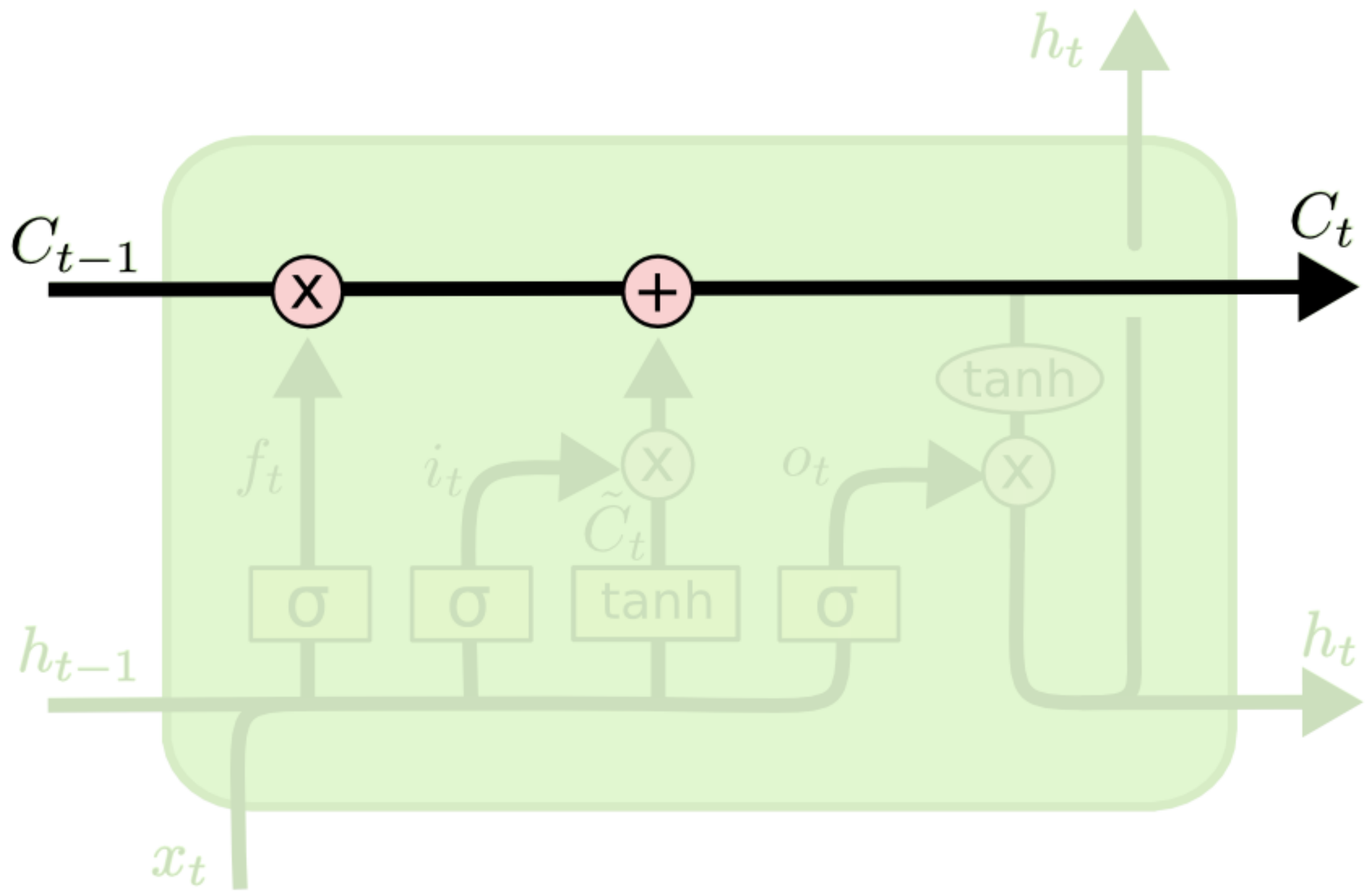
Long short-term memory (LTSM)



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

Long short-term memory (LTSM)

