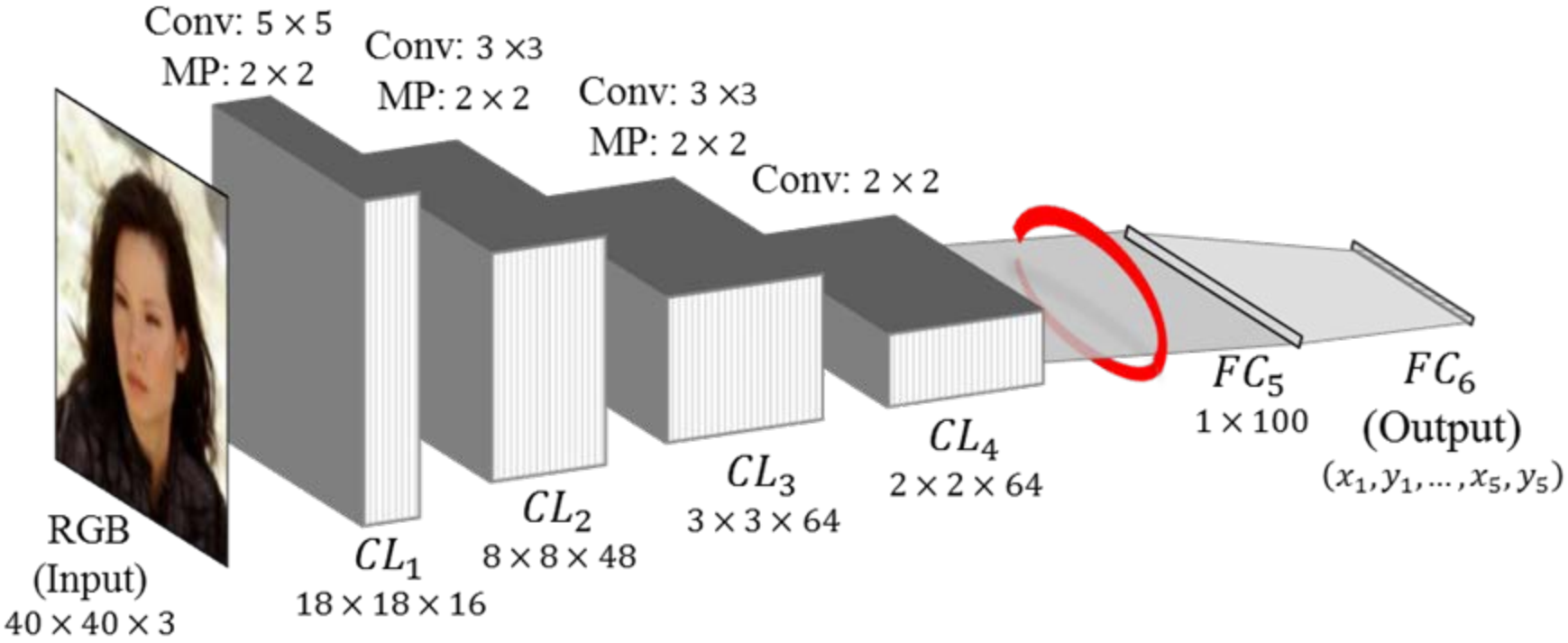
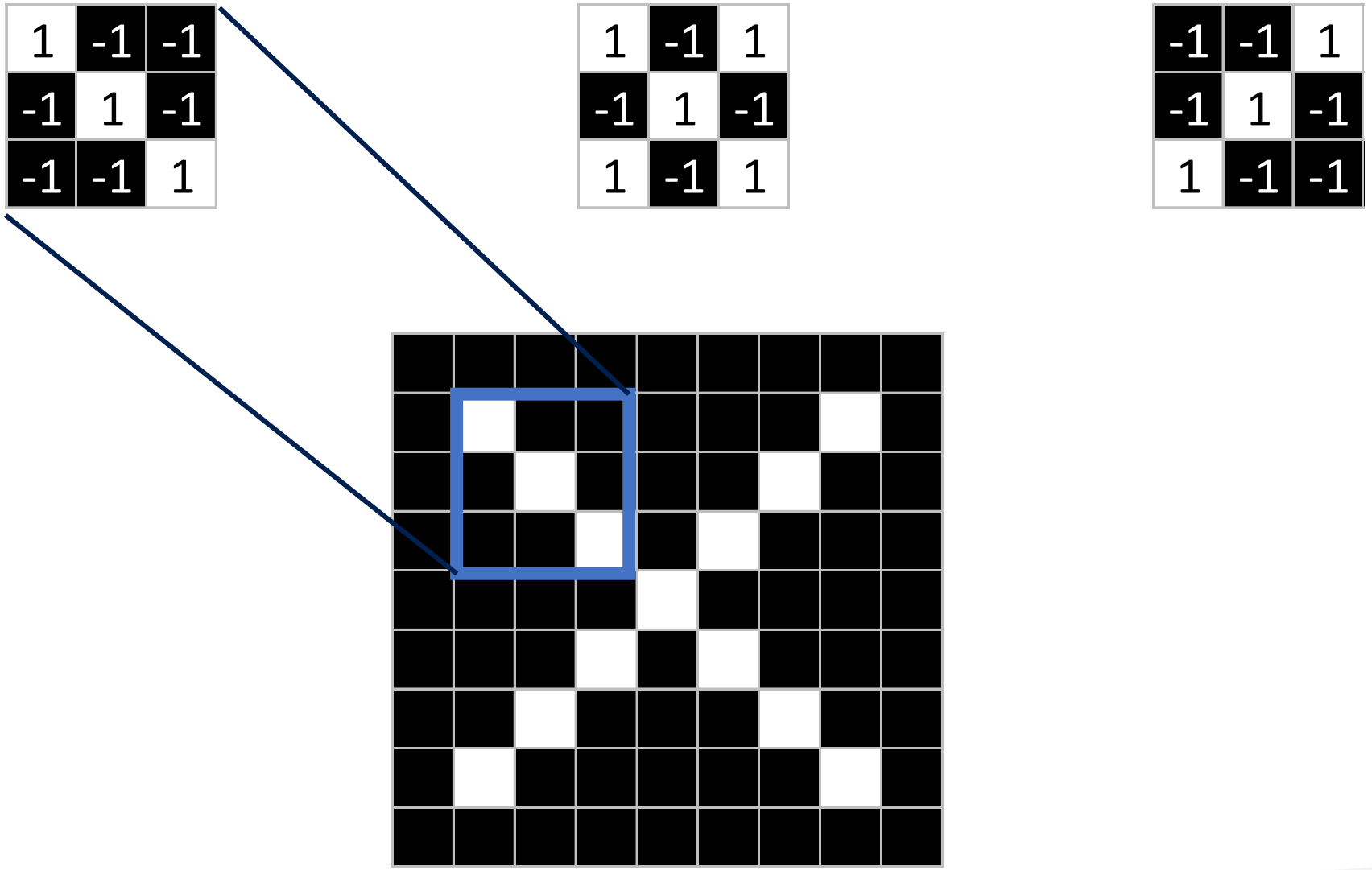
Modelling the time

Deep Neural Networks

ConvNets

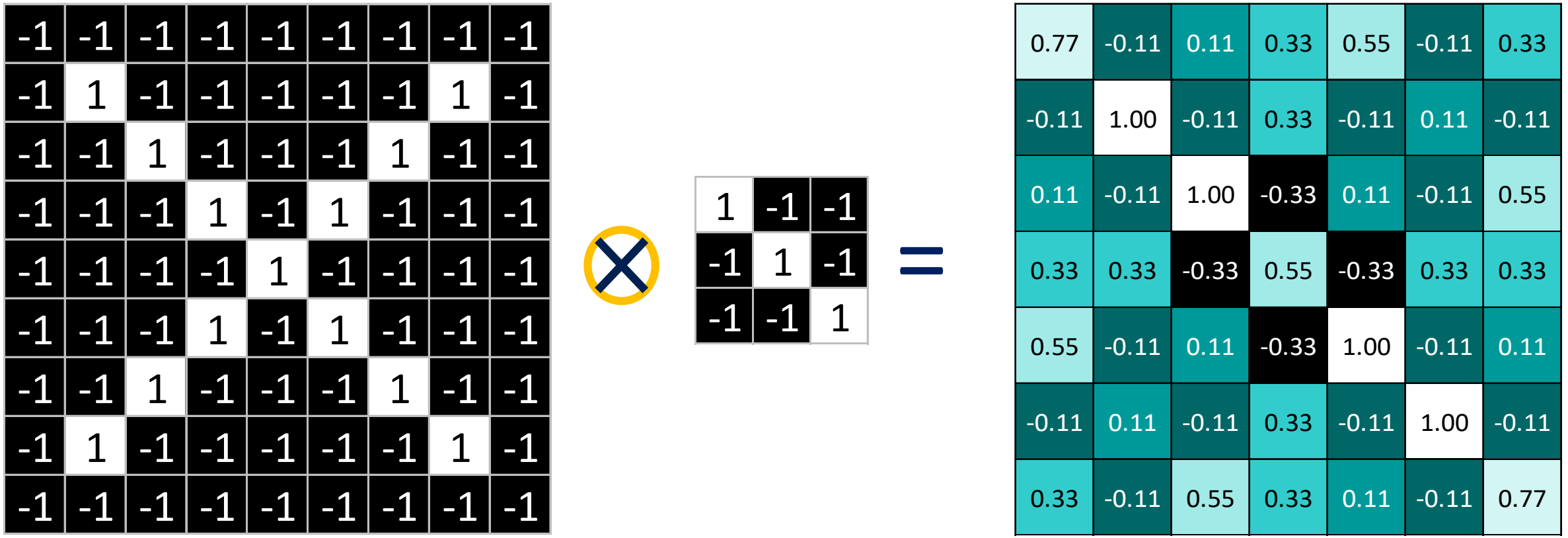

Conv: 5 × 5
MP: 2 × 2
Conv: 3 ×3
MP: 2 × 2
Conv: 3 ×3
MP: 2 × 2
Conv: 2 × 2
RGB
(Input)
40 × 40 × 3
CL_1
18 × 18 × 16
CL_2
8 × 8 × 48
CL_3
3 × 3 × 64
CL_4
2 × 2 × 64
FC_5
1 × 100
FC_6
(Output)
$(x_1, y_1, \ldots, x_5, y_5)$

Filters

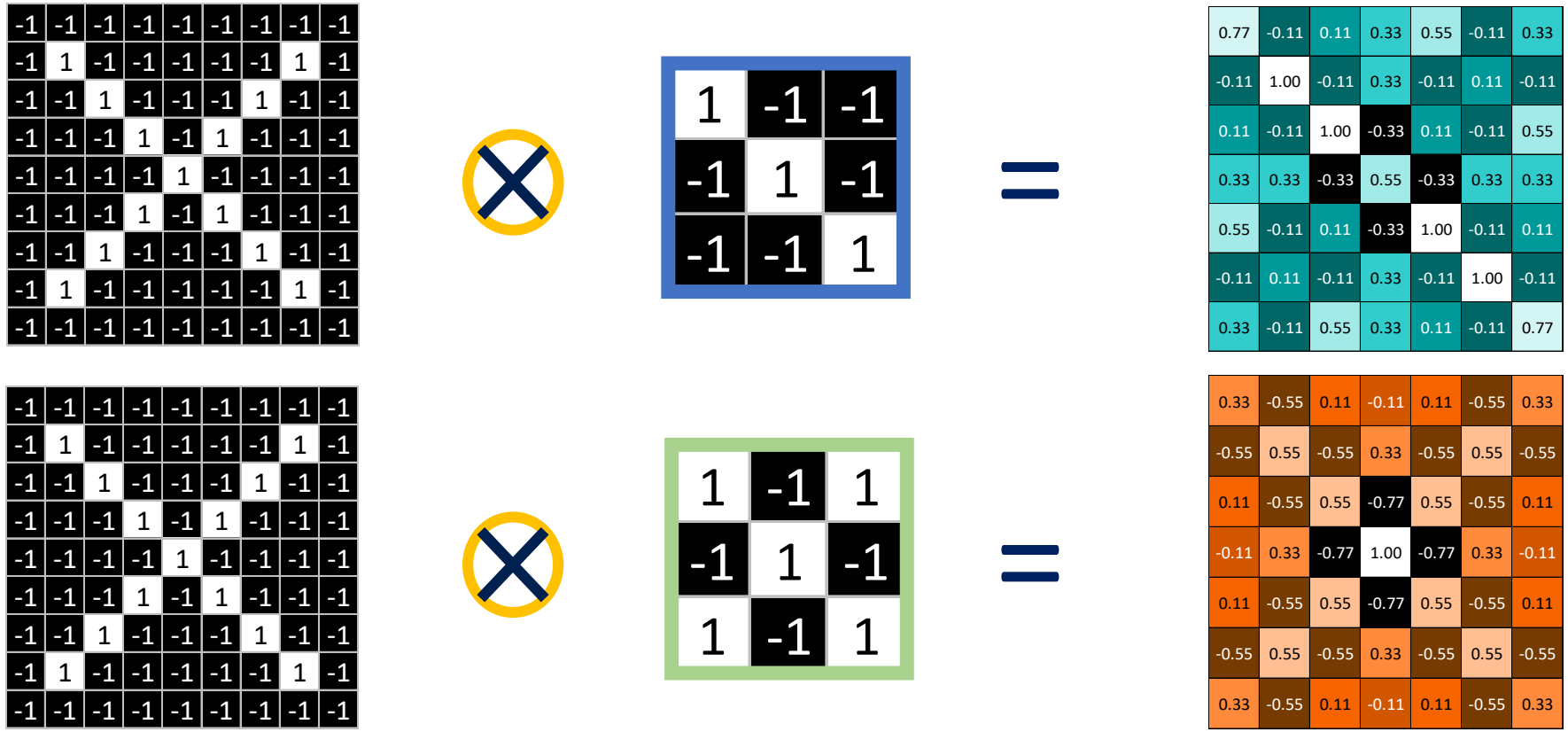


Slide credit to **Brandon Rohrer** - http://brohrer.github.io/how_convolutional_neural_networks_work.html

