Big Data Analytics

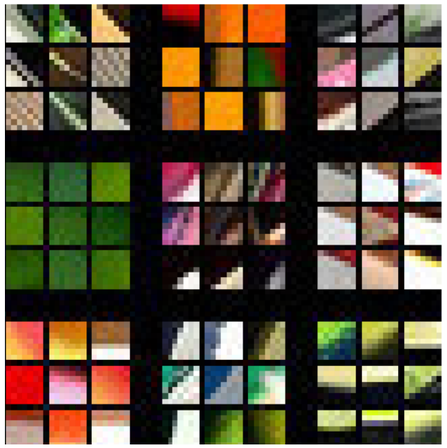
Prof. Dr.-Ing. Youness Dehbi
Computational Methods Lab
HafenCity University Hamburg

Slides are based on the lecture of Prof. Ribana Roscher from the University of Bonn

Convolutional & Recurrent Neural Networks

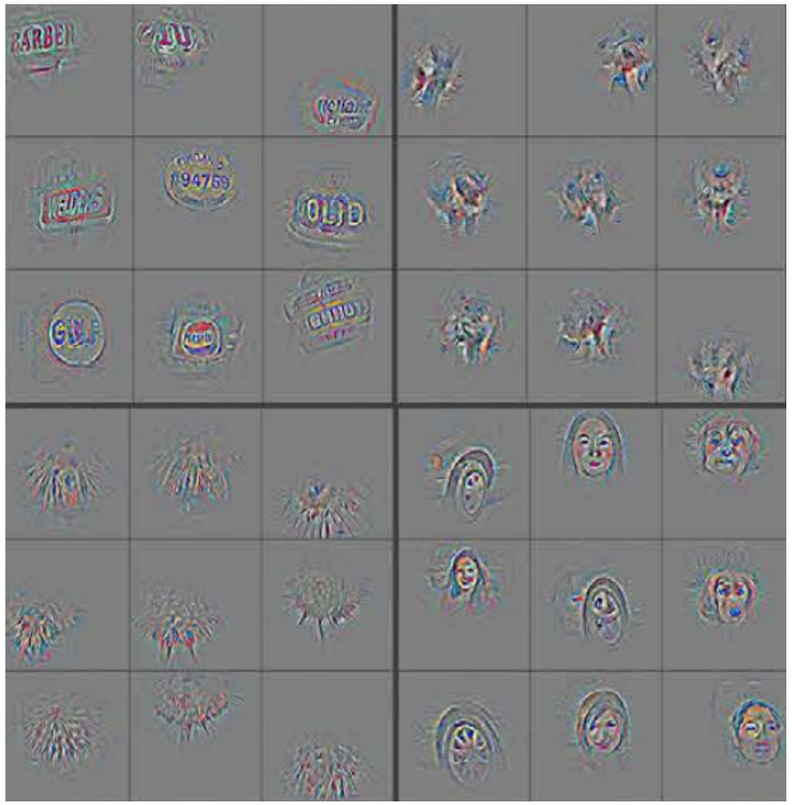
Convolutional neural networks

- ConvNets or CNNs



first layer

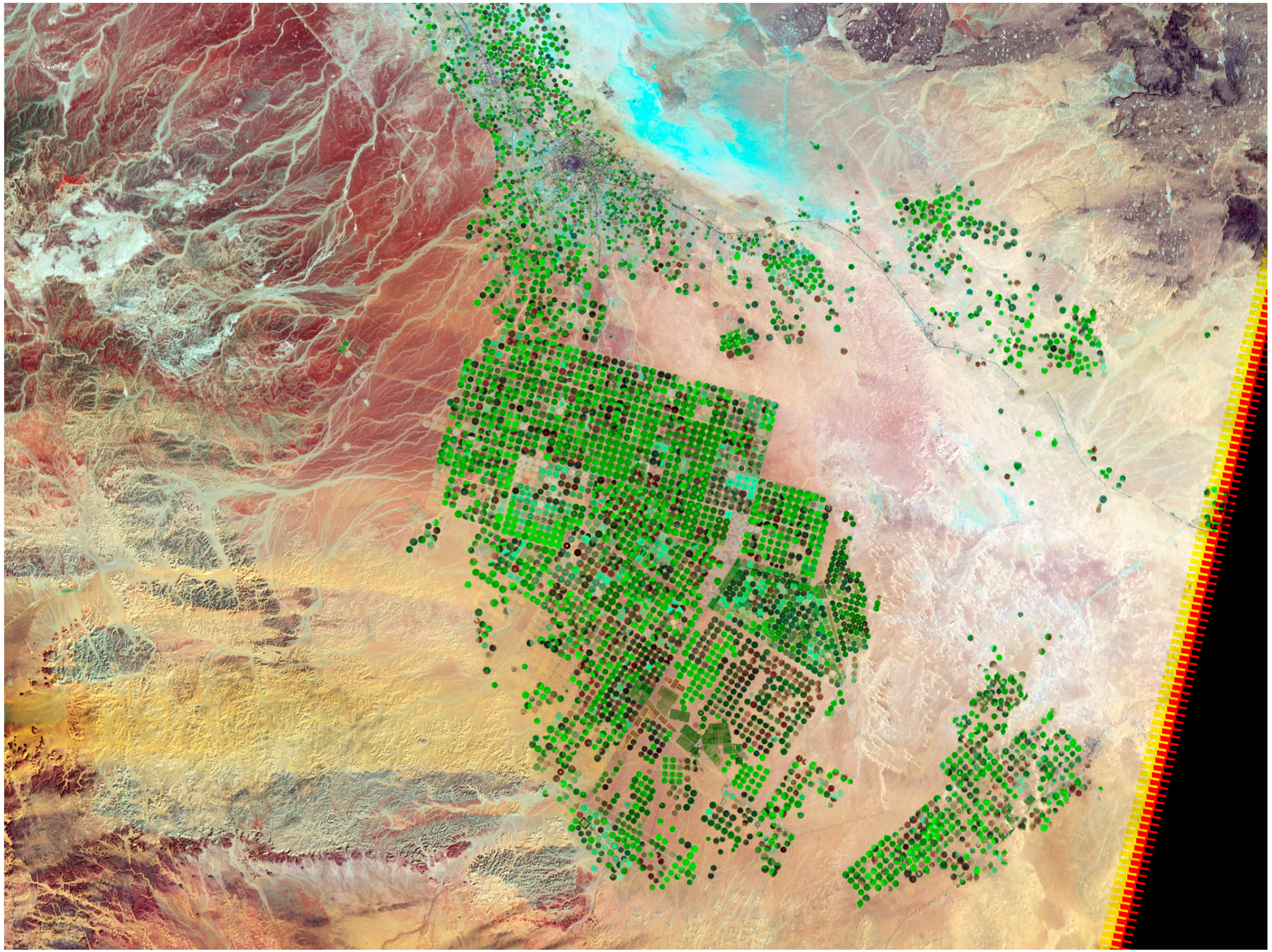
intermediate layer



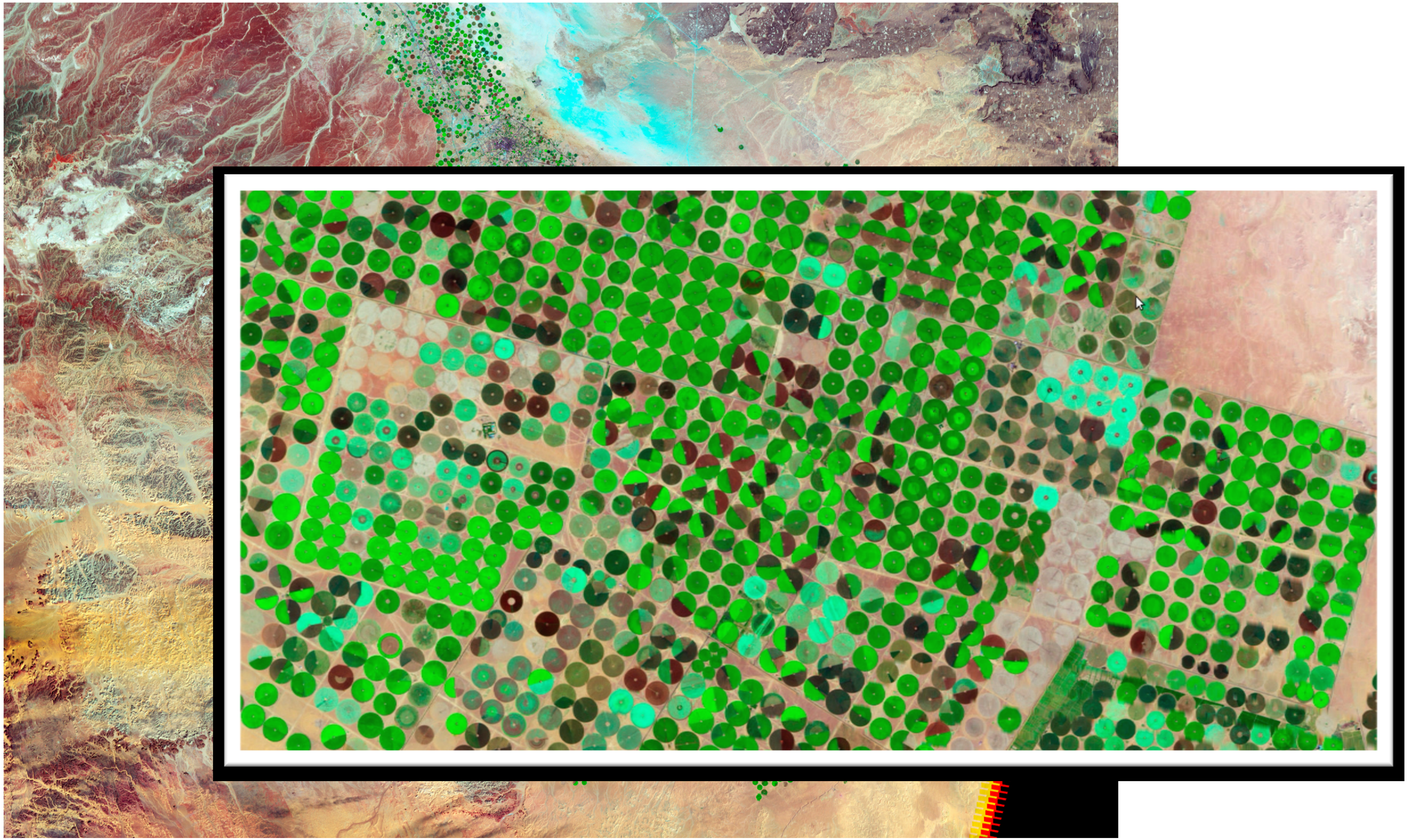
deep layer

Zeiler, M. D., & Fergus, R. (2014, September). Visualizing and understanding convolutional networks. In European conference on computer vision (pp. 818-833). Springer.