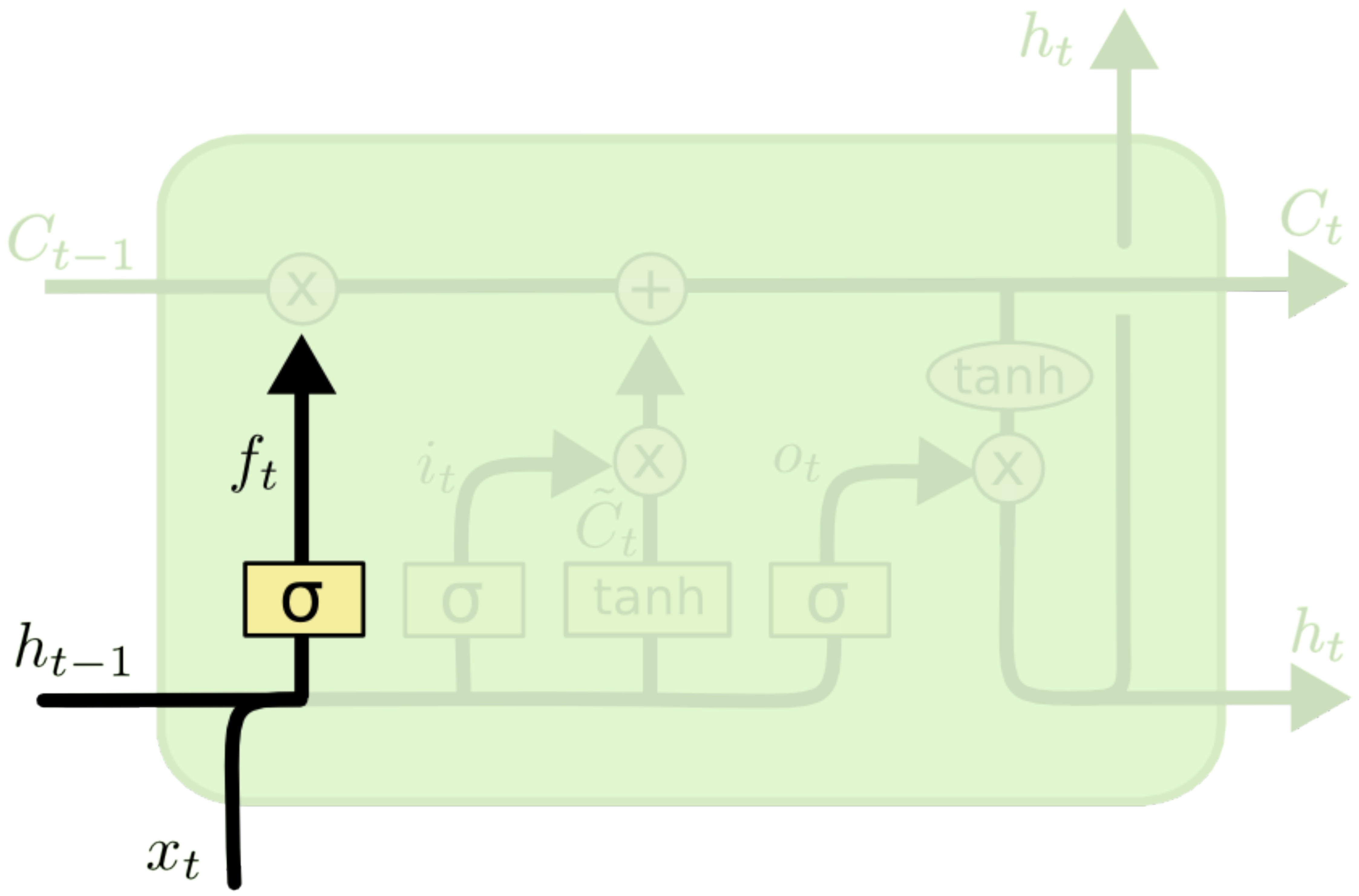
Cell state



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

Long short-term memory (LTSM)

Forget gate (how much is kept of the cell state)