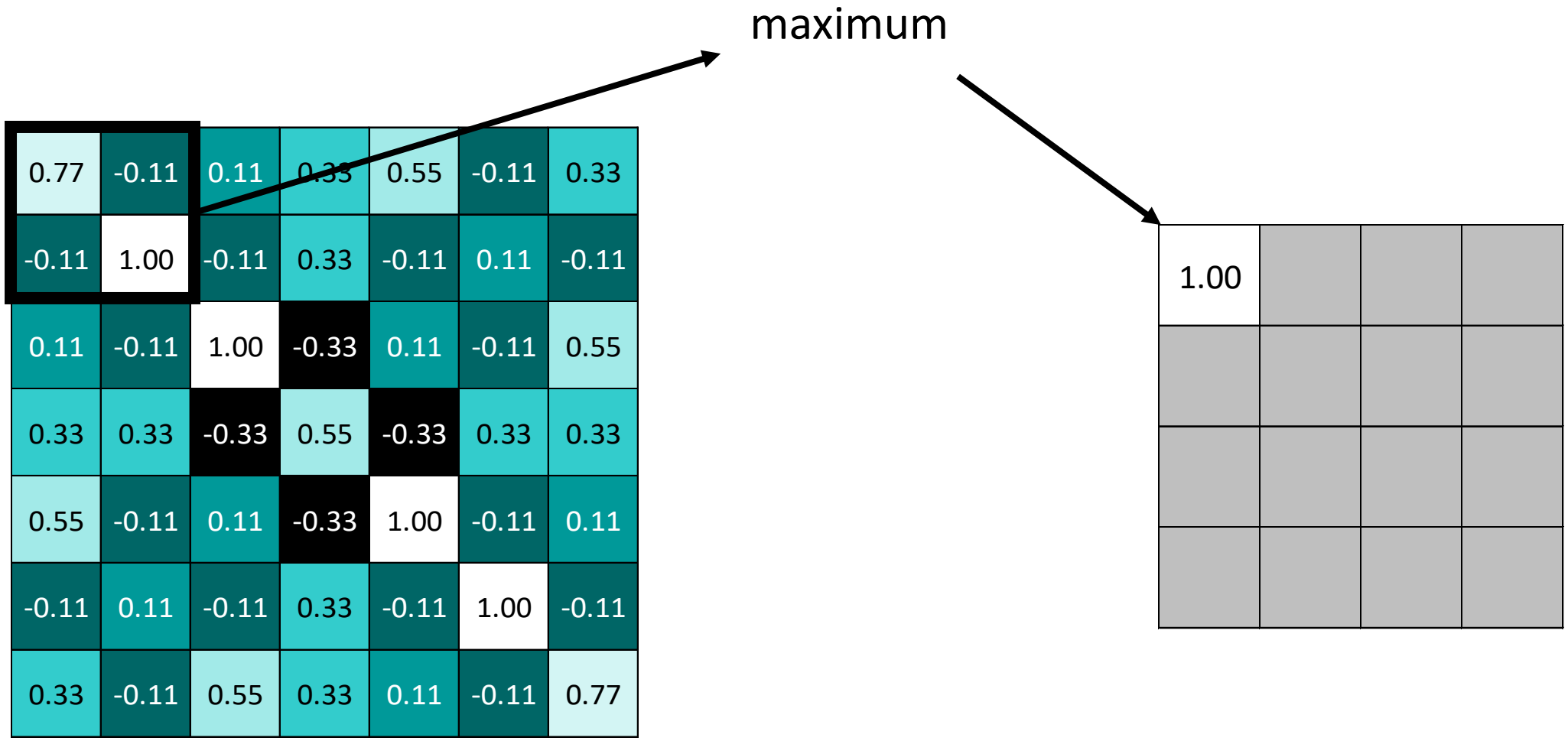
Convolutions



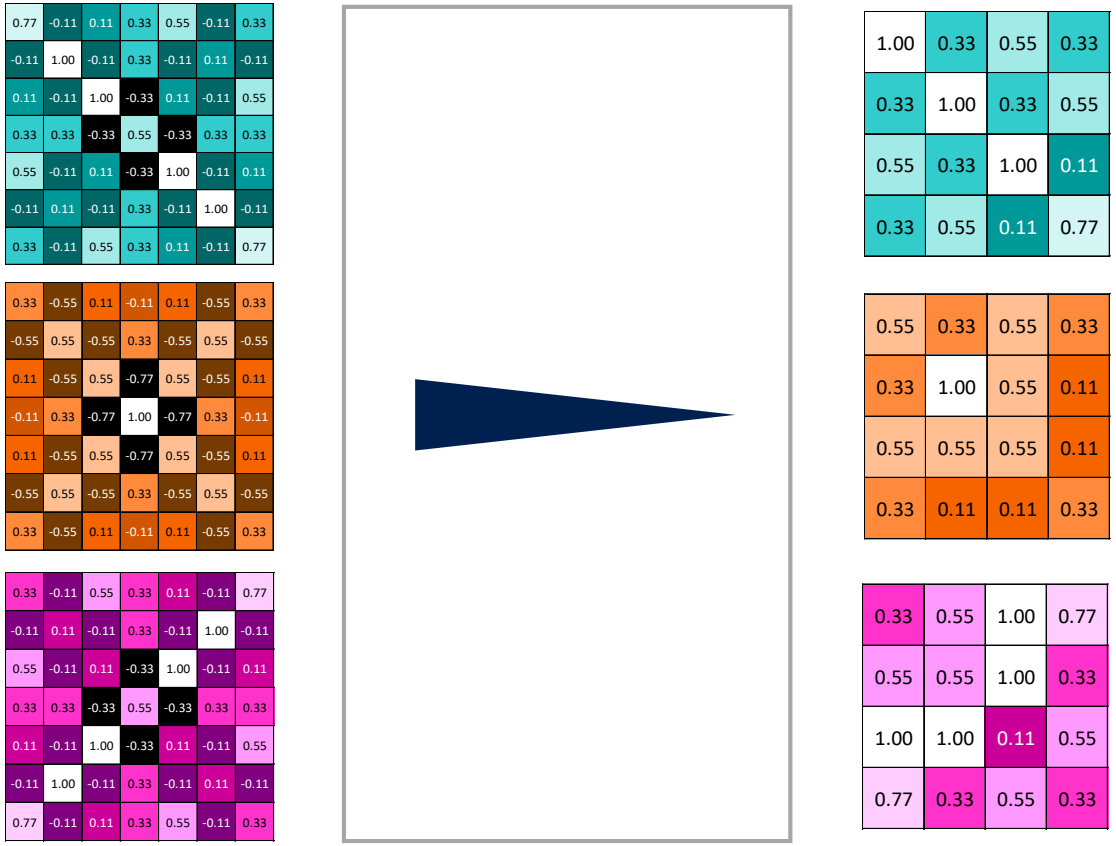
Feature Maps



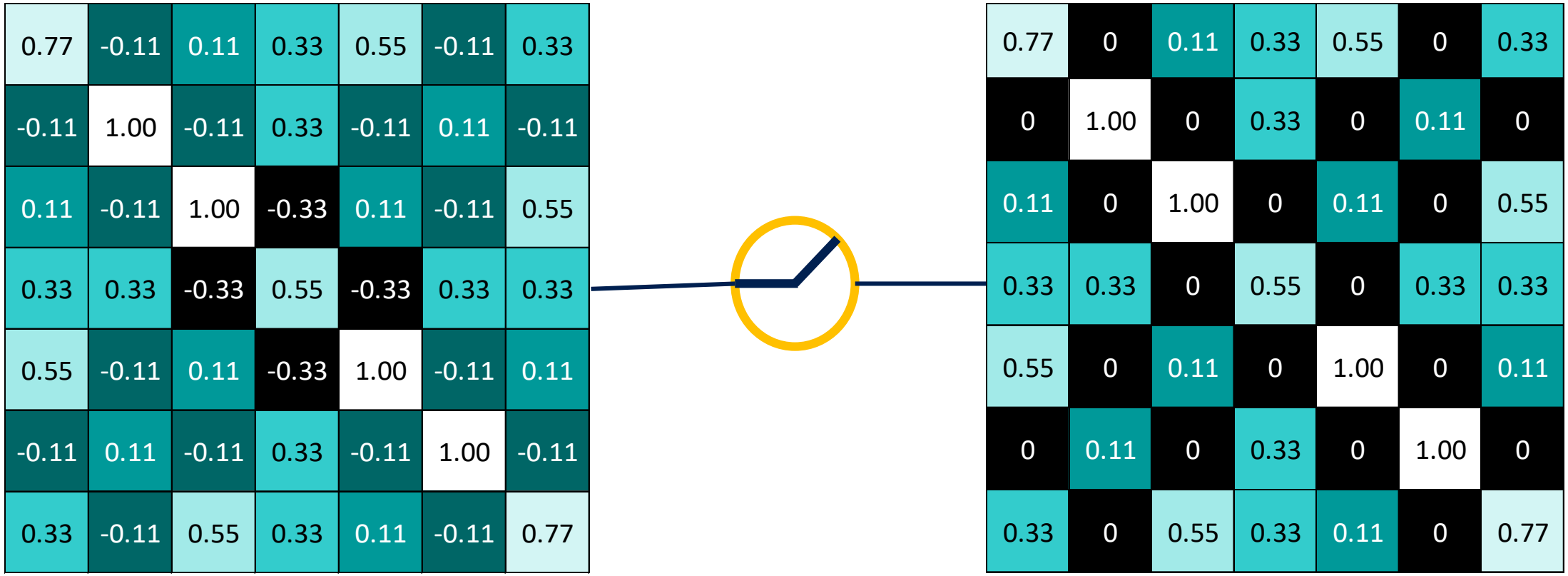
Pooling


maximum
0.77 -0.11 0.11 0.33 0.55 -0.11 0.33
-0.11 1.00 -0.11 0.33 -0.11 0.11 -0.11
0.11 -0.11 1.00 -0.33 0.11 -0.11 0.55
0.33 0.33 -0.33 0.55 -0.33 0.33 0.33
0.55 -0.11 0.11 -0.33 1.00 -0.11 0.11
-0.11 0.11 -0.11 0.33 -0.11 1.00 -0.11
0.33 -0.11 0.55 0.33 0.11 -0.11 0.77
1.00