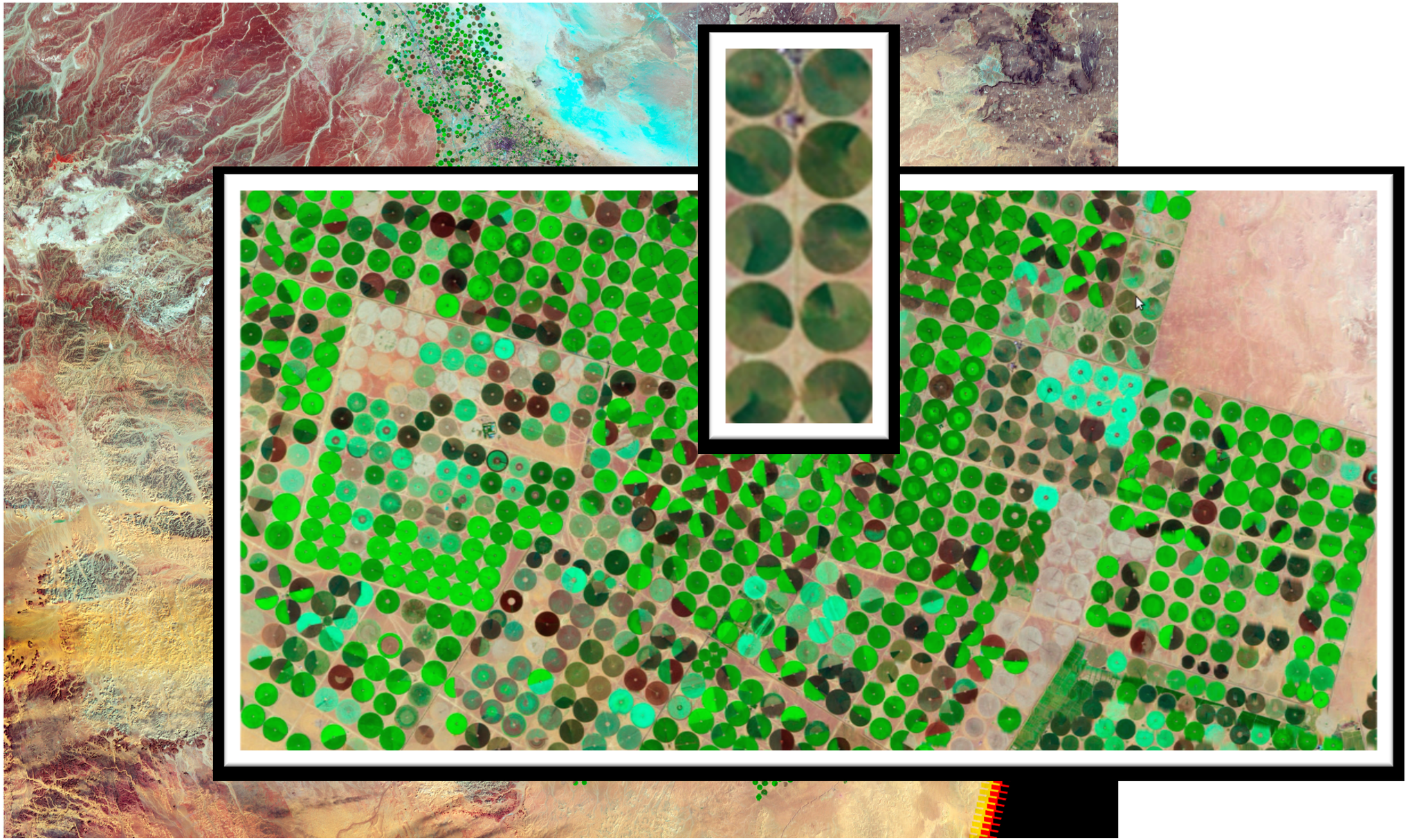
Basic principle



Basic principle



Basic principle



Basic principle

CNN

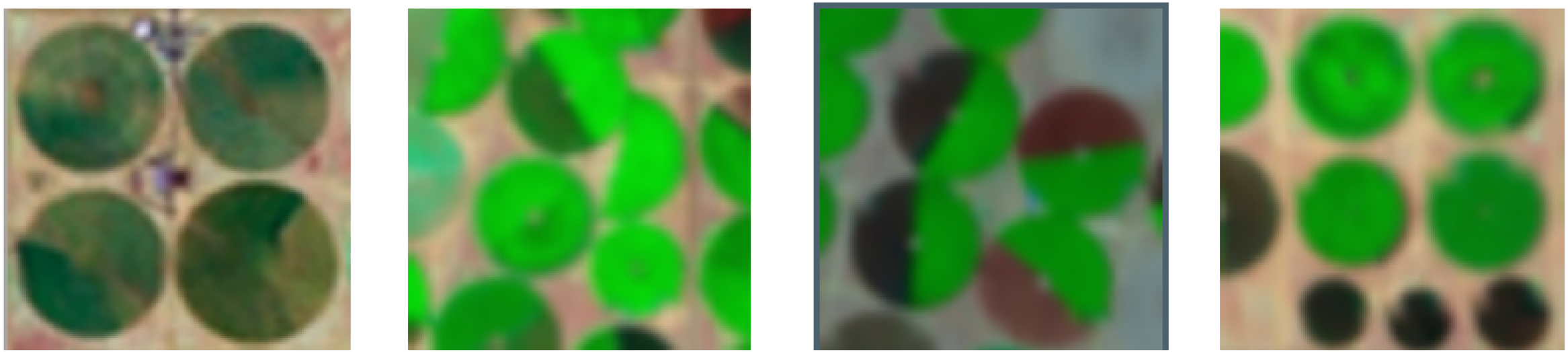
class 1
(agricultural area)

Or

class 2
(desert)

Basic principle

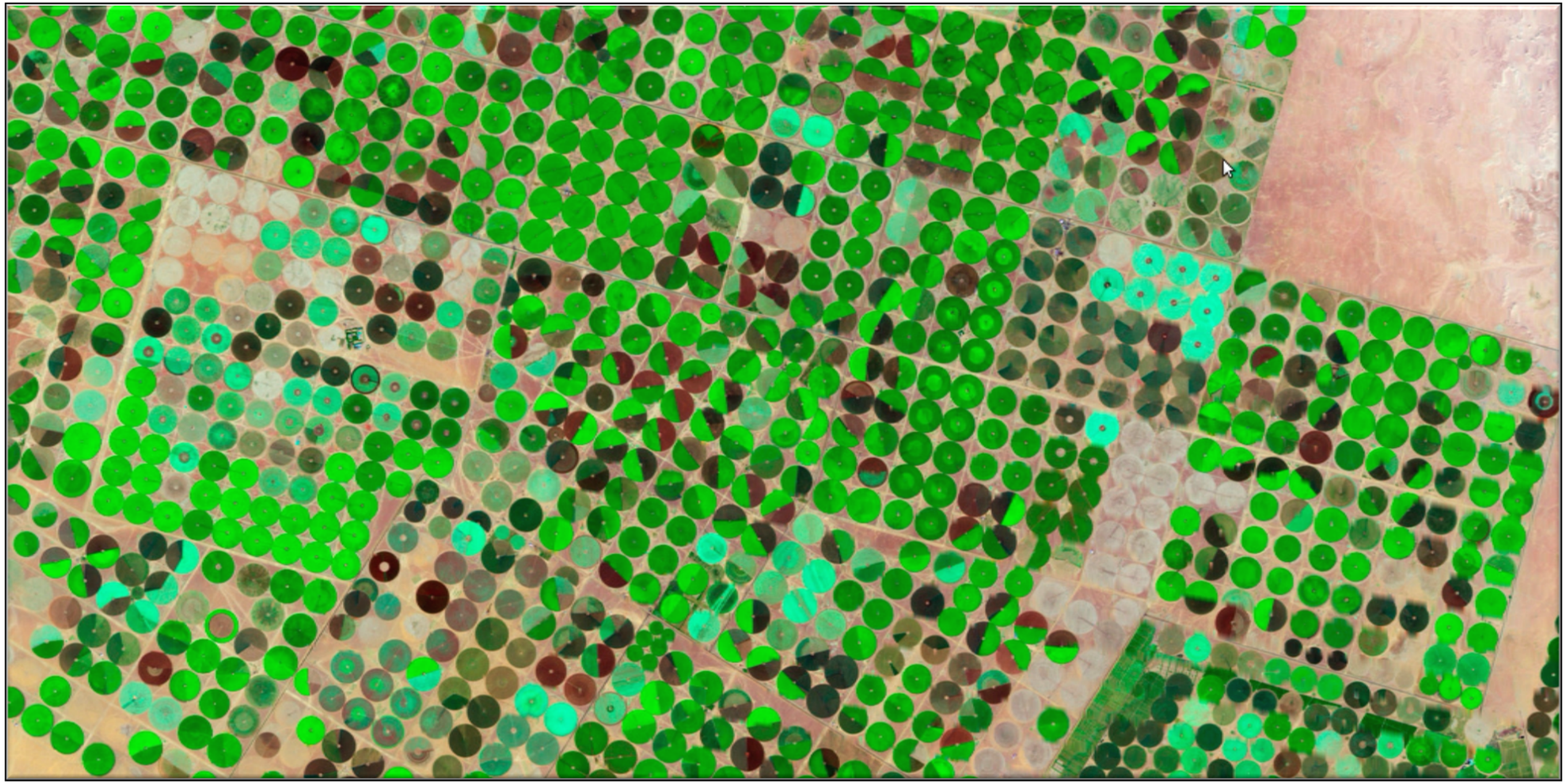
- Objects can be translated, scaled, rotated, weighted, ...



Contain similar **structures**

Basic principle

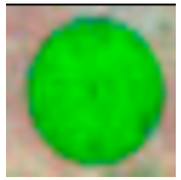
Find similar parts, which characterize the image



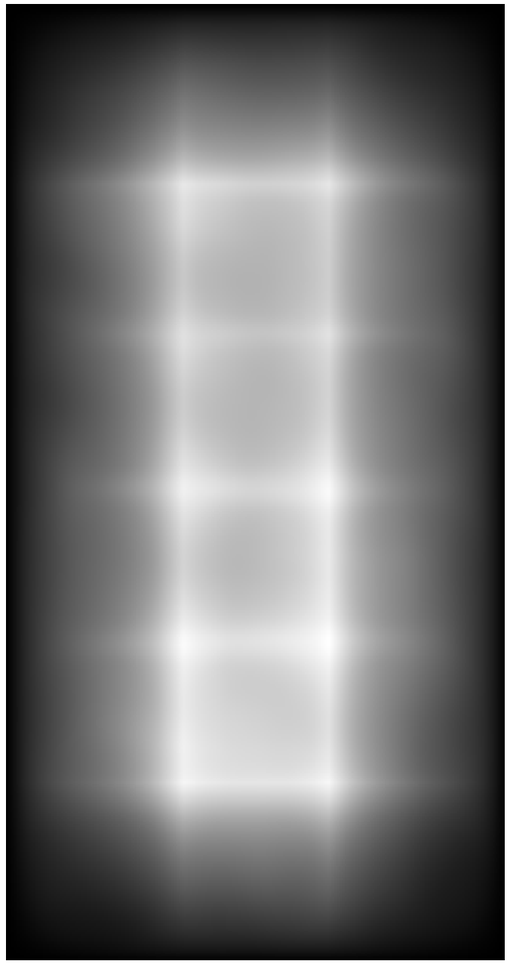
- Math behind: filtering/**convolution**