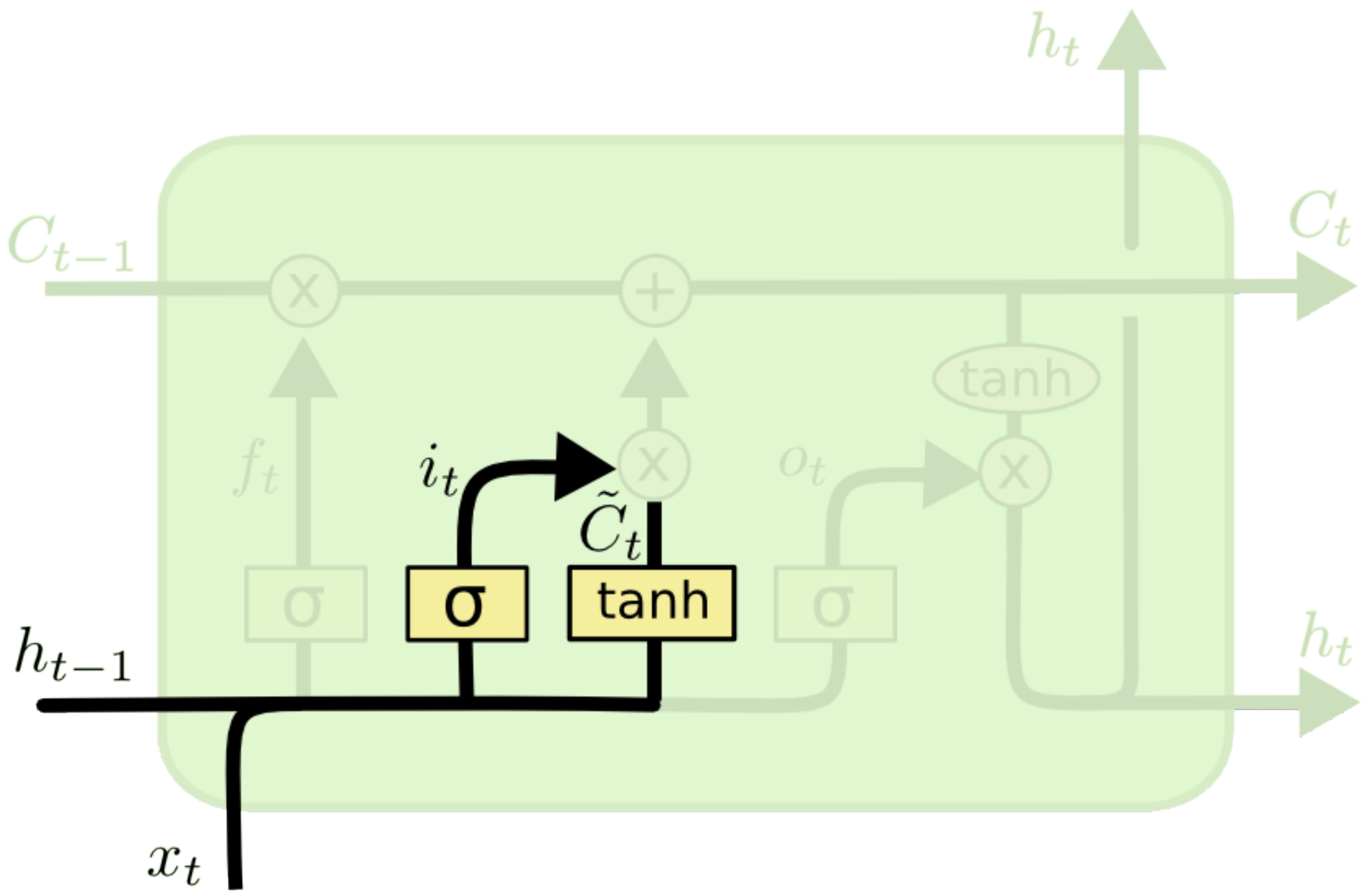


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

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

Long short-term memory (LTSM)

Input gate (how much is added to the cell state)