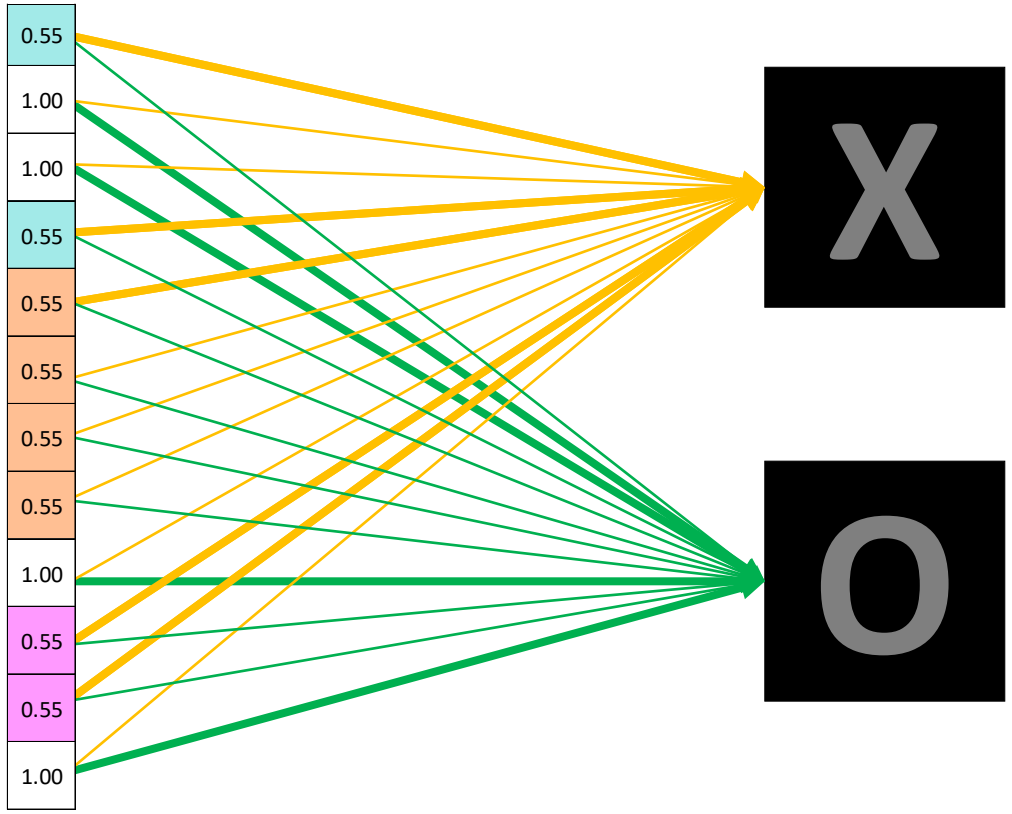
Pooling



Rectified Linear Unit (ReLu)



Fully Connected Layers



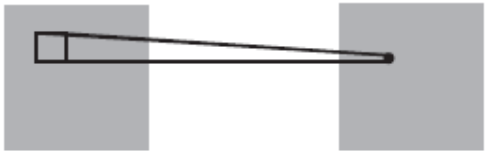
Slide credit to **Brandon Rohrer** - http://brohrer.github.io/how_convolutional_neural_networks_work.html

3D Convnet

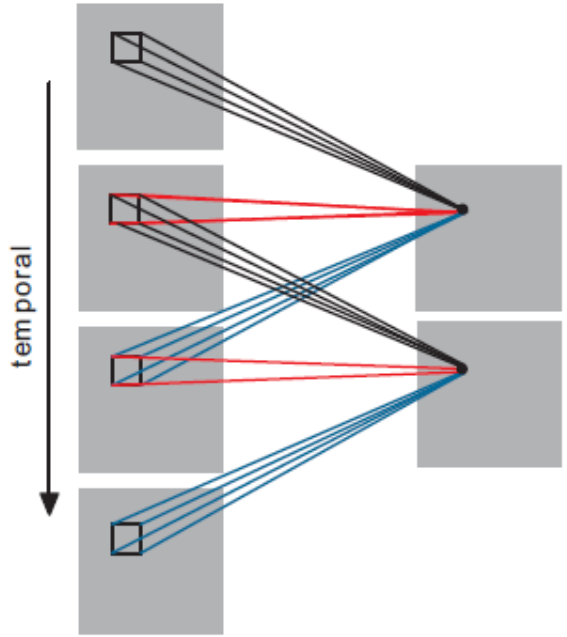
The 3D convolution is achieved by convolving a 3D kernel to the cube formed by stacking multiple contiguous frames together.

By this construction, the feature maps in the convolution layer is connected to multiple contiguous frames in the previous layer, thereby capturing motion information.

2D Convolution



3D Convolution



Ji, Shuiwang, et al. "3D convolutional neural networks for human action recognition." *IEEE transactions on pattern analysis and machine intelligence* 35.1 (2013): 221-231.

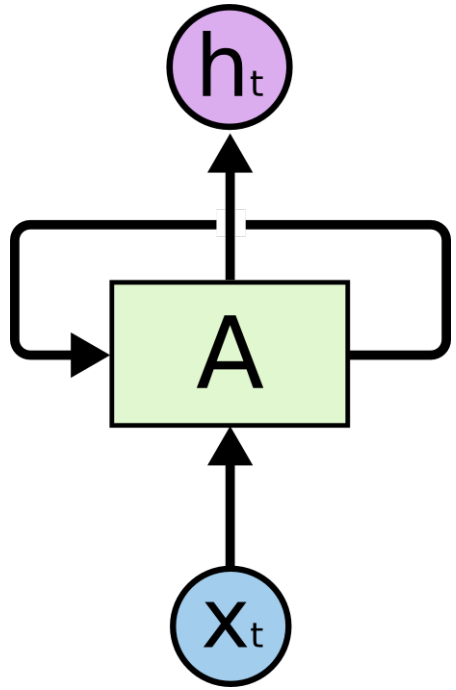
Main intuition

Humans don’t start their thinking from scratch!

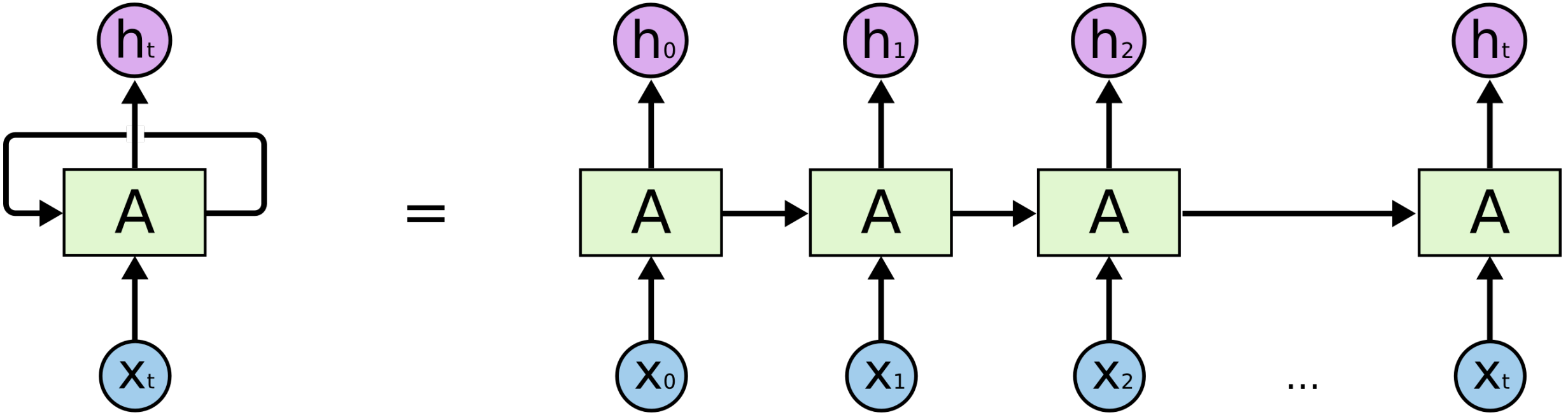
Thoughts have persistence

Recurrent Neural Networks have loops

In the diagram, a chunk of neural network, $\boldsymbol{A}$, looks at some input x_i and outputs a value h_t. A loop allows information to be passed from one step of the network to the next.



An Unrolled Recurrent Neural Network



Issues

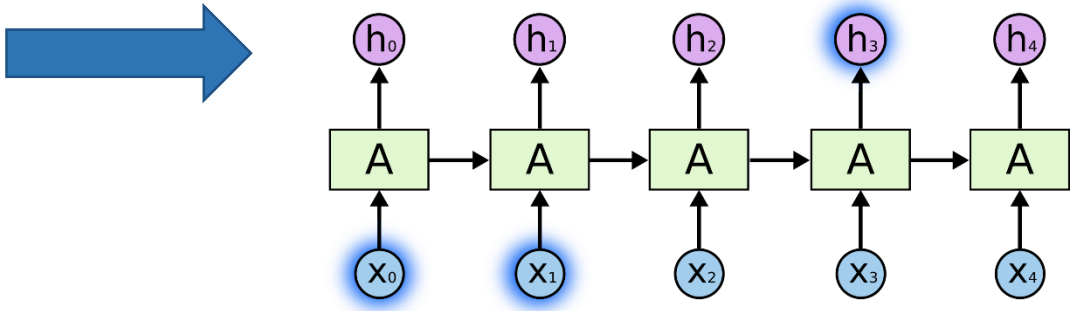
Can RNN connect previous information to the present task?

- Sometimes the present information is sufficient, a regular RNN would just smooth the decision along time.

Issues

- Consider a language model in which the network is trying to predict the next word in the sentence:

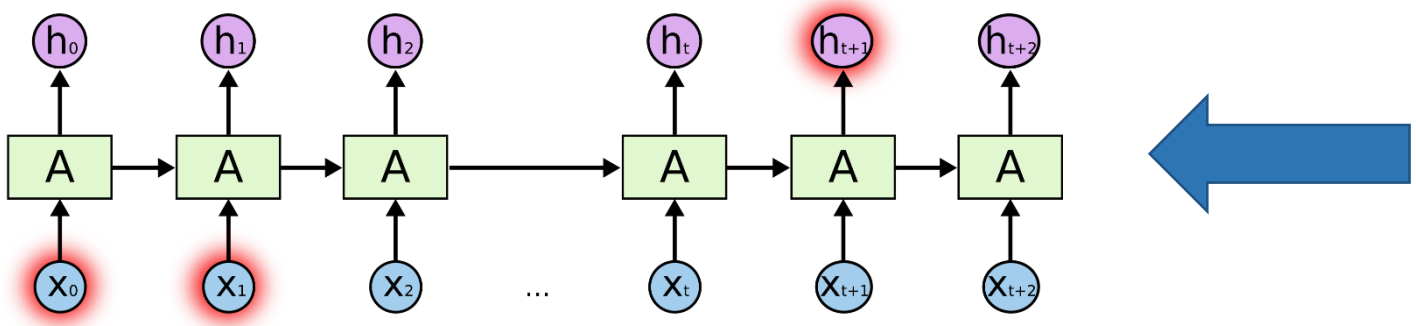
  "the cloud are in the ***sky***"



- Sometimes more context is needed:

  "I grew up in Italy… I speak fluent ***Italian***."

- In the latter case, recent information suggests that the next word should be the name of a language, but only the word ***Italy*** is informative in order to decide what language I am speaking fluently.



This becomes complicated when the gap is very large.