Convolutional layer

filter



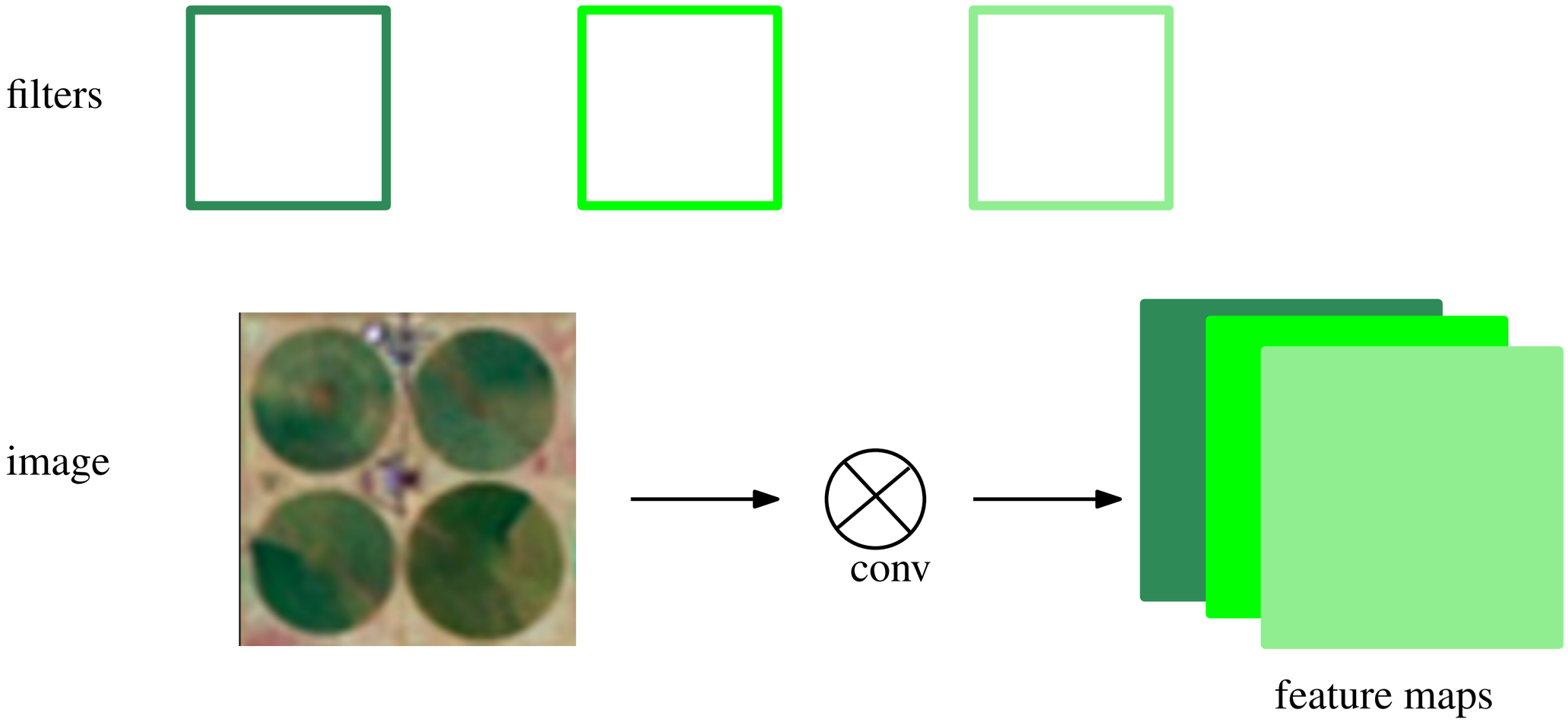
image



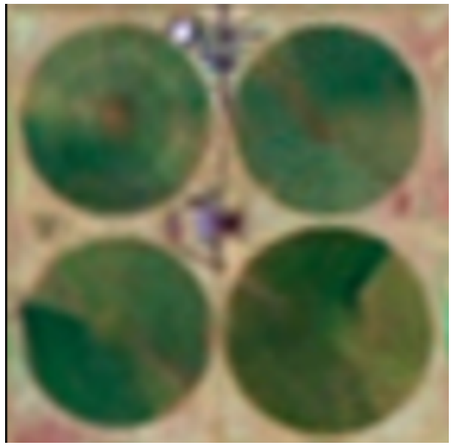
feature map

Convolutional layer

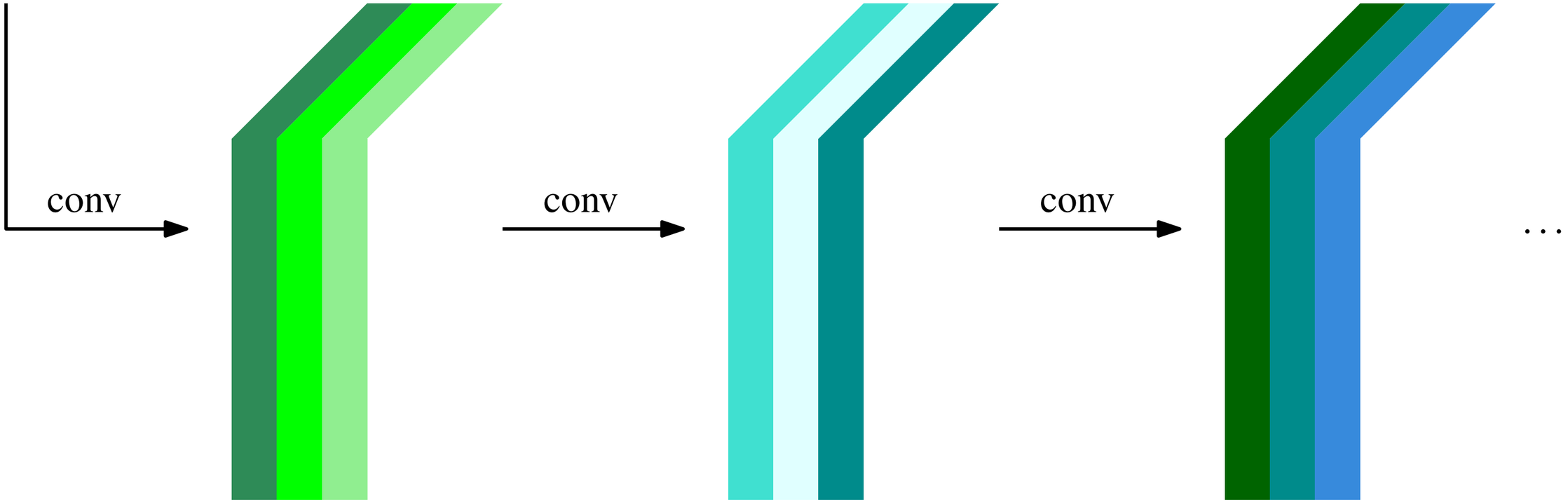
- One image becomes a stack of filtered images



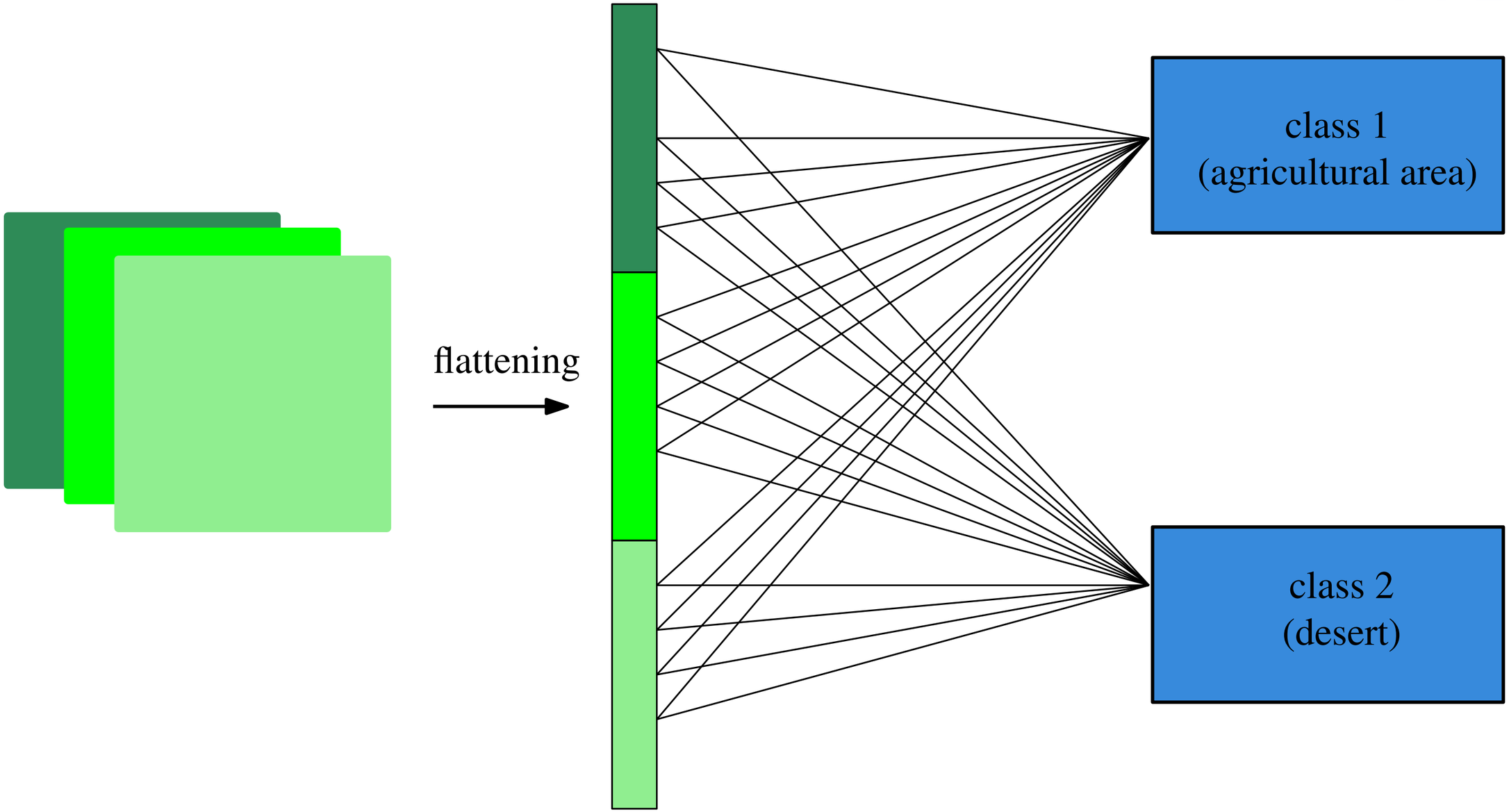
Convolutional neural network



CNNs contain a number of convolutional layers



Fully connected layer

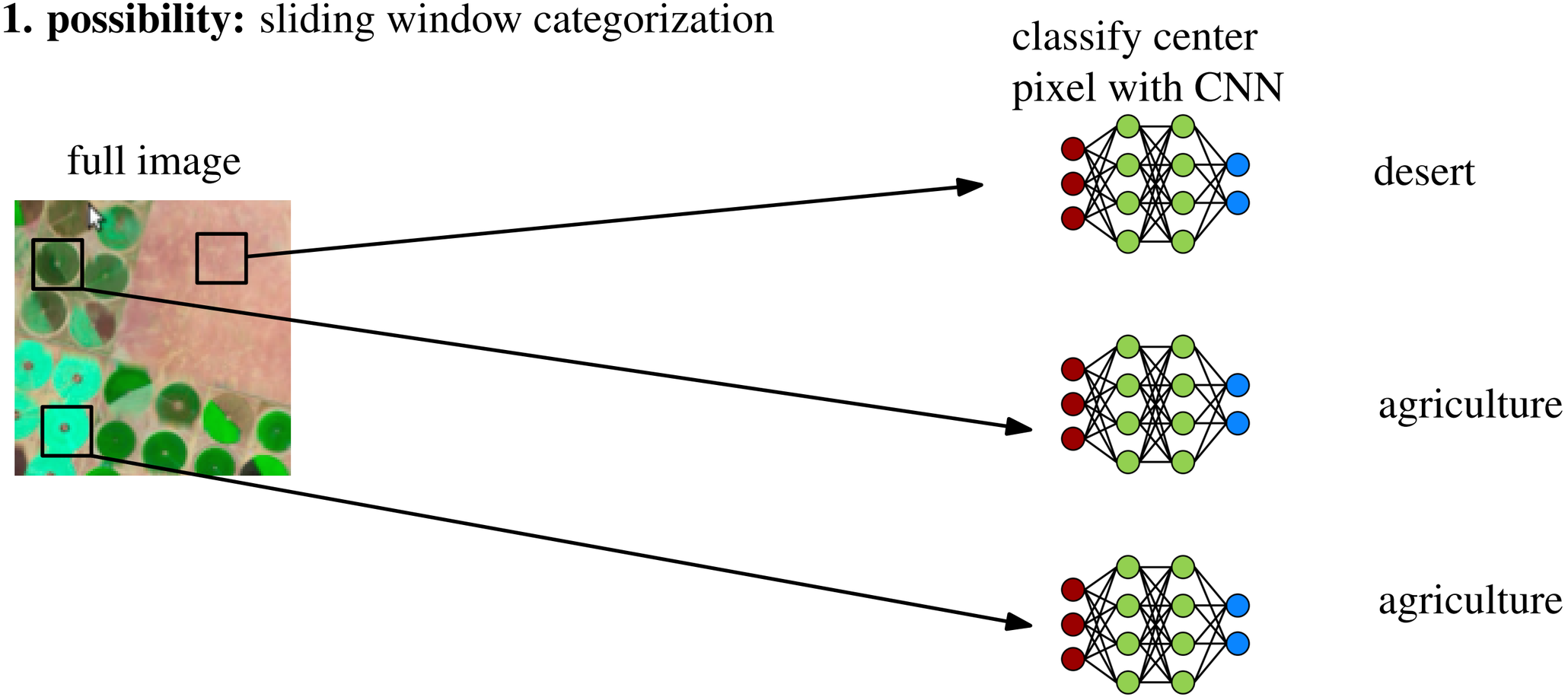


- Each feature value becomes a list of votes

Image segmentation

- How can you pixelwise segment the image (instead of categorizing it)?