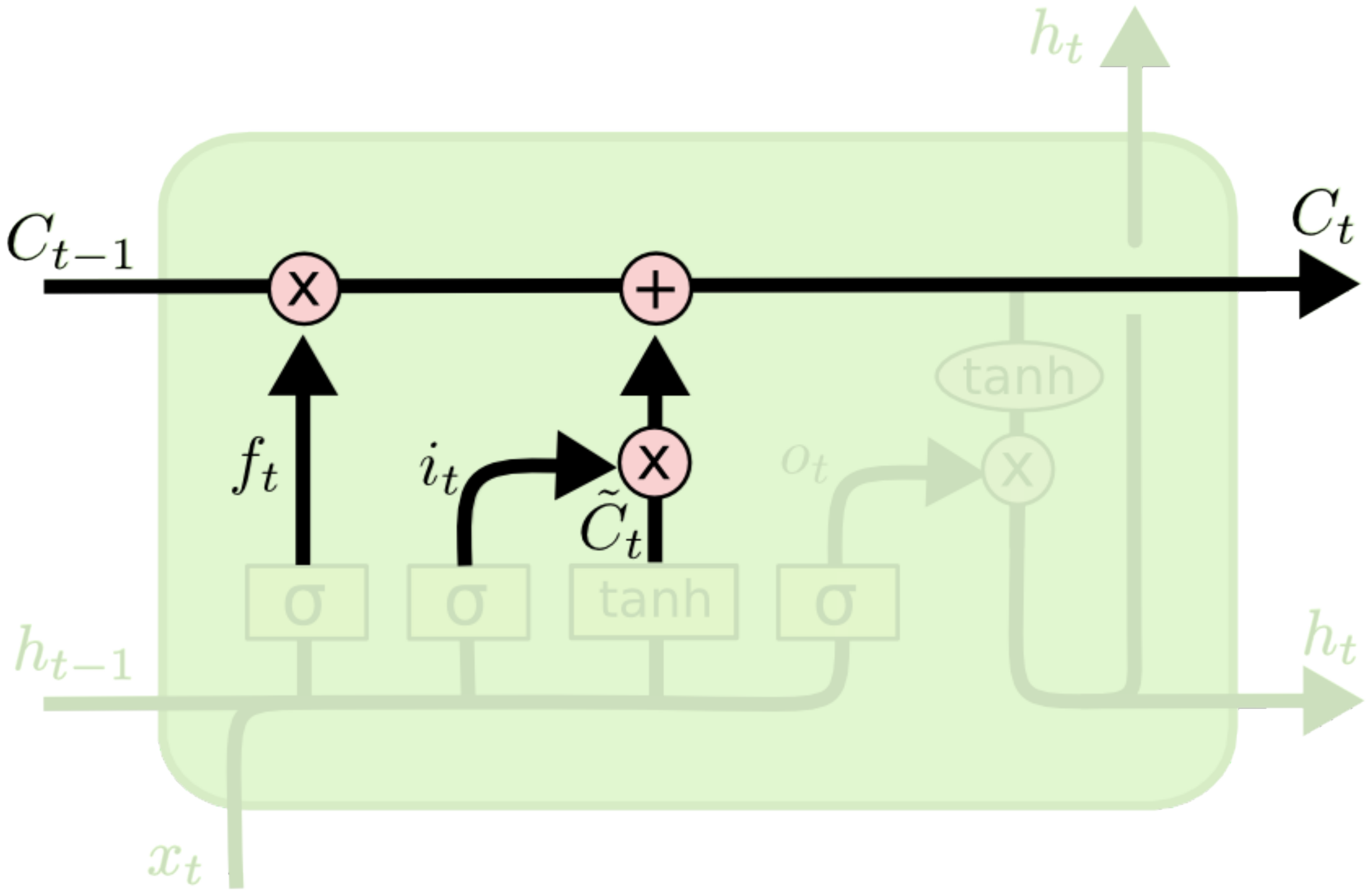
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

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

Long short-term memory (LTSM)