LSTM Networks

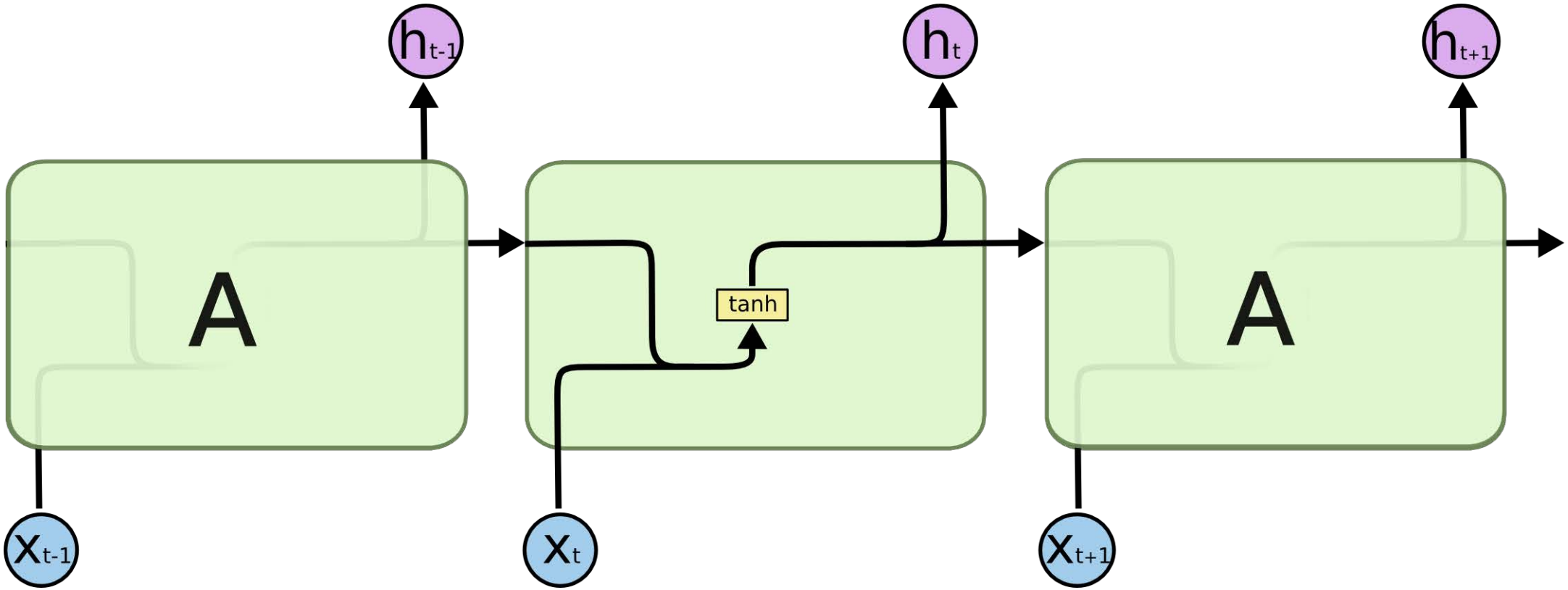
- LSTM – Long-Short Term Memory Networks are special kind of RNN, capable of learning long-term dependencies.[1]

Motivation:

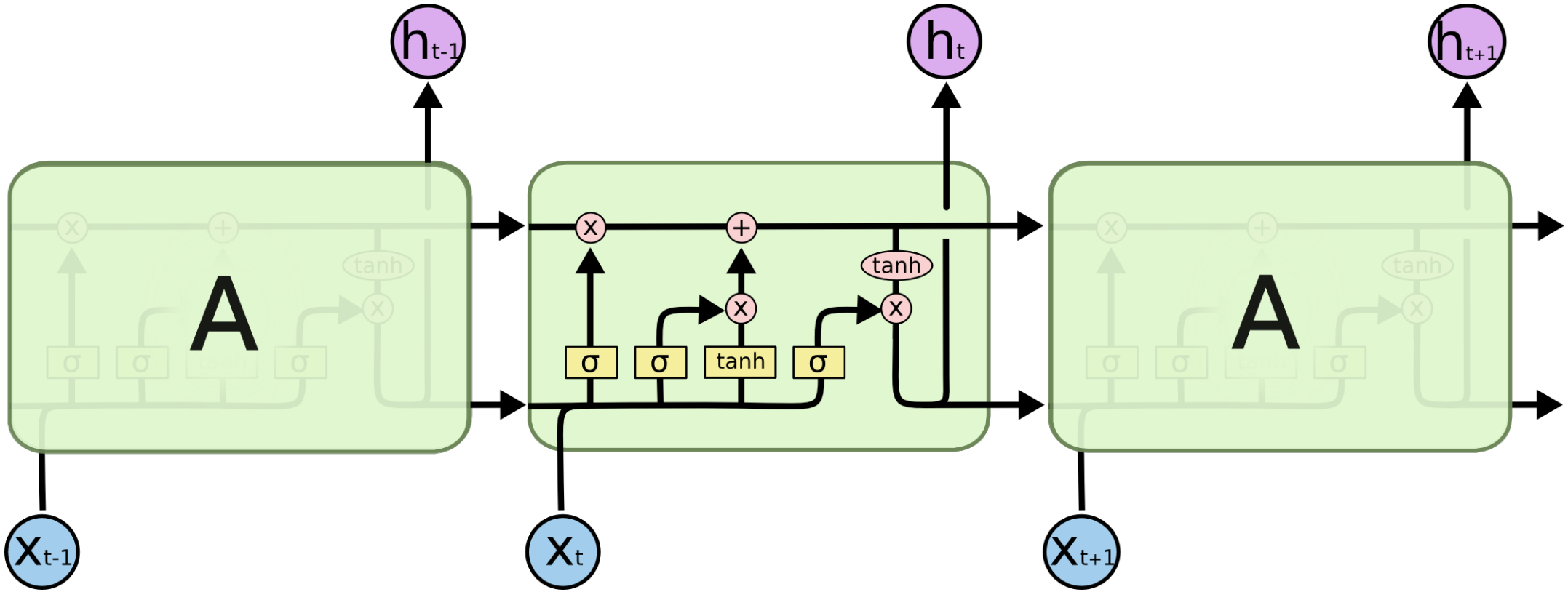
- To avoid the long-term dependency problem.
  - Remembering information for long periods of time is their default behaviour, not something they struggle to learn!

(1) Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780.

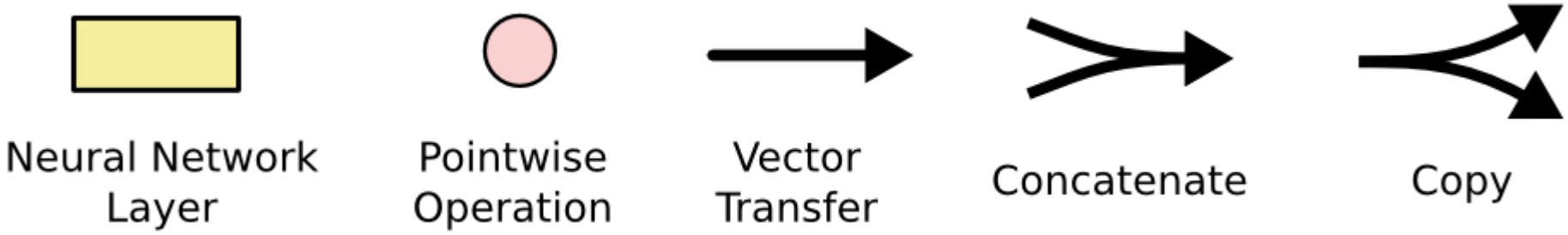
LSTM Networks vs Traditional RNN



LSTM Networks vs Traditional RNN

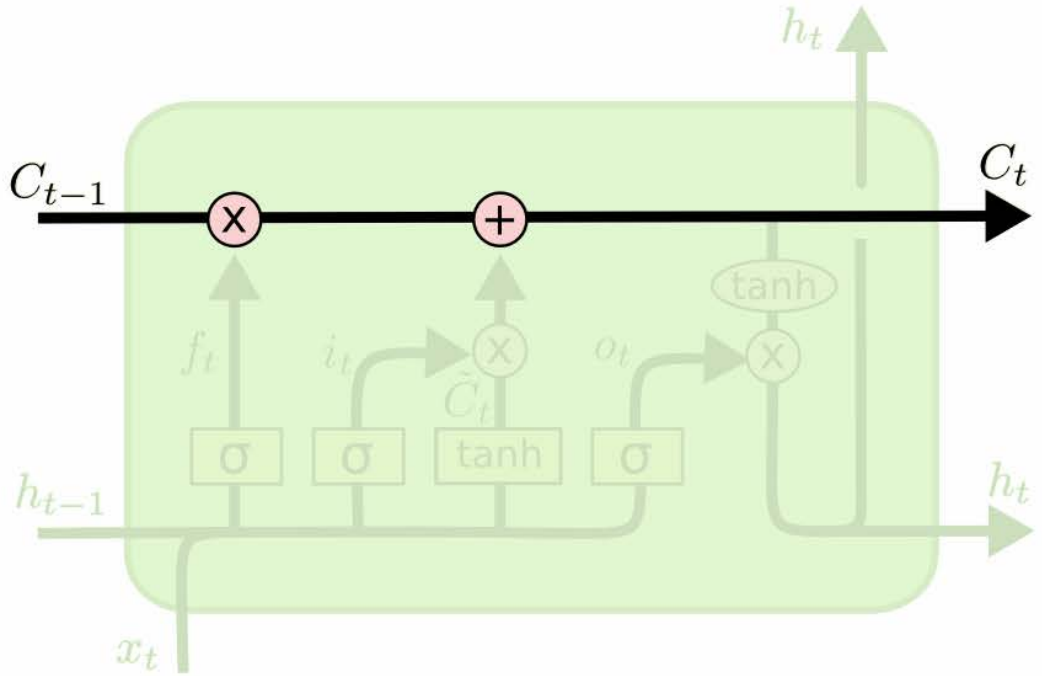


Notation


Neural Network Layer
Pointwise Operation
Vector Transfer
Concatenate
Copy

The cell state line

- The core of LSTM is the **cell state,** the horizontal line running through the top of the diagram



Gates

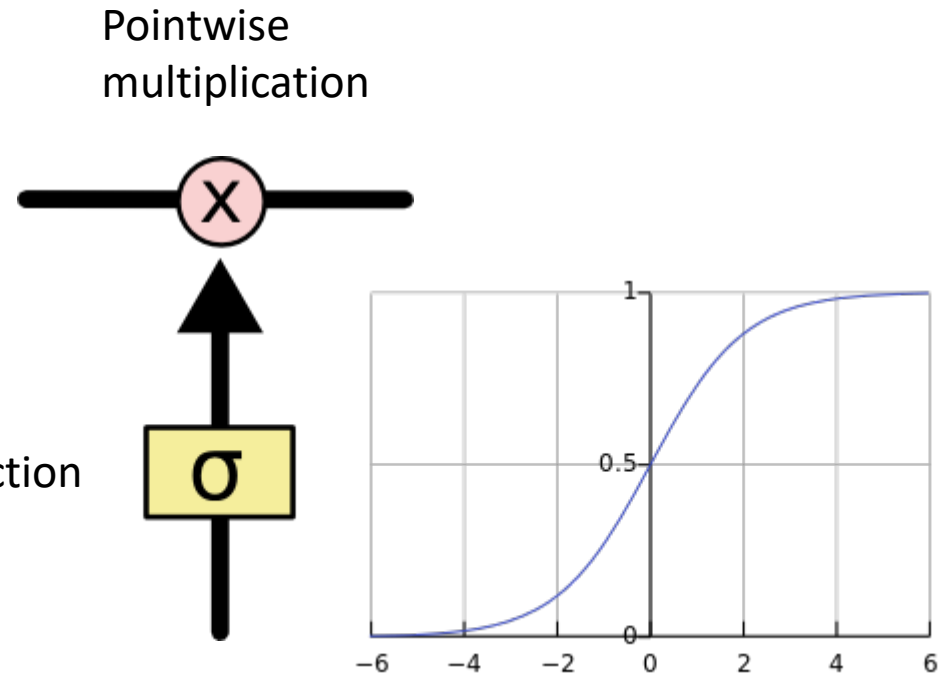
Gates are a way to optionally let information through.