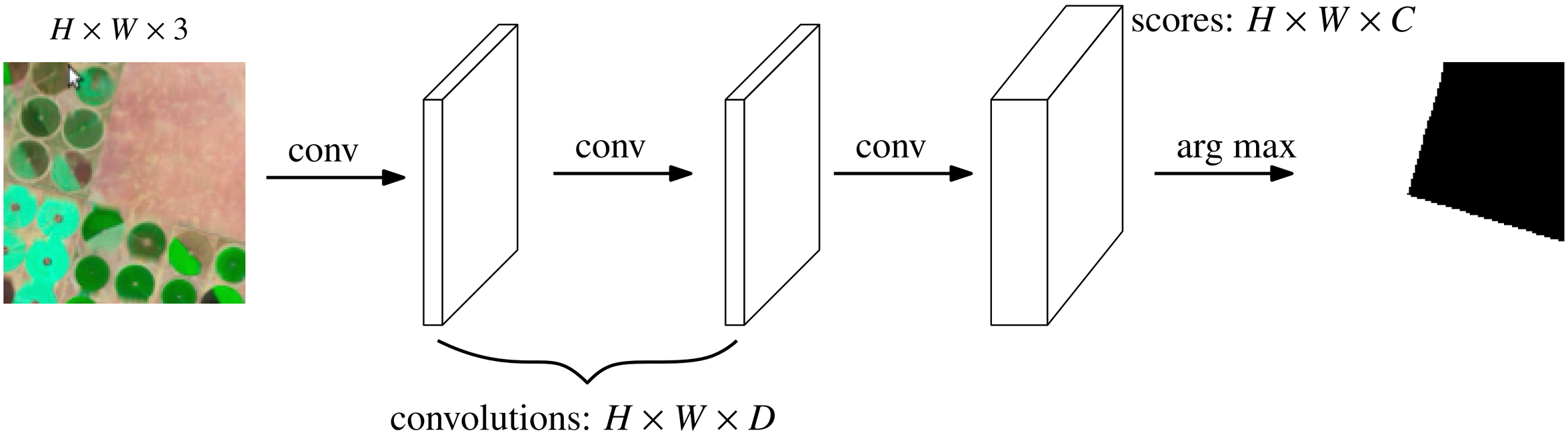
1. possibility: sliding window categorization



- too inefficient

Image segmentation

2. possibility: bunch of convolutional layers to make predictions for pixels all at once

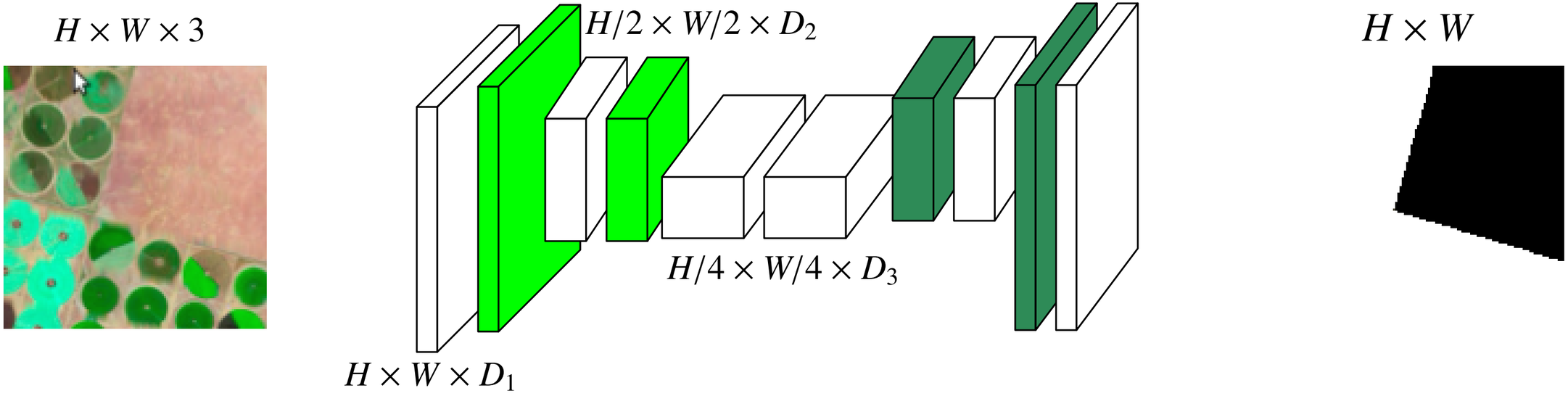


- Each convolutional layer is as big as the input

$\Rightarrow$ Computationally expensive

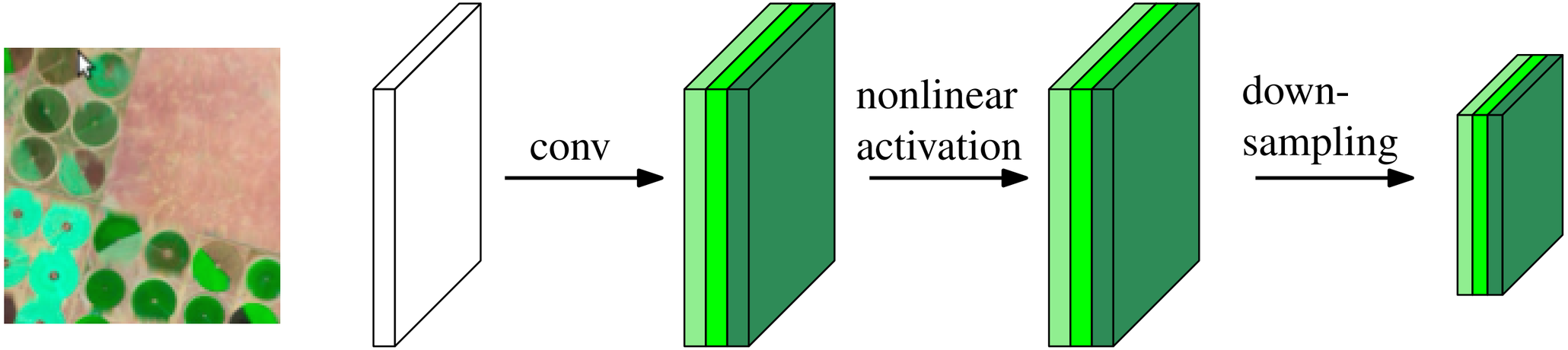
Image segmentation

3. possibility: Downsampling and upsampling



Blocks of layers

- CNNs oftentimes consist of a sequence of alternating convolutional layers and downsampling layers (a sequence of blocks)



Why downsampling?

- Reduces the complexity in the network
- Generates translation-invariance and rotation-invariance to some extent
- If the convolutional filter size stays the same, the downsampling layers cause an **increase in the receptive field** in deeper layers

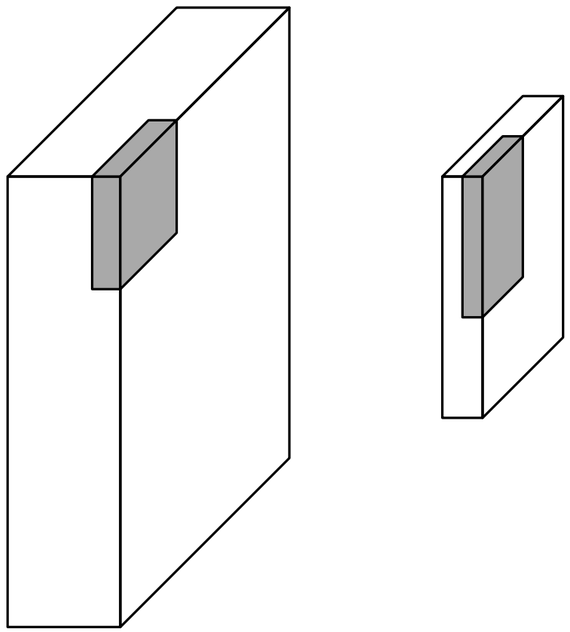
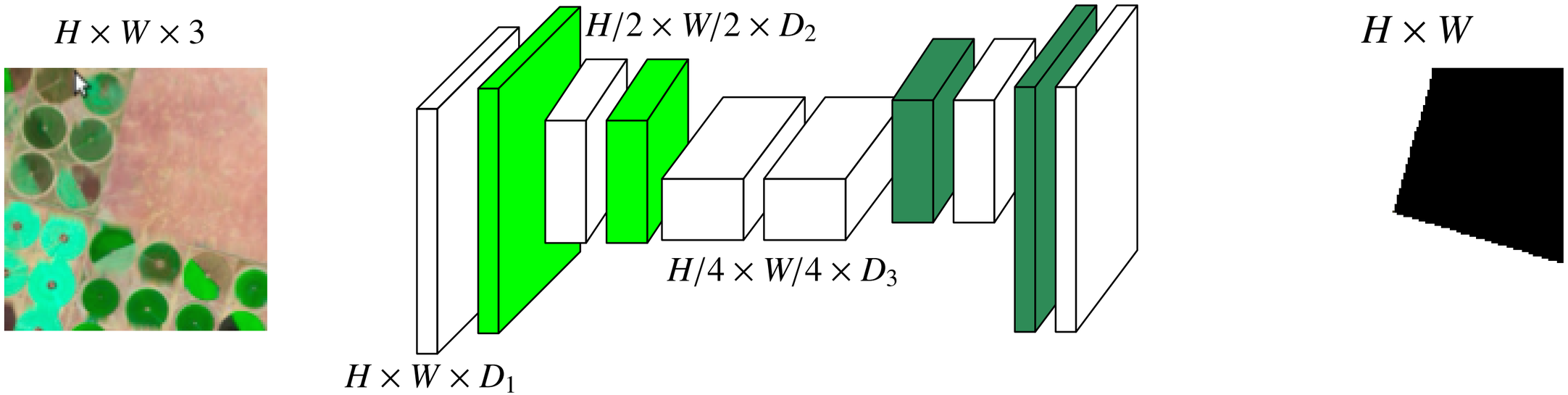


Image segmentation

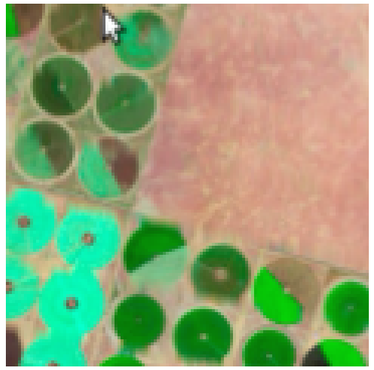


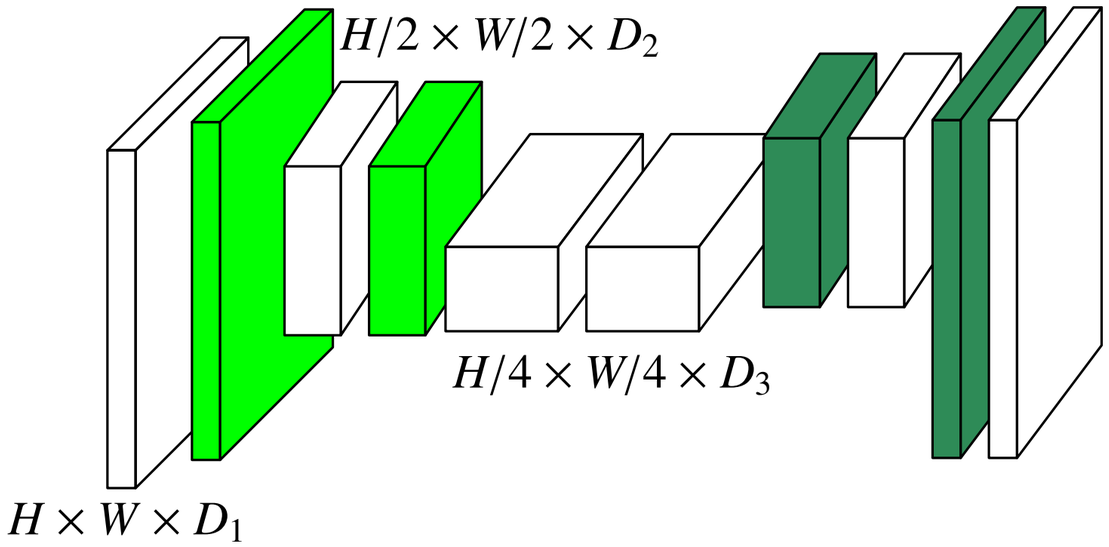
- Generally, the number of filters increase with depth, because:
  - lower layers with too many filters are too computationally intense
  - the correlation between filters decreases with depth

Noh, H., Hong, S., & Han, B. (2015). Learning deconvolution network for semantic segmentation. In Proceedings of the IEEE International Conference on Computer Vision (pp. 1520-1528).