Update of the cell state

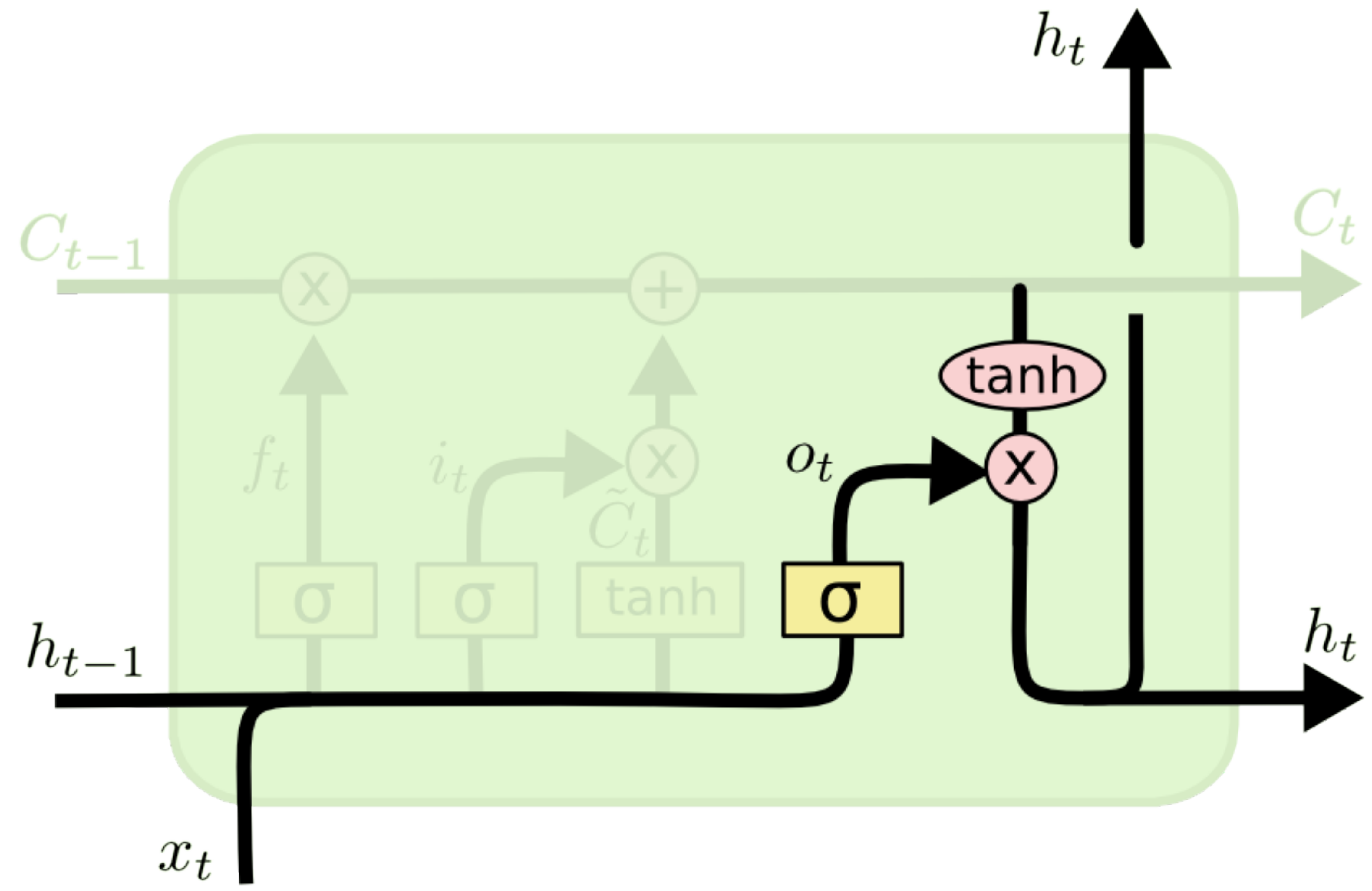


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

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

Long short-term memory (LTSM)