Function:

- 0 means “nothing goes through”
- 1 means “let everything through”

Motivation:

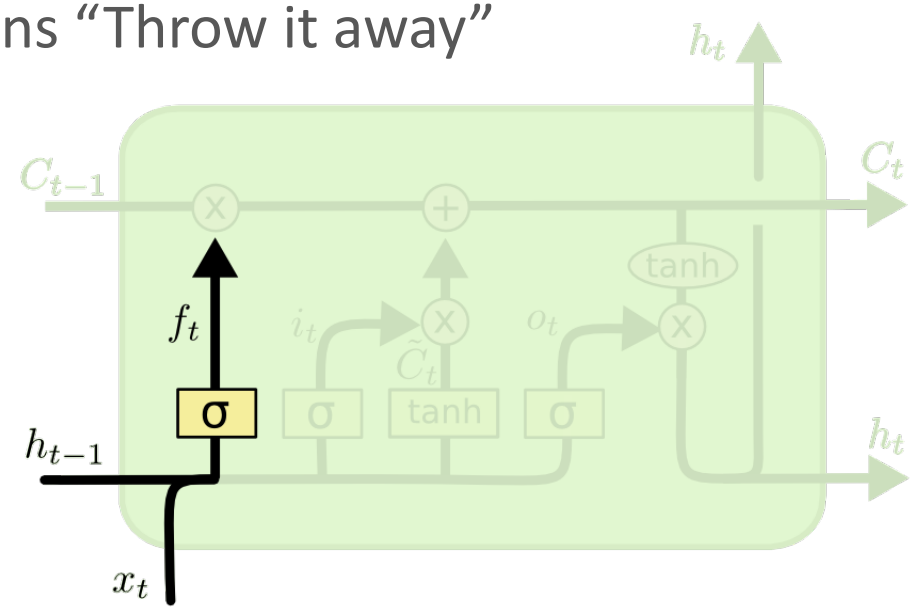
- A LSTM has three of these gates, to protect and control the cell state



Forget Gate Layer

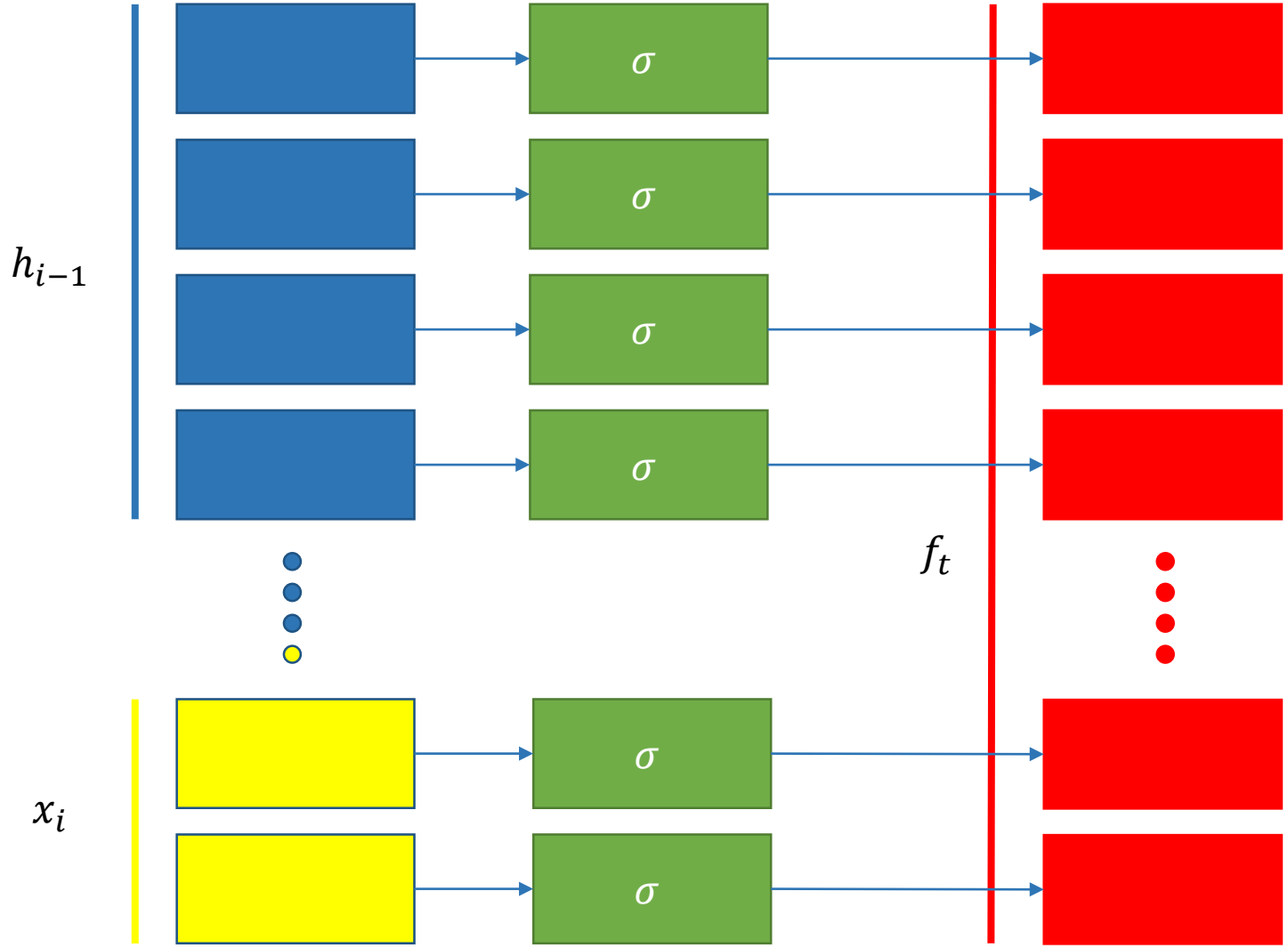
What information we are going to throw away of the previous cell?

- It looks at h_{t-1} and x_t, and outputs a number between 0 and 1 for each number in the cell state C_{t-1}.
  - 1 means “Completely keep this”
  - 0 means “Throw it away”



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

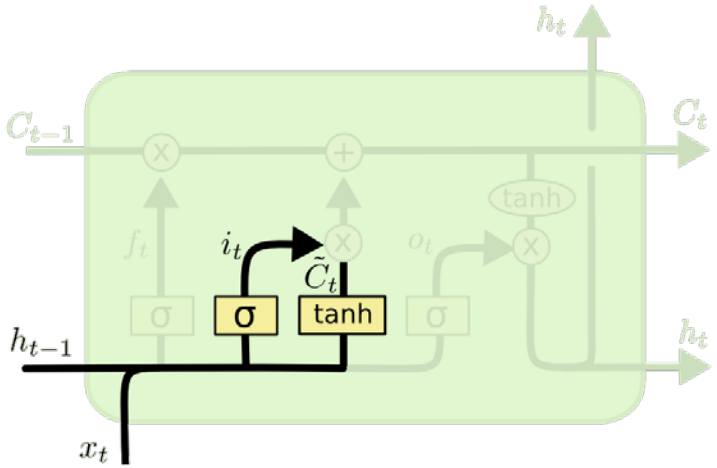
Data input remark



Input cell

- What is the information we are going to store in the cell state?

- It is composed by 2 parts:
  1. A sigmoid layer called "*input layer*" decides which values to update.
  2. A tanh layer that creates new candidate values, $\tilde{C}_t$, that could be added to the state.

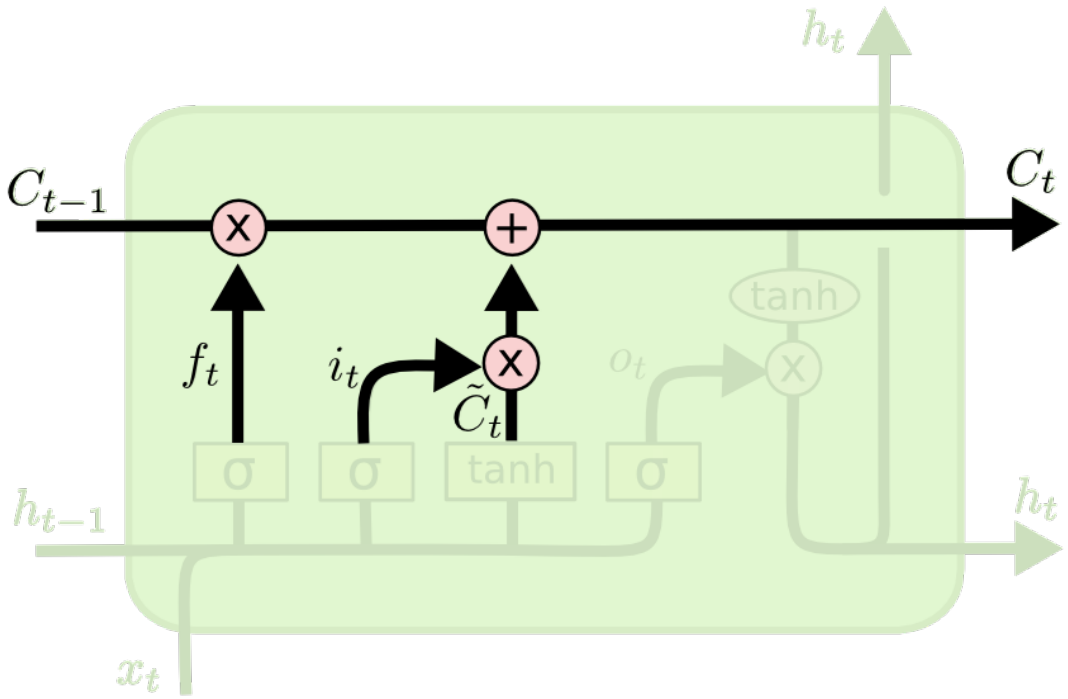


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

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

Update the cell

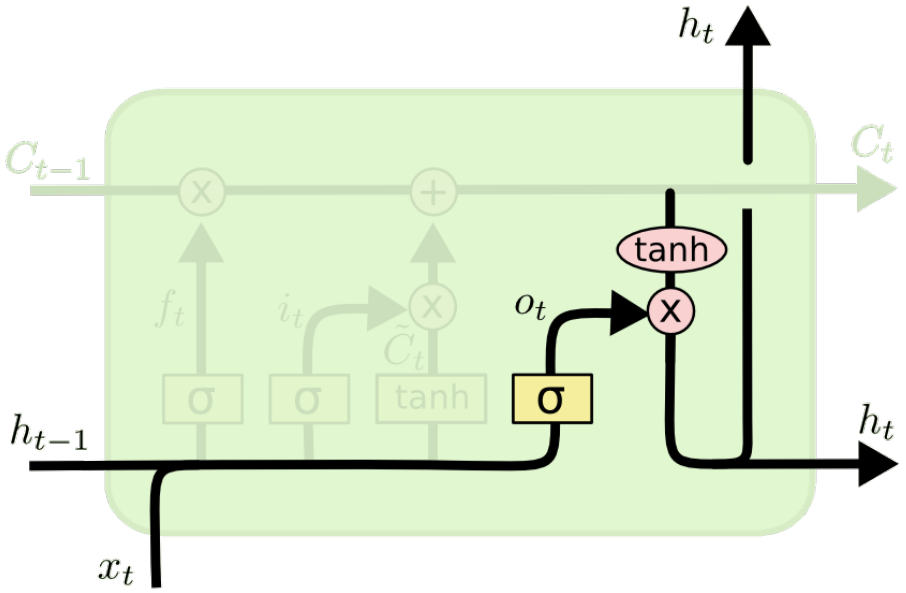
- It is time to update the cell state, C_{t-1}, into the new cell C_t.



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

Output

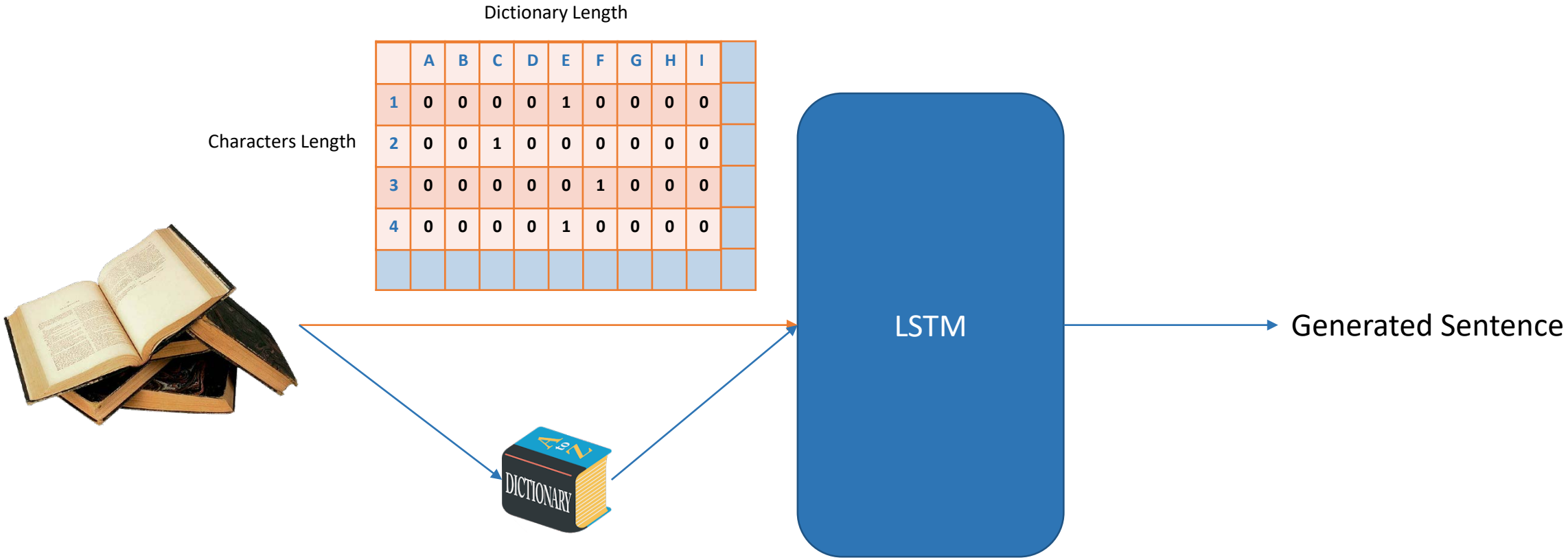
- The output will be based on our cell state, but will be a filtered version.