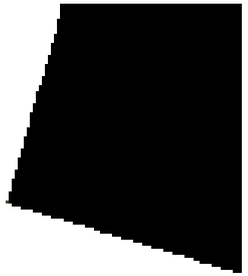
Image segmentation network

$H \times W \times 3$





$H \times W$



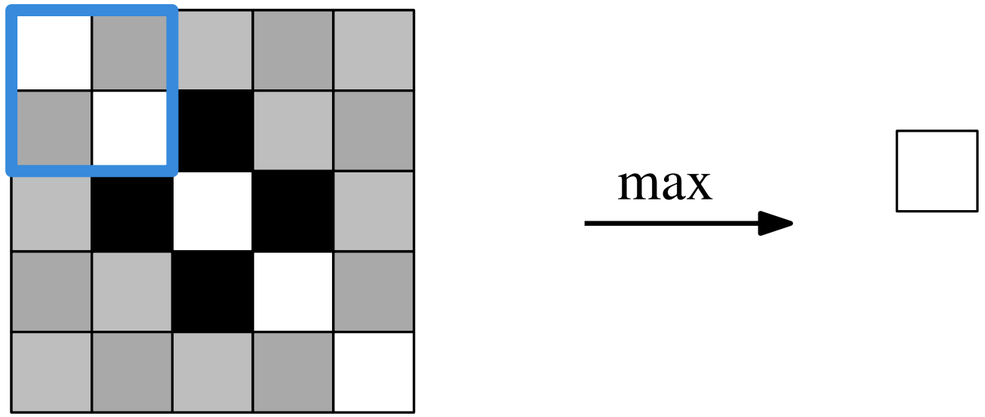
downsampling
- pooling
- convolution

upsampling
- unpooling
- deconvolution

- Mostly symmetric structure

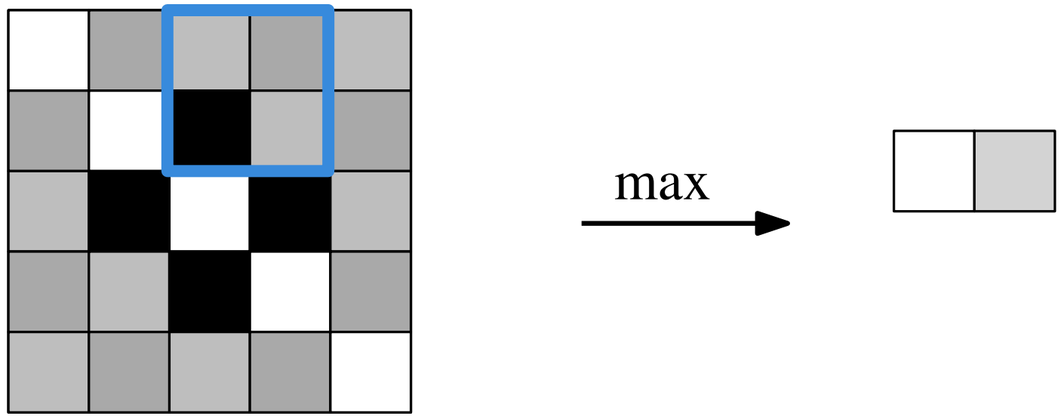
Downsampling: pooling

- Result: shrinked image stack
- Commonly used: **max-pooling** and mean-pooling
- Pooling layer defined by stride (2) and window size (2×2)



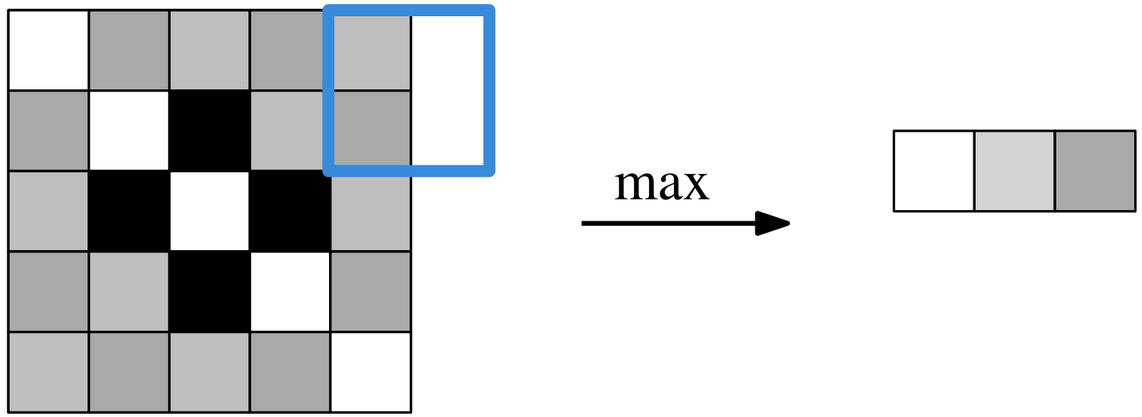
Downsampling: pooling

- Result: shrinked image stack
- Commonly used: **max-pooling** and mean-pooling
- Pooling layer defined by stride (2) and window size (2×2)



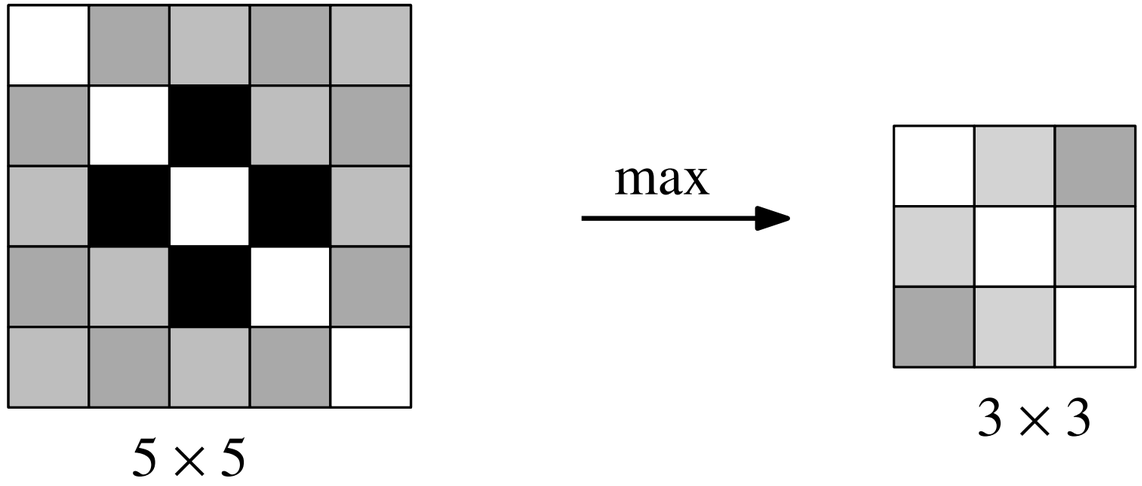
Downsampling: pooling

- Result: shrinked image stack
- Commonly used: **max-pooling** and mean-pooling
- Pooling layer defined by stride (2) and window size (2×2)



Downsampling: pooling

- Result: shrinked image stack
- Commonly used: **max-pooling** and mean-pooling
- Pooling layer defined by stride (2) and window size (2×2)



Upsampling: unpooling

Nearest neighbor

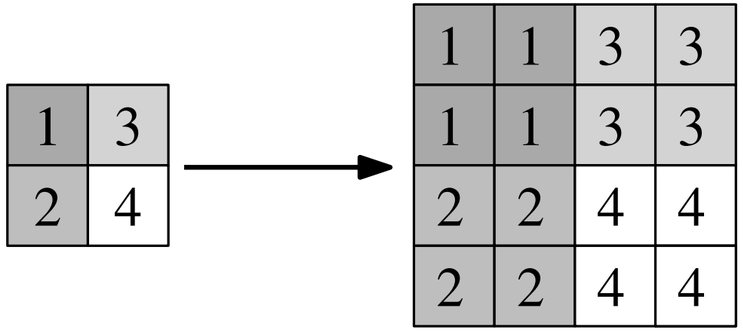
1	3
2	4

→

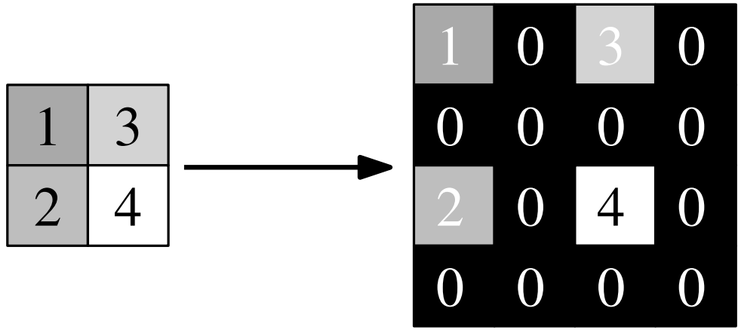
1	1	3	3
1	1	3	3
2	2	4	4
2	2	4	4

Upsampling: unpooling

Nearest neighbor



Bed of Nails



Upsampling: unpooling

Nearest neighbor

1	3
2	4

→

1	1	3	3
1	1	3	3
2	2	4	4
2	2	4	4

Bed of Nails

1	3
2	4

→

1	0	3	0
0	0	0	0
2	0	4	0
0	0	0	0

Max unpooling

1	3
2	4

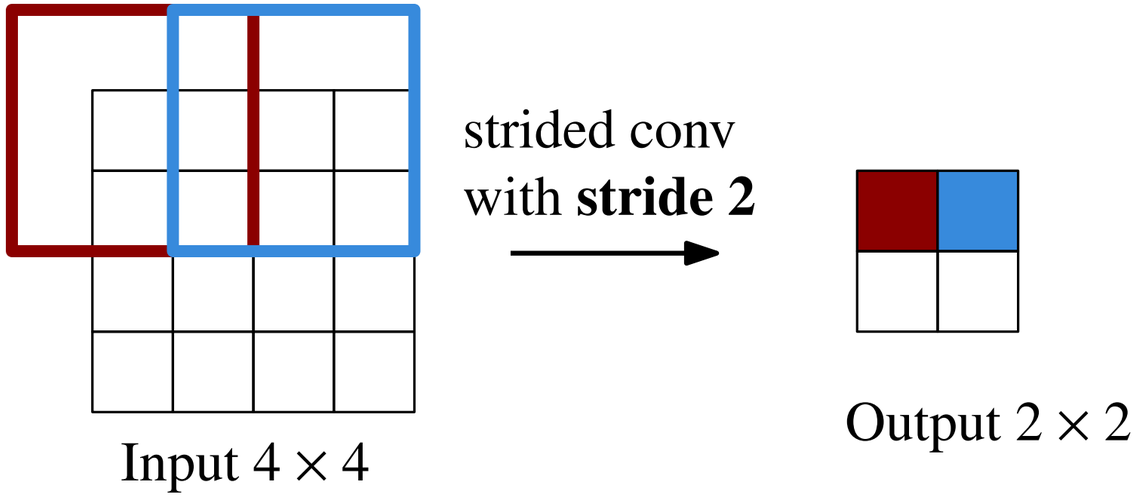
→

0	0	3	0
0	1	0	0
0	0	0	0
2	0	0	4

- Positions of max values in downsampling layers are stored

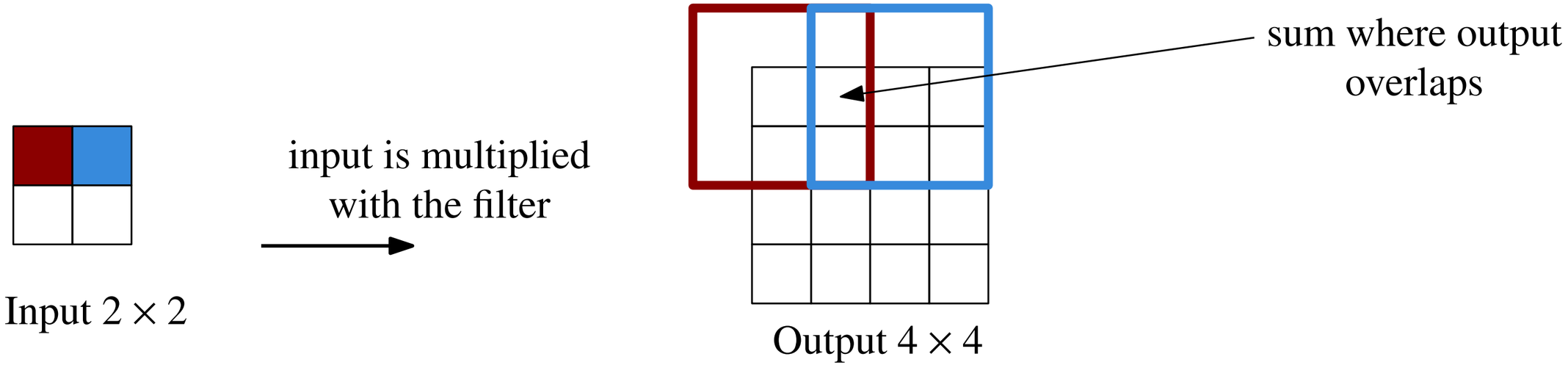
Downsampling: convolution

- Normally, stride 1 is used for convolution, but a stride > 1 causes downsampling



Upsampling: deconvolution

- Aka upconvolution, transpose convolution, backward strided convolution, ..., but it is not an inverted convolution!
- 3×3 deconvolution, stride 2, pad 1



Learning the parameters

- What needs to be learned:
  - Filters
  - Weights in fully connected/dense layers
  - But: Pooling layers have no parameters

- How?
  - **Backpropagation**

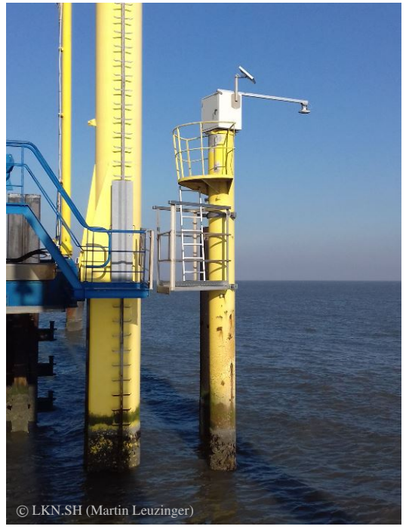
Hyperparameters which need to be set:

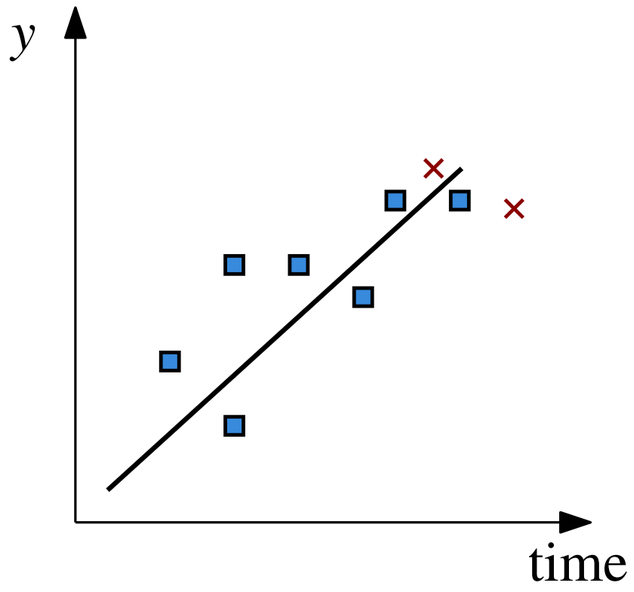
- Convolutions
  - Number of filters
  - Size of filters
  - (Window stride)
- Pooling
  - Window size
  - Window stride

Recurrent Neural Networks & LSTM

Time-series data

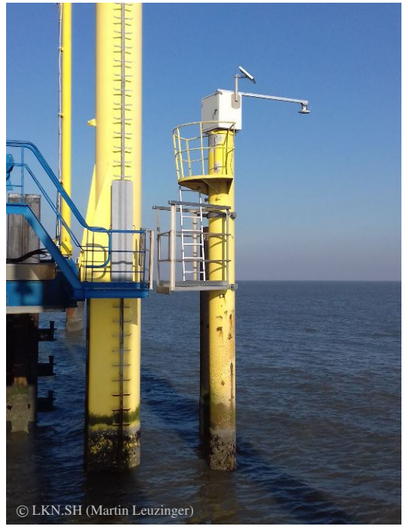
- Measurement of sea level at an oceanographic tide gauge station

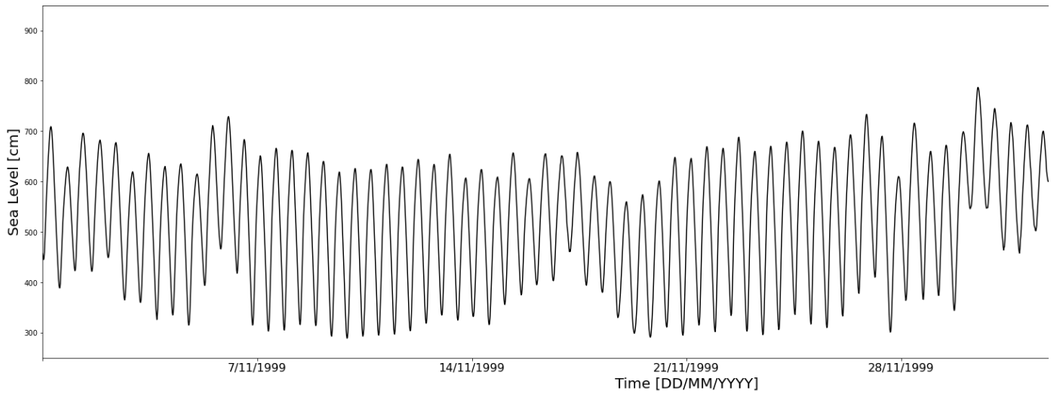


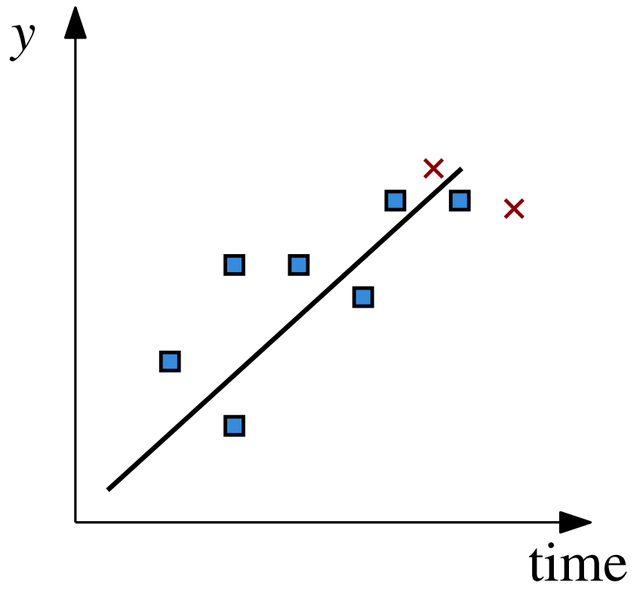


Time-series data

- Measurement of sea level at an oceanographic tide gauge station

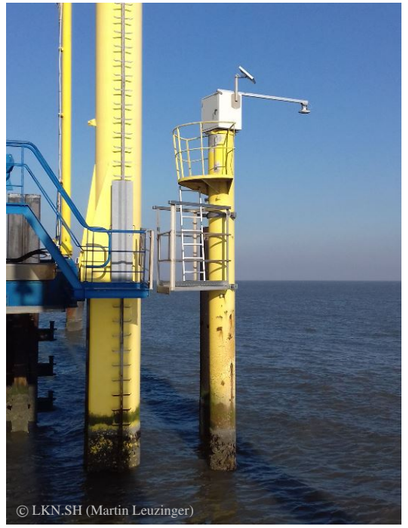




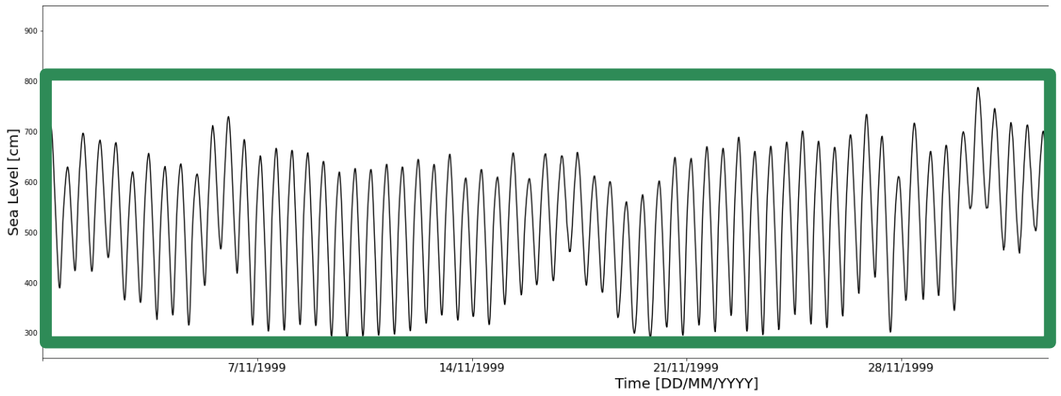


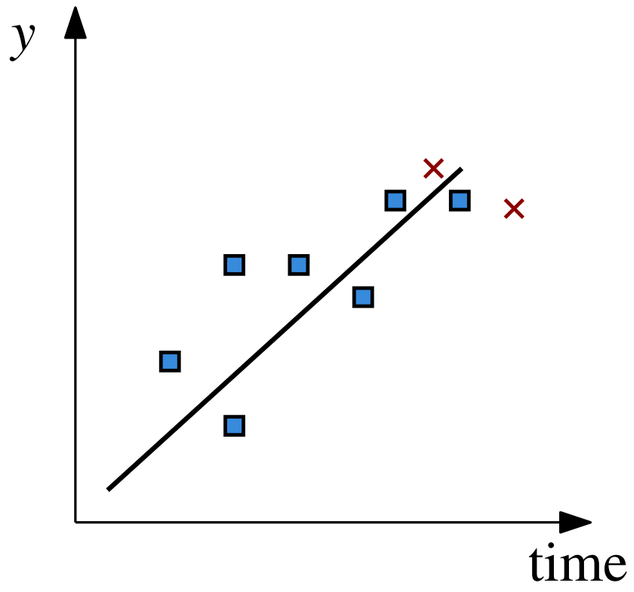
Time-series data

- Measurement of sea level at an oceanographic tide gauge station



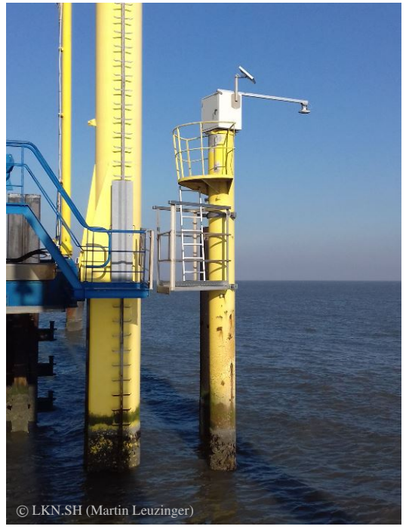
$$I = x_{n-t} \rightarrow x_{n-t+1} \rightarrow \ldots \rightarrow x_n$$



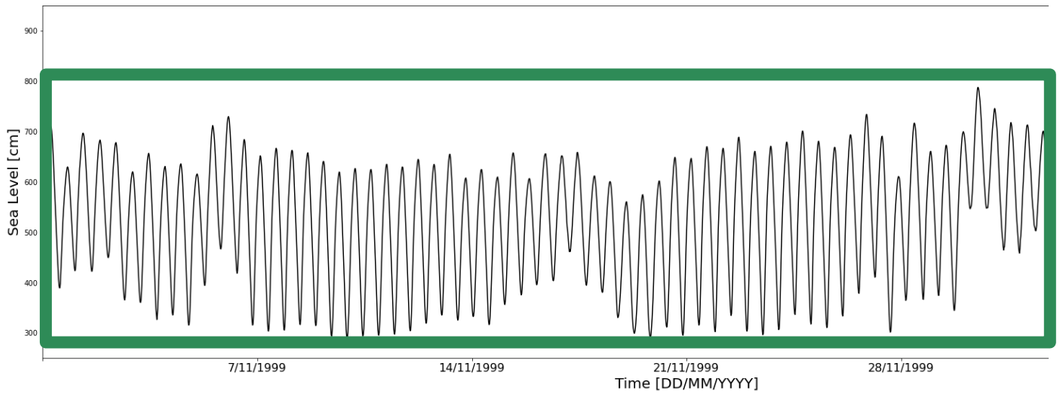


Time-series data

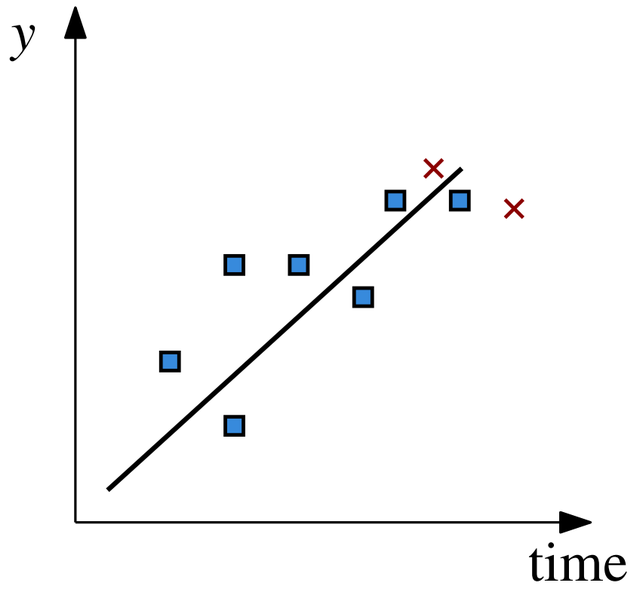
- Measurement of sea level at an oceanographic tide gauge station



$$I = x_{n-t} \rightarrow x_{n-t+1} \rightarrow \ldots \rightarrow x_n \qquad \Rightarrow \qquad o = x_{n+1}$$

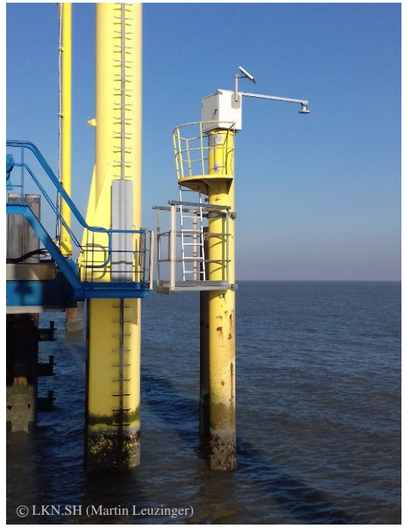


?