Output

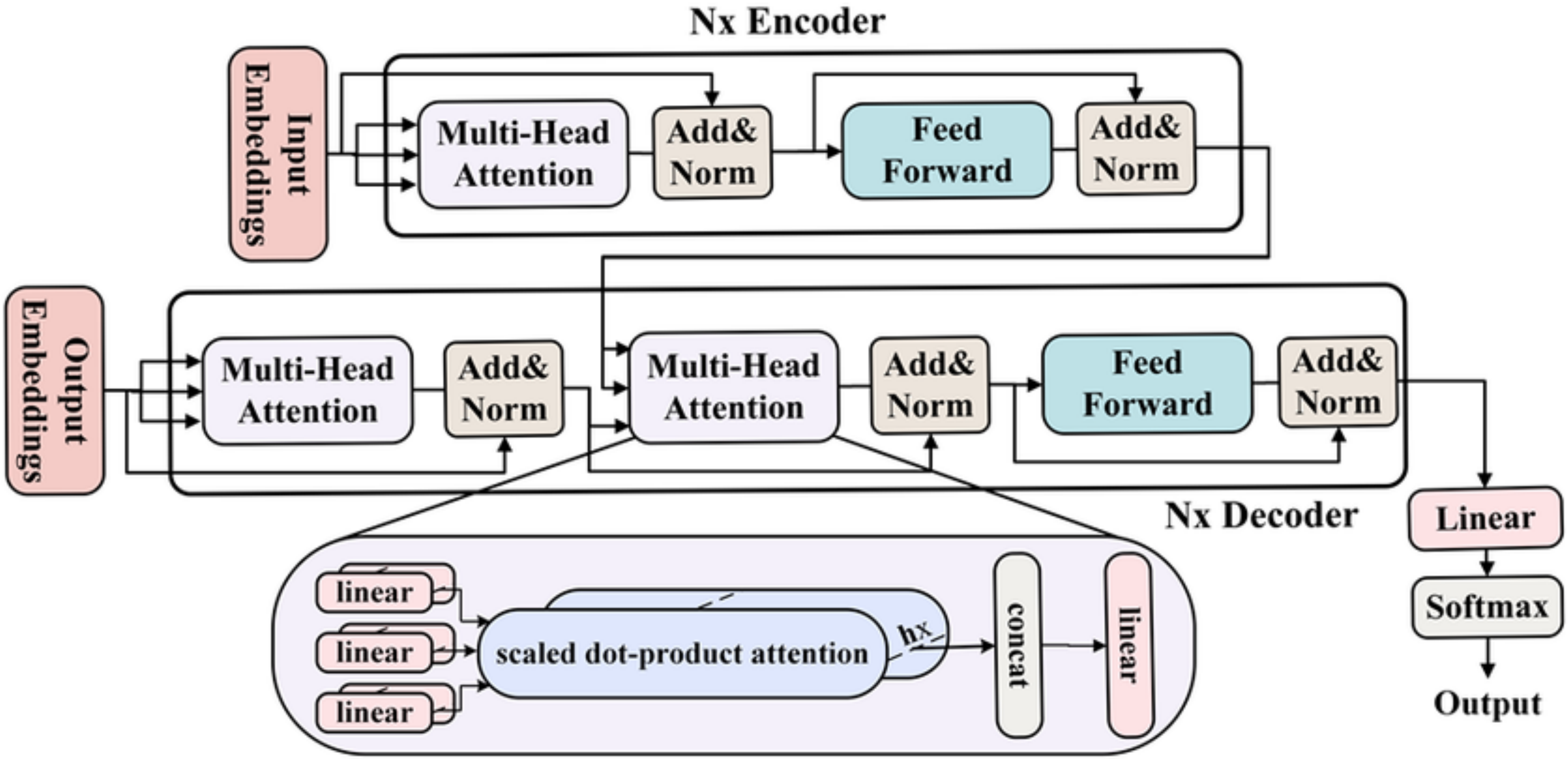


$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

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

Transformers



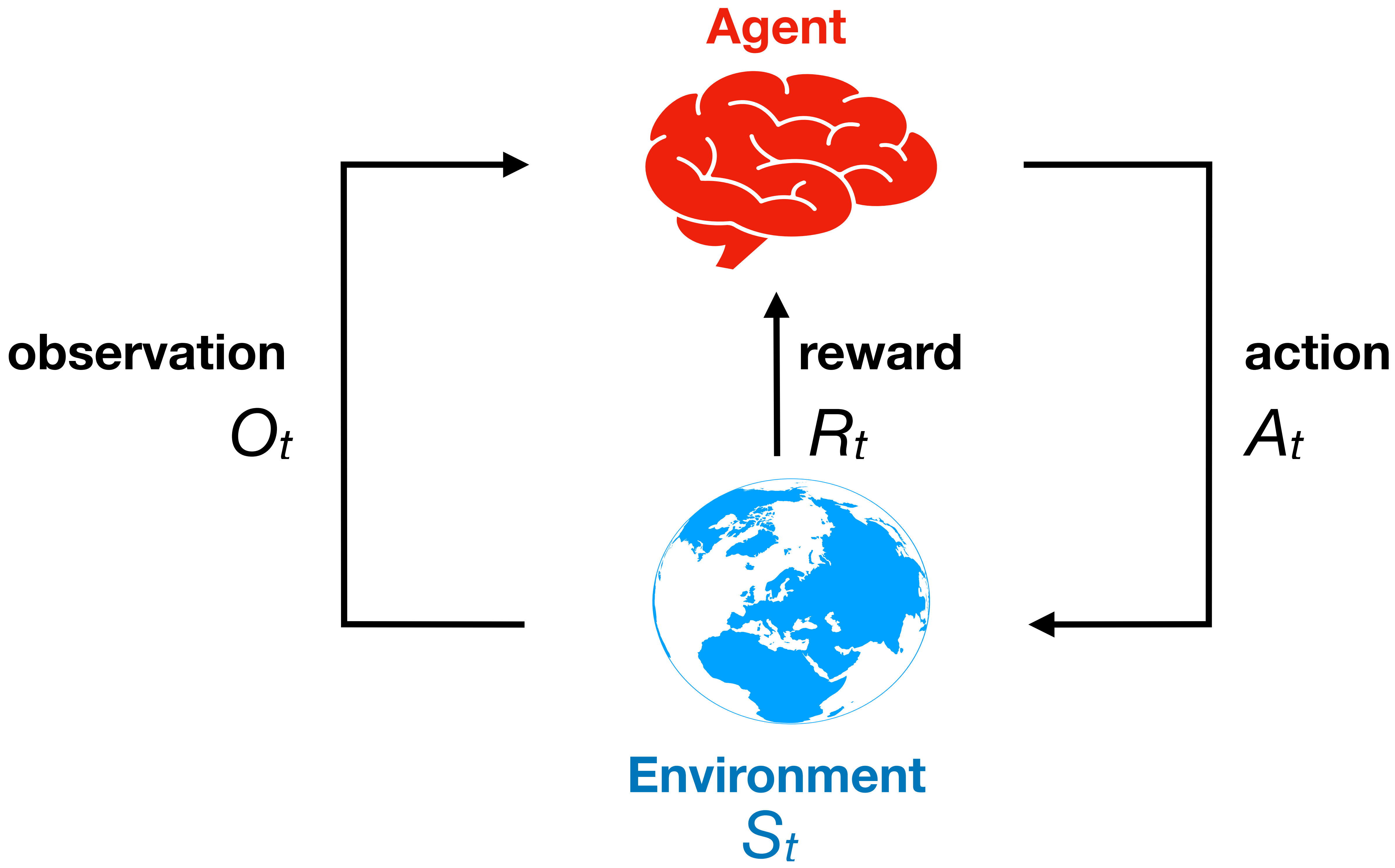
Reinforcement Learning

Agent

- Receives observation O_t
- Receives scalar reward R_t
- Executes action A_t

Environment

- Receives action A_t
- Changes state from S_t to S_{t+1}
- Emits observation O_{t+1}
- Emits scalar reward R_{t+1}



Goal

- Find policy $\pi(a \mid o) = \mathbb{P}\left[A_t = a \mid O_t = o\right]$
- Maximising future rewards

$$V_\pi(o) = \mathbb{E}_\pi\left[\sum_i \gamma^i R_{t+i} \mid O_t = o\right]$$

Reinforcement Learning

- **Goal:** select actions to maximise total future reward
- Actions may have long term consequences
- Reward may be delayed
- It may be better to sacrifice immediate reward to gain more long-term reward
- Examples: Blocking opponent moves (might help winning chances many moves from now); A financial investment (may take months to mature)