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

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

LSTM for text generation

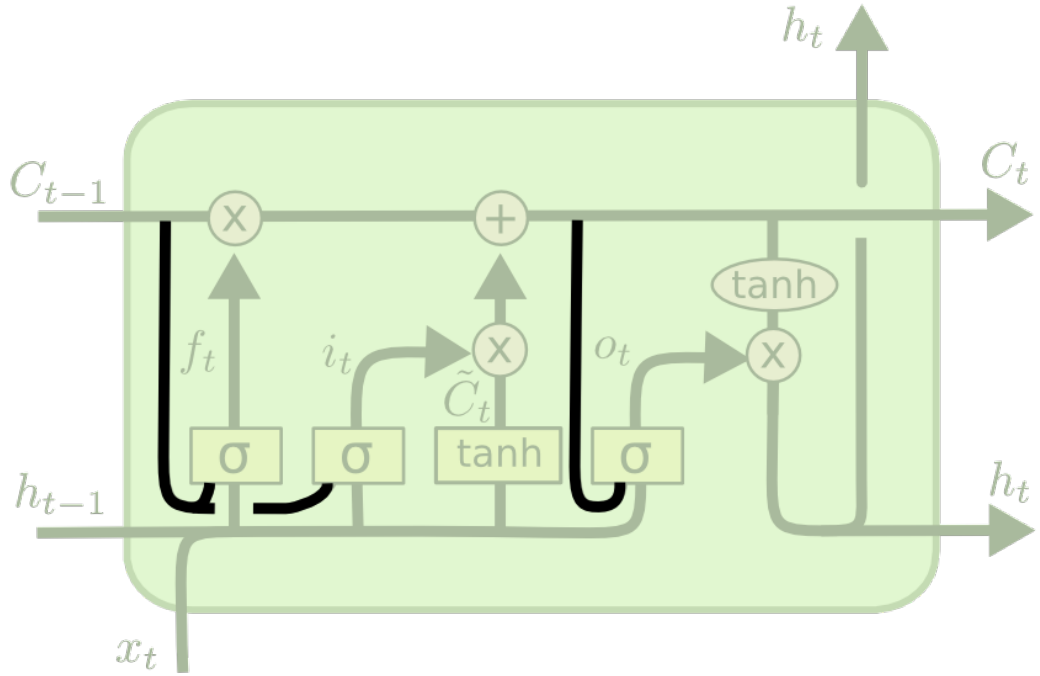


Example

NLP application

Variants

- Peephole connections
  - More info is accessible to forget and input layer

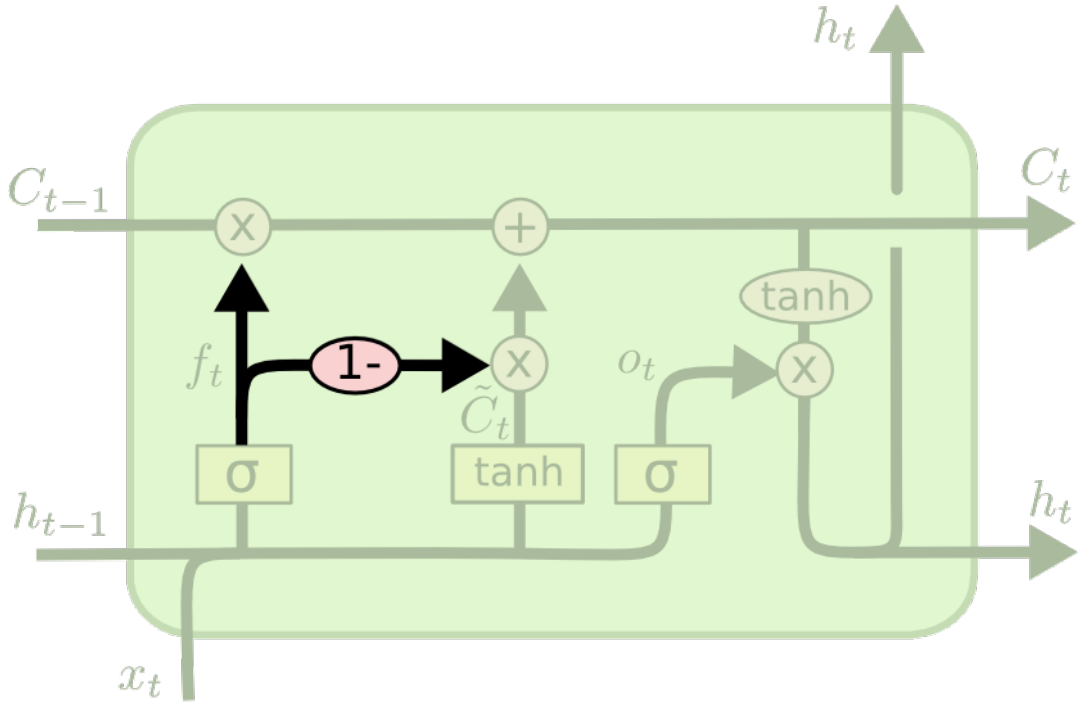


$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_i\right)$$
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] \; + \; b_o\right)$$

Gers, Felix A., and Jürgen Schmidhuber. "Recurrent nets that time and count." *Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on*. Vol. 3. IEEE, 2000.

Variants

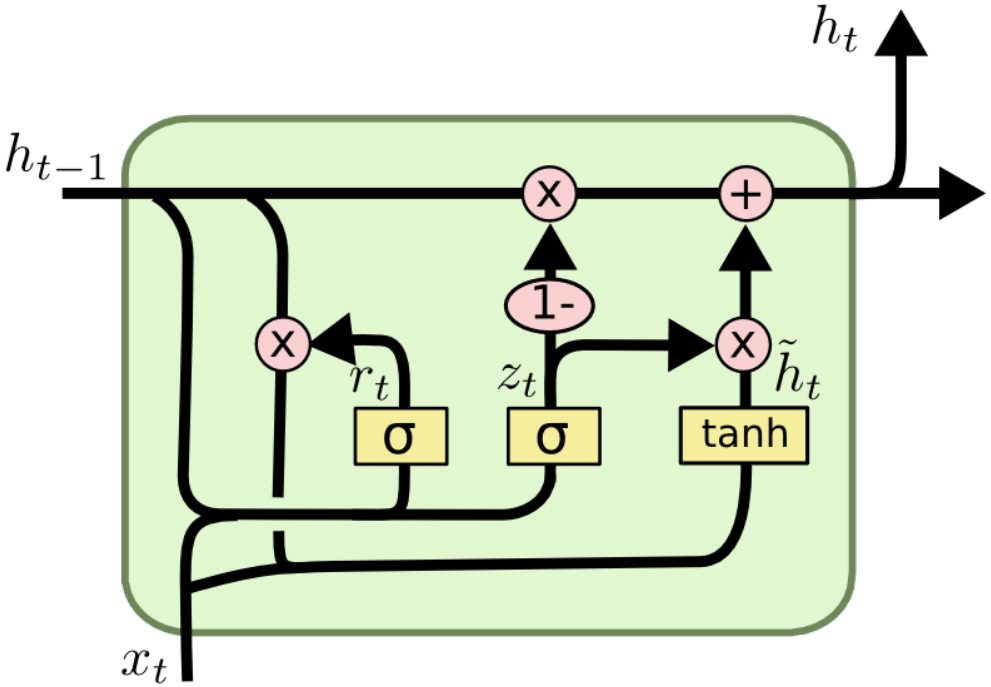
- Coupled forget
  - Instead of separately deciding what to forget and what we should add new information to, we make those decisions together



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

Variants

- More complicated but faster…
  - Forget and input gate are merged in an unique update gate.



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Cho, Kyunghyun, et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." *arXiv preprint arXiv:1406.1078* (2014).

Discussion

*If training traditional neural nets is **optimization over functions**, training recurrent nets is **optimization over programs**.*

- LSTM works a lot better then traditional RNN for most tasks.
- Architecture may change but the main idea is pretty much the same.
- The use of the LSTM is not limited to temporal sequences

LTSM applications

one to one

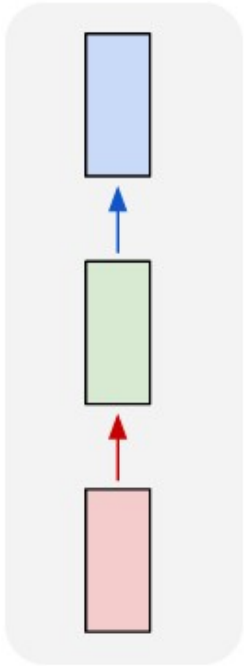


Image classification

one to many

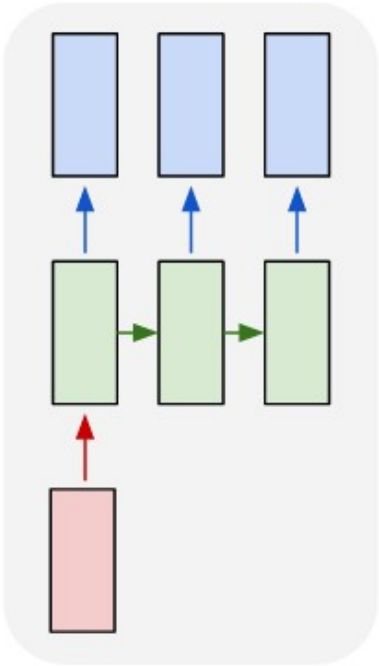
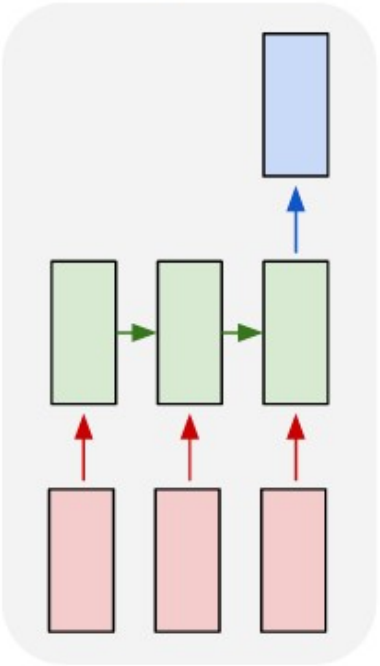