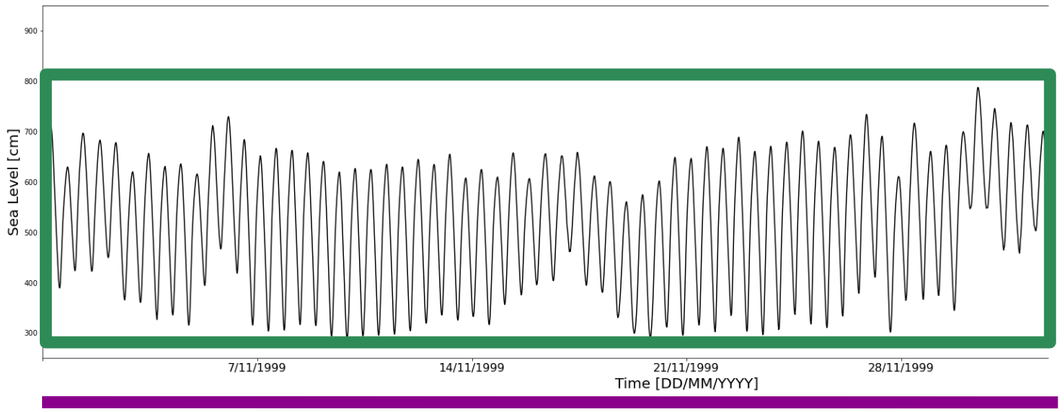
Time-series data

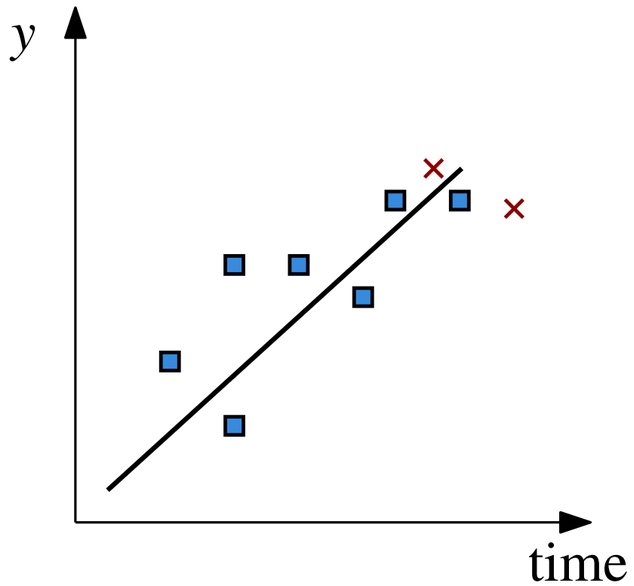
- Measurement of sea level at an oceanographic tide gauge station



$$I = x_{n-t} \rightarrow x_{n-t+1} \rightarrow \ldots \rightarrow x_n \qquad \Rightarrow \qquad o = x_{n+1}$$

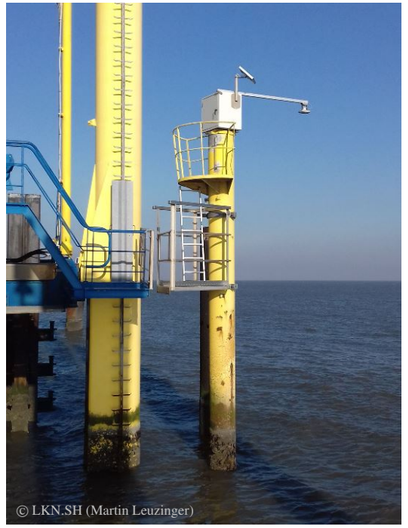
- Modeling of sequences of variable length





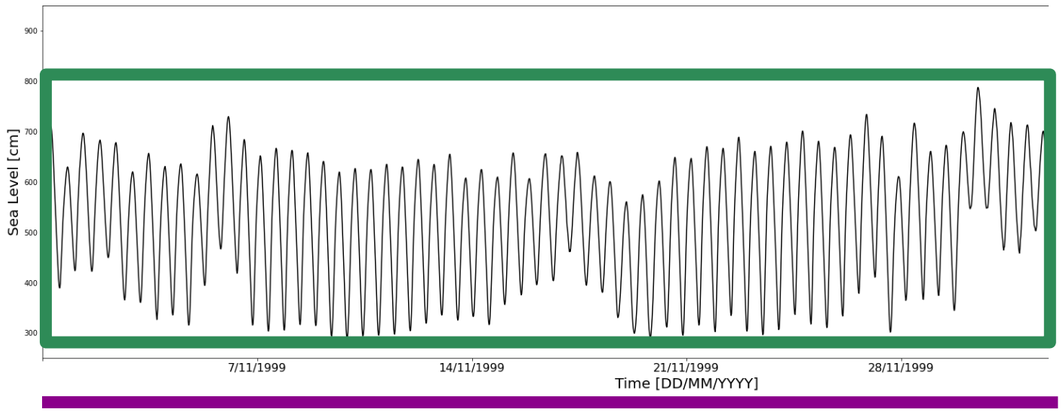
Time-series data

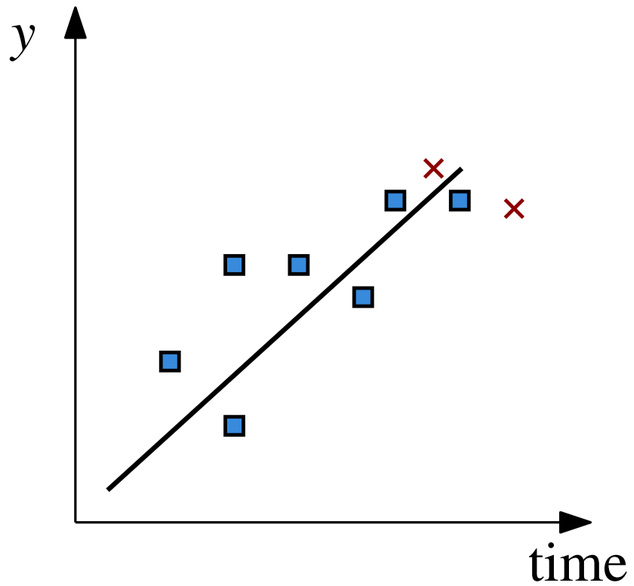
- Measurement of sea level at an oceanographic tide gauge station



$$I = x_{n-t} \rightarrow x_{n-t+1} \rightarrow \ldots \rightarrow x_n \qquad \Rightarrow \qquad o = x_{n+1}$$

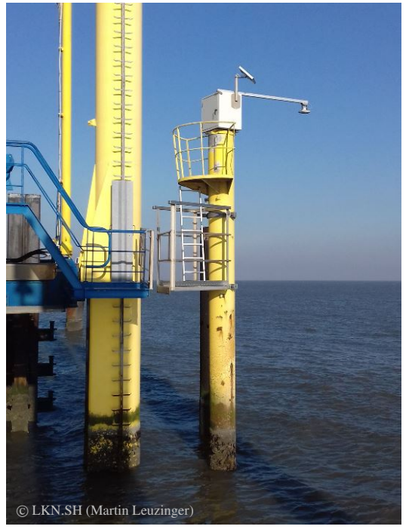
- Modeling of sequences of variable length
- Holding on to long-term dependencies





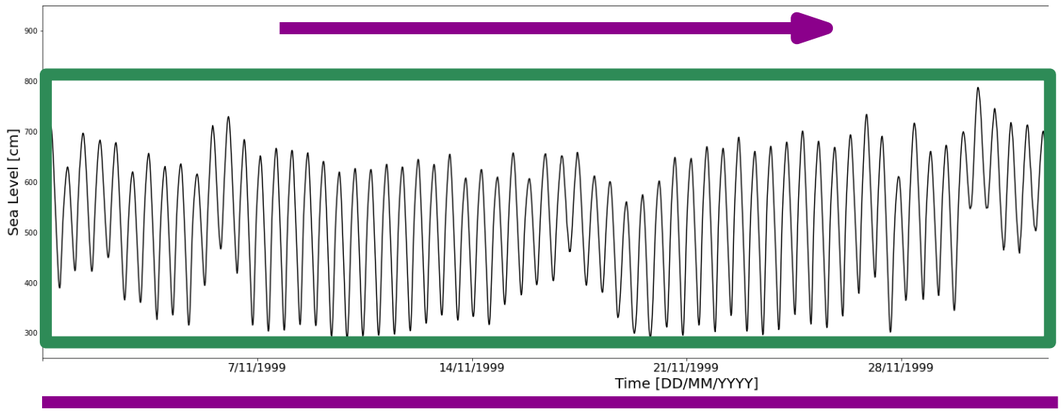
Time-series data

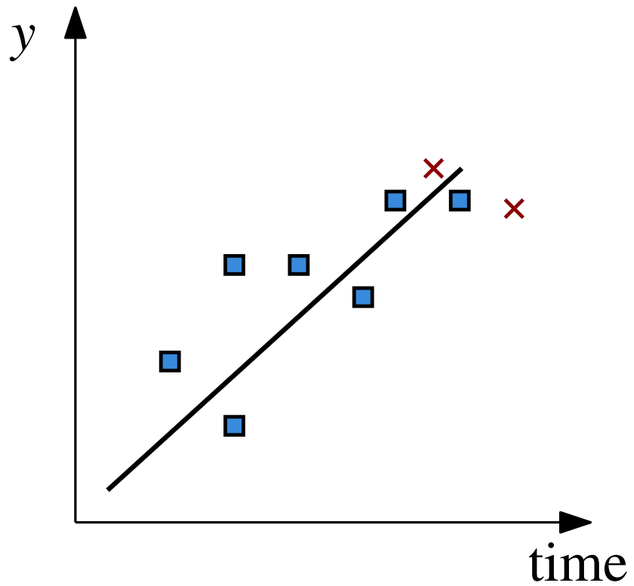
- Measurement of sea level at an oceanographic tide gauge station



$$I = x_{n-t} \rightarrow x_{n-t+1} \rightarrow \ldots \rightarrow x_n \qquad \Rightarrow \qquad o = x_{n+1}$$

- Modeling of sequences of variable length
- Holding on to long-term dependencies
- Propagate parameters along the sequence





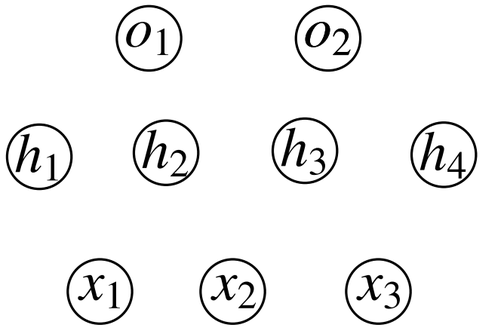
Neural networks for time-series analysis

- Recurrent neural networks (RNNs)
- Long short-term memory networks (LSTM)
- Convolutional neural networks with temporal convolutions (temporal CNNs)

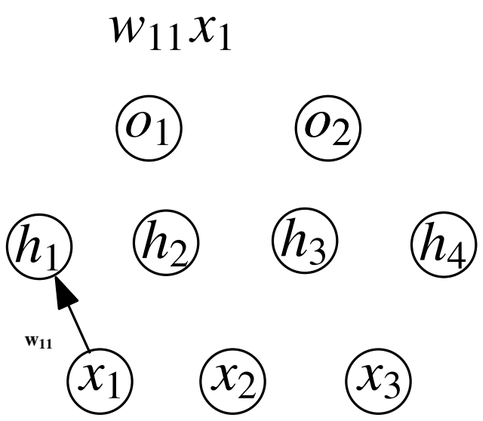
Feed-forward neural network vs. recurrent neural network

- Name indicates how information is channeled
- In a feed-forward network the information only goes in one direction (directly from input to output)
- Not designed for sequences
- Does not model memory
- RNNs are especially designed for sequences and extracting information from them
- RNNs consider the current input and what it has learned before (**hidden state**)

Recurrent Neural Networks (RNNs)

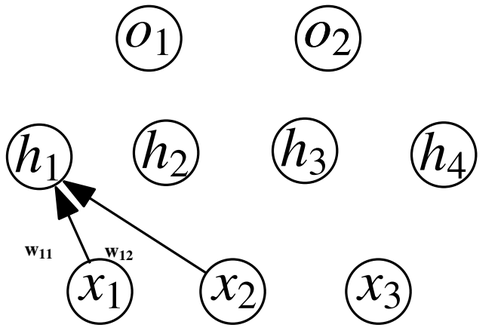


Recurrent Neural Networks (RNNs)