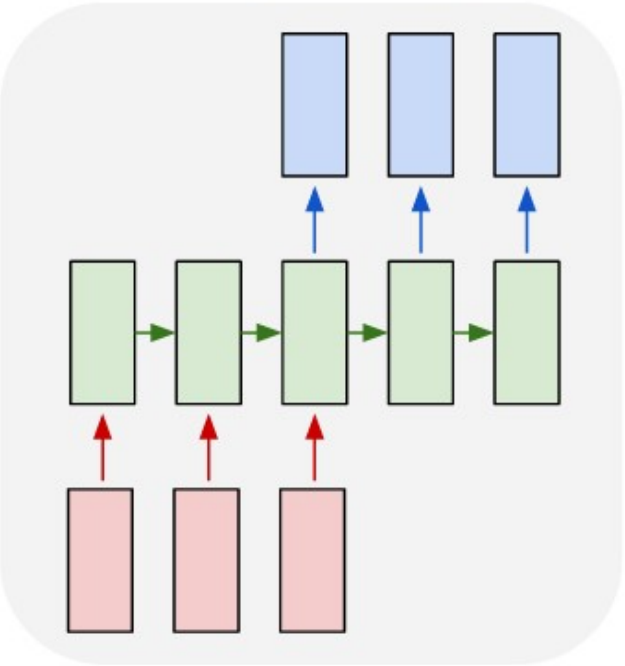
Image captioning

many to one



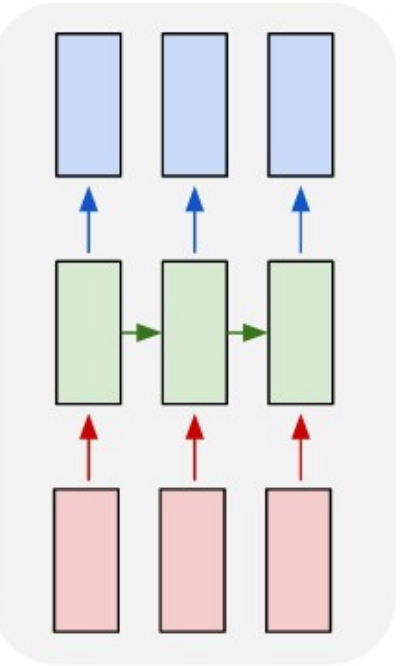
Sentiment Analysis where a given sentence is classified as expressing positive or negative sentiment

many to many



Machine Translation

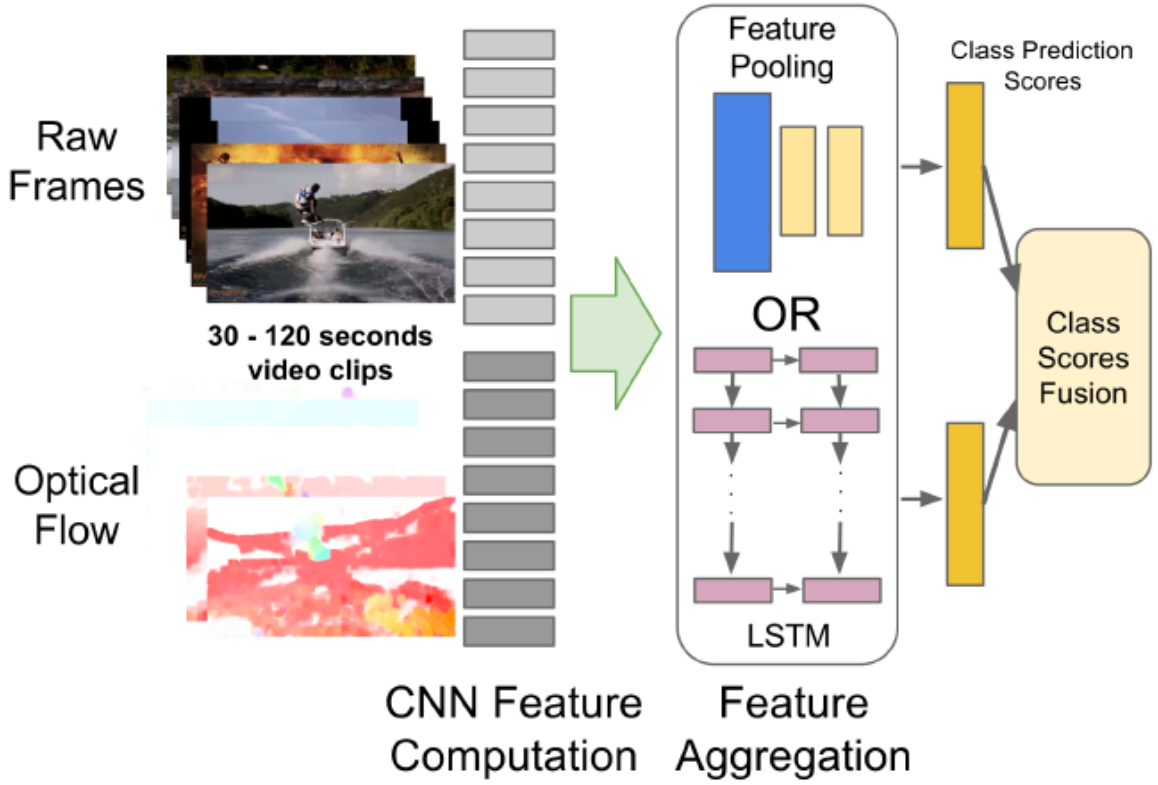
many to many



Video Classification where we wish to label each frame of the video

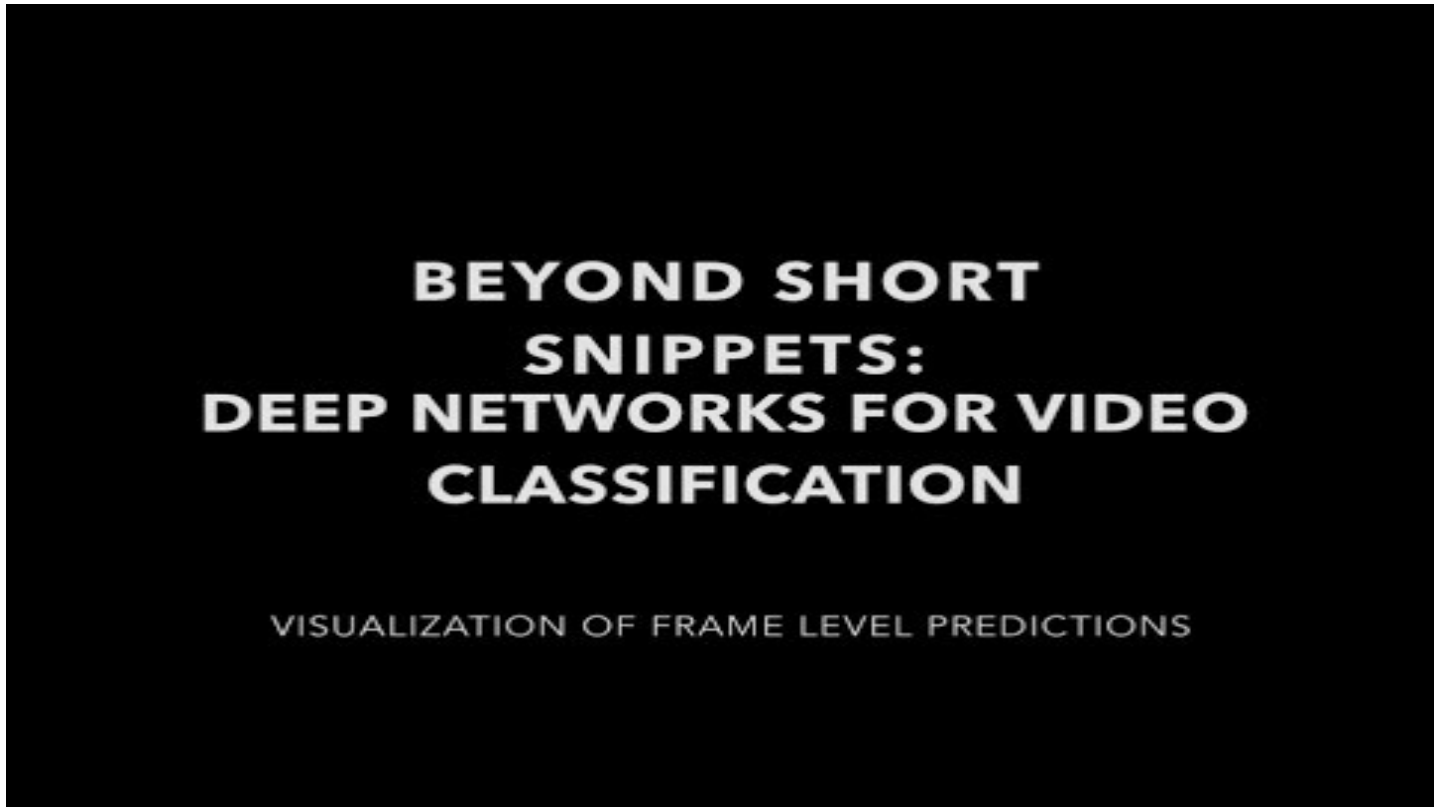
Computer Vision Examples

LSTM for Video Classification



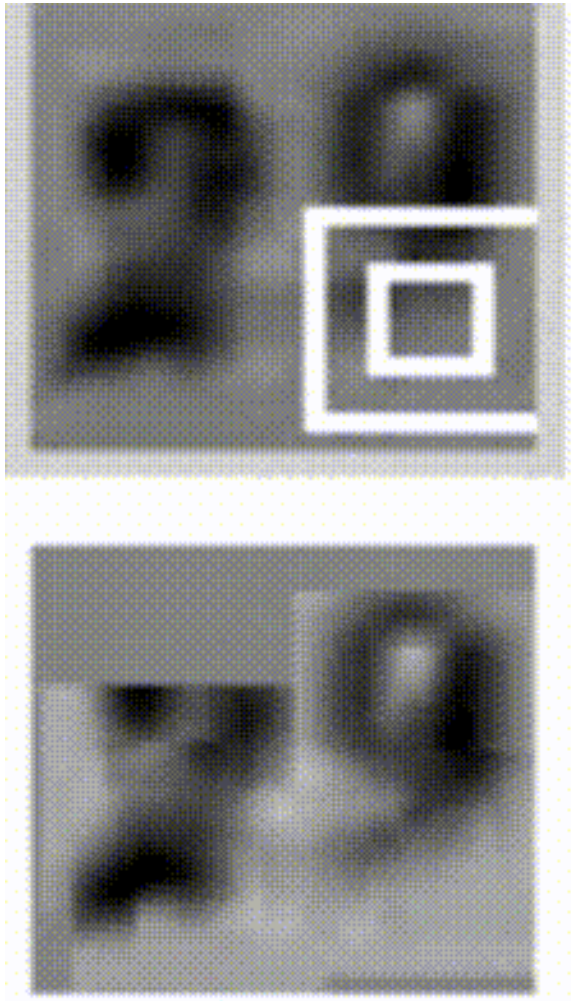
Yue-Hei Ng, Joe, et al. "Beyond short snippets: Deep networks for video classification." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015.

LSTM for Video Classification



Yue-Hei Ng, Joe, et al. "Beyond short snippets: Deep networks for video classification." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015.

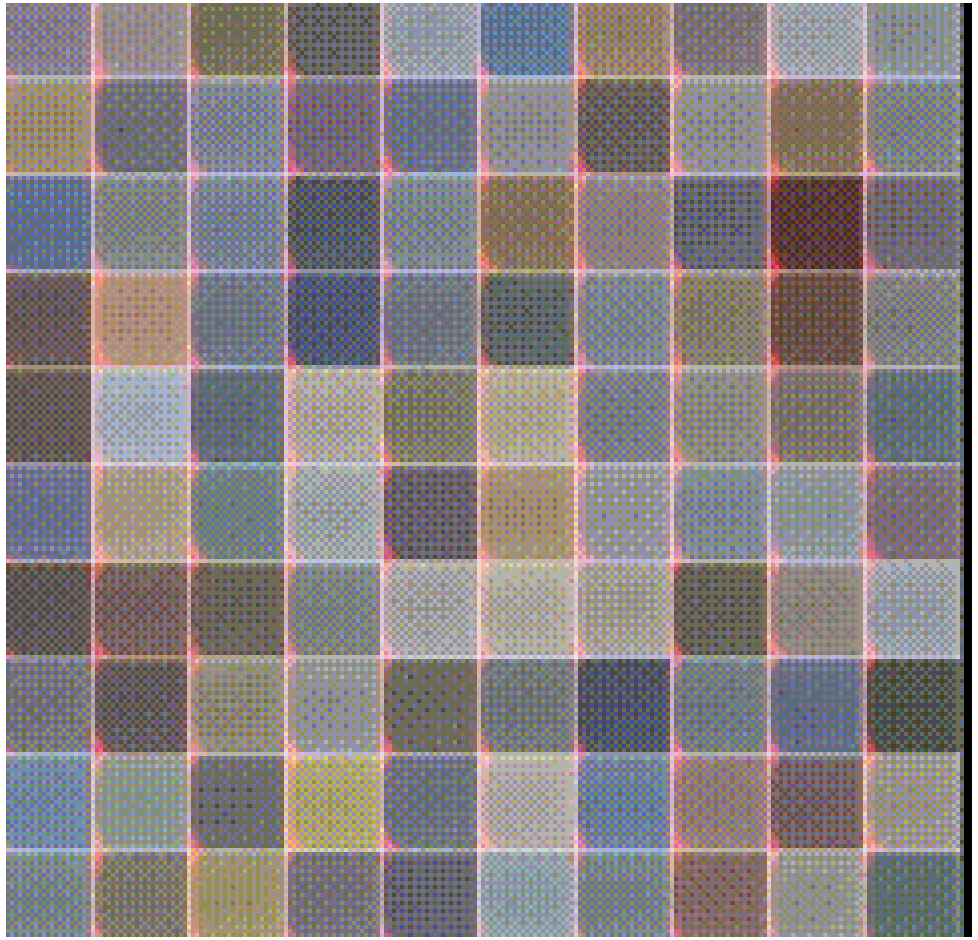
LSTM networks trained to read house numbers from left to right



DeepMind

Ba, Jimmy, Volodymyr Mnih, and Koray Kavukcuoglu. "Multiple object recognition with visual attention." *arXiv preprint arXiv:1412.7755* (2014).

RNN generates images of digits by learning to sequentially add color to canvas



DeepMind

Gregor, Karol, et al. "DRAW: A recurrent neural network for image generation." *arXiv preprint arXiv:1502.04623* (2015).

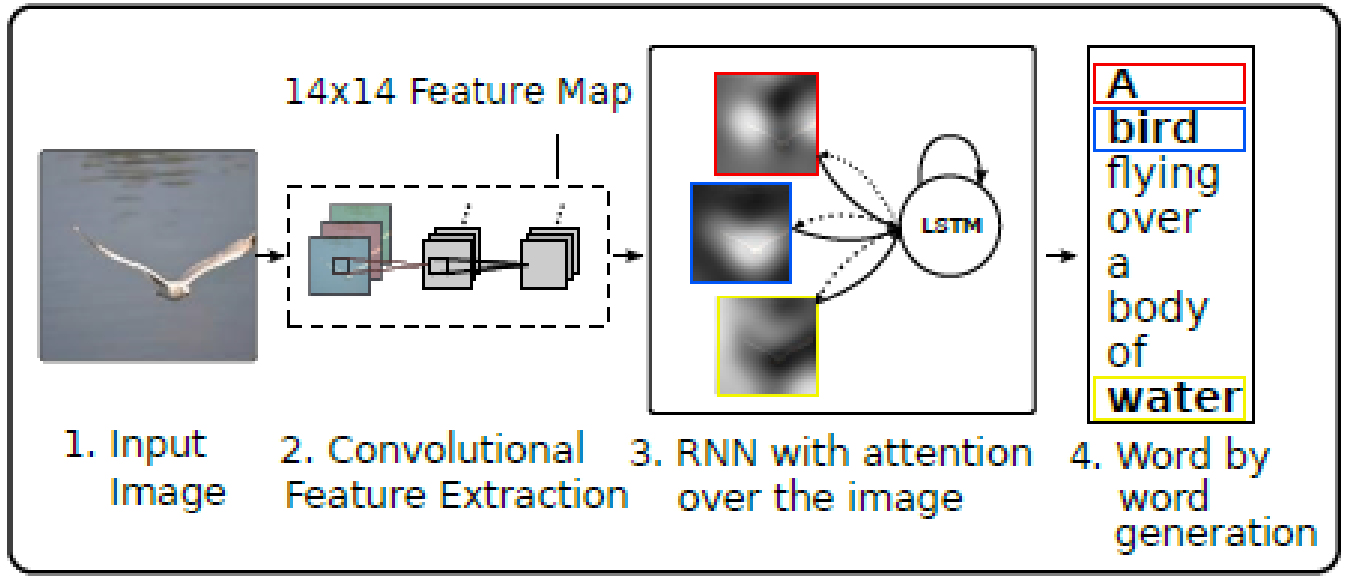
Analysing visual attention

Image Captioning:

- Determining what objects are in the image
- Express their relationship in a natural language

Xu, Kelvin, et al. "Show, attend and tell: Neural image caption generation with visual attention." *arXiv preprint arXiv:1502.03044* 2.3 (2015): 5.

Analysing visual attention



Xu, Kelvin, et al. "Show, attend and tell: Neural image caption generation with visual attention." *arXiv preprint arXiv:1502.03044* 2.3 (2015): 5.

Analysing visual attention - Encoder

- Generated caption:

$$\boldsymbol{y} = \{y_1, \dots, y_C\}, y_i \in \mathbb{R}^K$$

Where *K* is the size of the vocabulary and *C* is the length of a caption.

- Convolutional networks are used to extract local features (Annotation Vectors):

$$\boldsymbol{a} = \{a_1, \dots, a_L\}, a_i \in \mathbb{R}^D$$

Where L is the number of vectors each is a D-dimensional representation corresponding to a part of an image.

Xu, Kelvin, et al. "Show, attend and tell: Neural image caption generation with visual attention." *arXiv preprint arXiv:1502.03044* 2.3 (2015): 5.

Analysing visual attention - Decoder

As a decoder an LSTM is used:

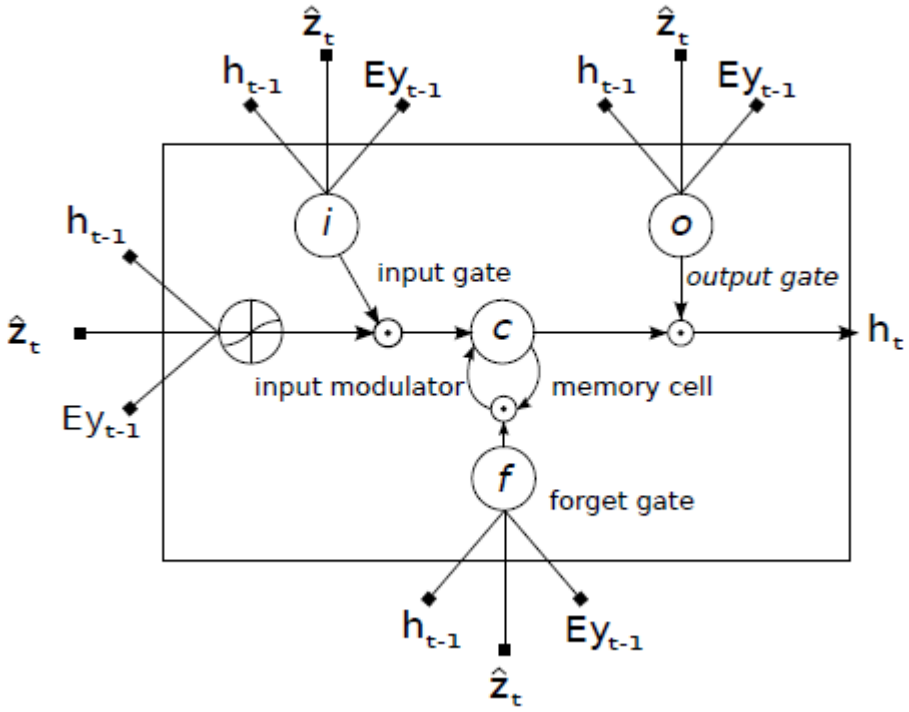
- $\hat{\boldsymbol{z}} \in \mathbb{R}^D$ is the context vector capturing the **visual information** associated with a **particular input location** (basically a dynamic representation of relevant part of the image at time t)
- A mechanism ϕ computes $\hat{z}_t$ from the annotation vector $a_i, i = 1, \ldots, L$ corresponding to the features extracted from different image locations.

$$\hat{\boldsymbol{z}}_t = \phi(\{a_i\}, \{\alpha_i\})$$

- $\forall i$ the mechanism ϕ generates positive weight α_i which can be interpreted either as the probability that location i is the right place to focus to produce the next word or as the relative importance to give to location i in blending the a_i's together.

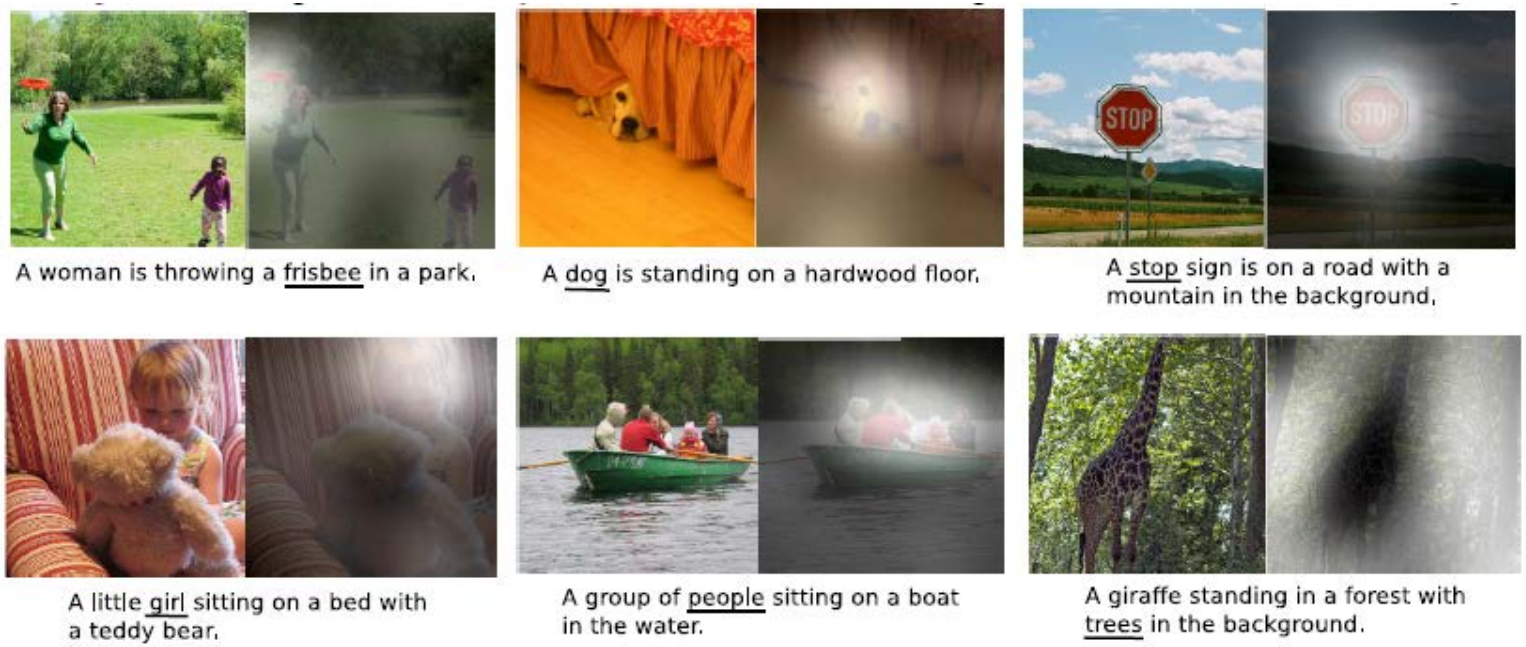
$$\alpha_{ti} = \frac{e^{g_{ti}}}{\sum_{k=1}^{L} e^{g_{tk}}}$$

$$g_{ti} = f_{att}(\boldsymbol{a}_i, \boldsymbol{h}_{t-1})$$



Xu, Kelvin, et al. "Show, attend and tell: Neural image caption generation with visual attention." *arXiv preprint arXiv:1502.03044* 2.3 (2015): 5.

Analysing visual attention



Xu, Kelvin, et al. "Show, attend and tell: Neural image caption generation with visual attention." *arXiv preprint arXiv:1502.03044* 2.3 (2015): 5.

Analysing visual attention - Decoder

Xu, Kelvin, et al. "Show, attend and tell: Neural image caption generation with visual attention." *arXiv preprint arXiv:1502.03044* 2.3 (2015): 5.

More Sources

- Y. Bengio et al. - Sequence Modeling: Recurrentand Recursive Nets - http://www.deeplearningbook.org/contents/rnn.html
- Colah's blog – Understanding LSTM Networks - http://colah.github.io/posts/2015-08-Understanding-LSTMs/
- Andrej Karpathy - The Unreasonable Effectiveness of Recurrent Neural Networks - http://karpathy.github.io/2015/05/21/rnn-effectiveness/
- Brandon Rohrer - How Convolutional Neural Networks work - http://brohrer.github.io/how_convolutional_neural_networks_work.html