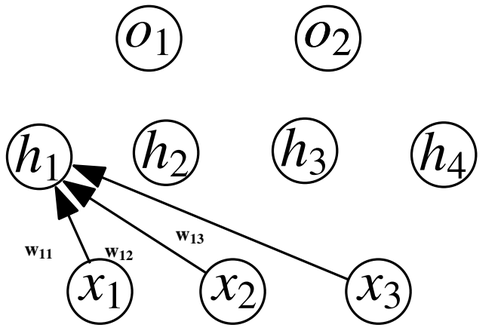
Recurrent Neural Networks (RNNs)

$$w_{11}x_1 + w_{12}x_2$$



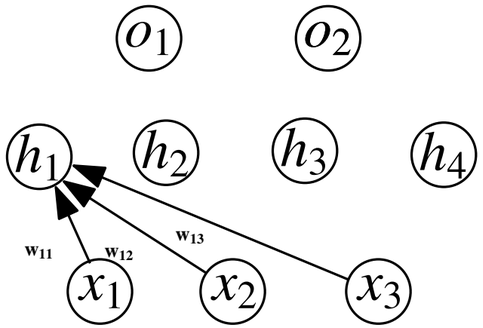
Recurrent Neural Networks (RNNs)

$$w_{11}x_1 + w_{12}x_2 + w_{13}x_3$$



Recurrent Neural Networks (RNNs)

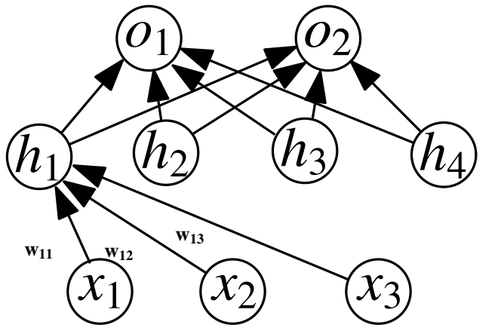
$$h_1 = g(w_{11}x_1 + w_{12}x_2 + w_{13}x_3)$$



Recurrent Neural Networks (RNNs)

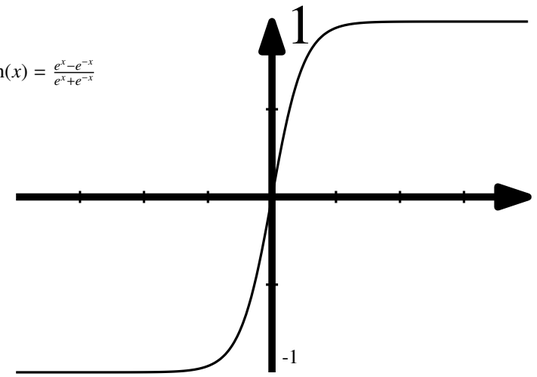
$$h_i = g(\sum jw_{ij}x_j)$$

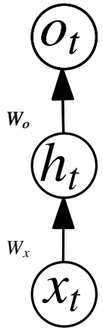
$$h_1 = g(w_{11}x_1 + w_{12}x_2 + w_{13}x_3)$$



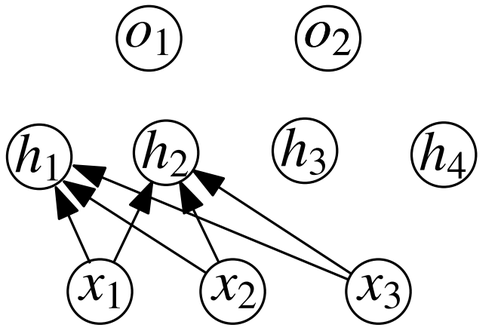
Recurrent Neural Networks (RNNs)

$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

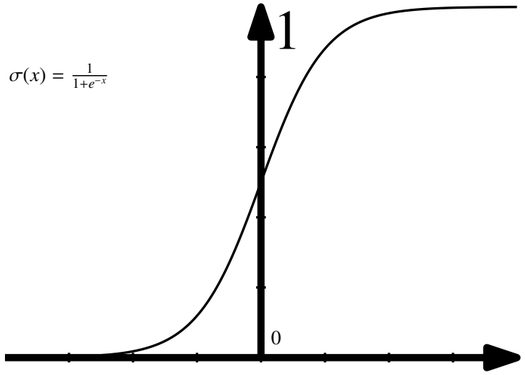




$$h_i = g(\sum jw_{ij}x_j)$$



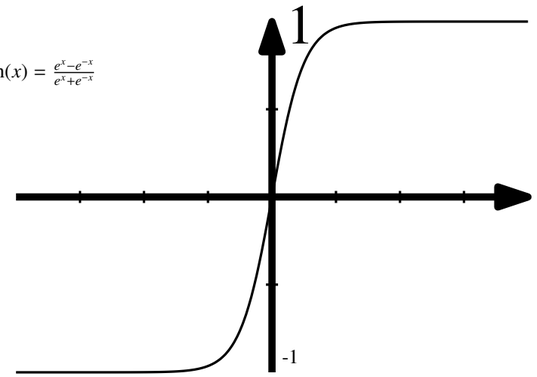
$\sigma(x) = \frac{1}{1+e^{-x}}$

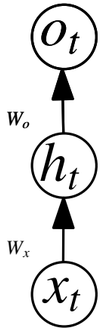


Recurrent Neural Networks (RNNs)

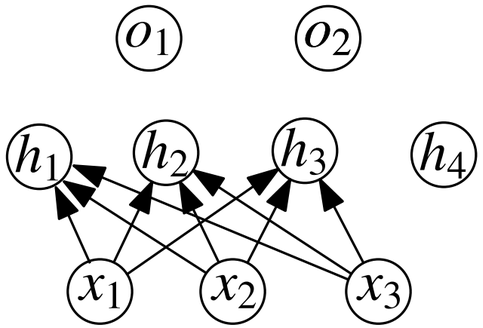
HafenCity Universität Hamburg

$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

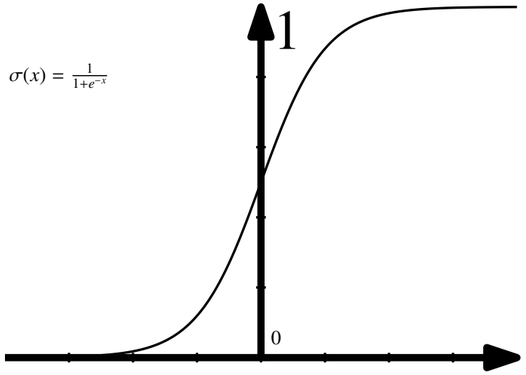




$$h_i = g(\sum jw_{ij}x_j)$$

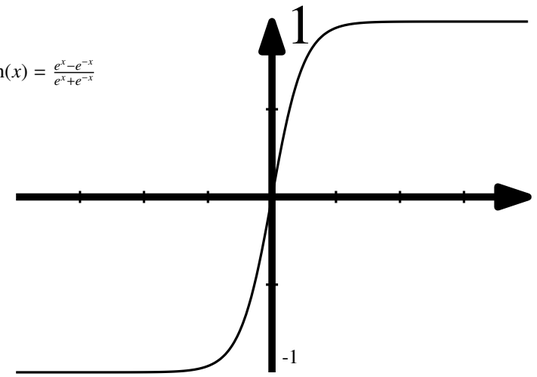


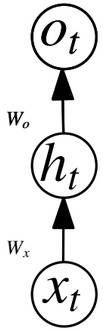
$\sigma(x) = \frac{1}{1+e^{-x}}$



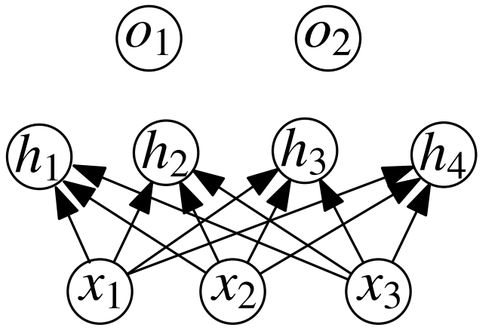
Recurrent Neural Networks (RNNs)

$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

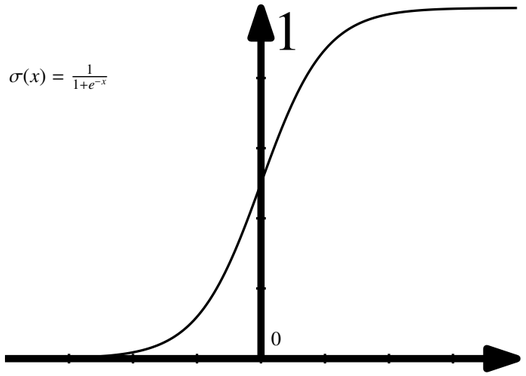




$$h_i = g(\sum jw_{ij}x_j)$$



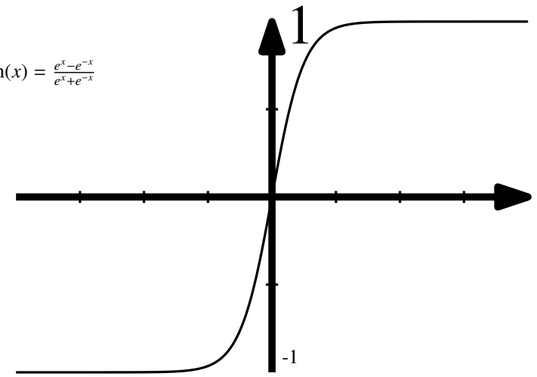
$\sigma(x) = \frac{1}{1+e^{-x}}$



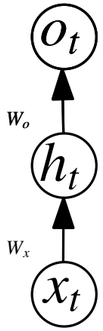
Recurrent Neural Networks (RNNs)

HafenCity Universität Hamburg

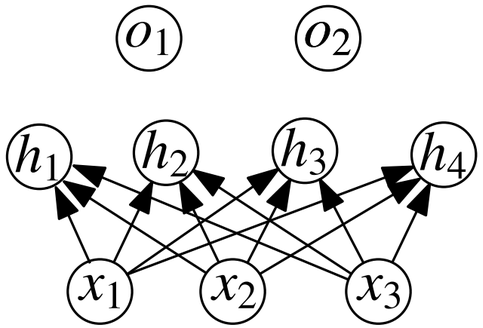
$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$



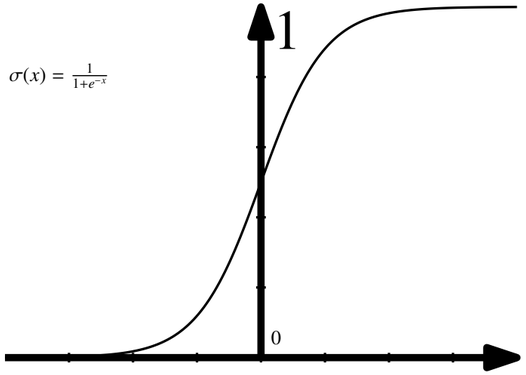
$$h_t = W_x x_t$$



$$h_i = g(\sum jw_{ij}x_j)$$



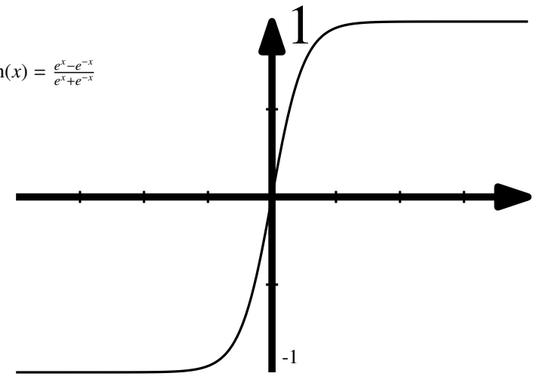
$\sigma(x) = \frac{1}{1+e^{-x}}$

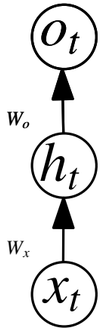


Recurrent Neural Networks (RNNs)

HafenCity Universität Hamburg

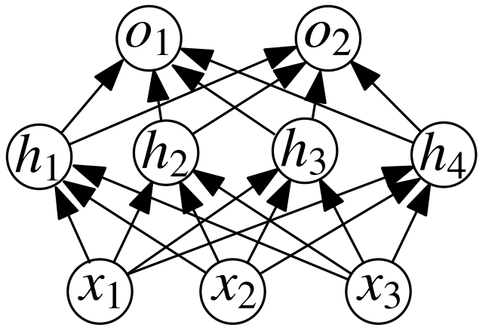
$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$



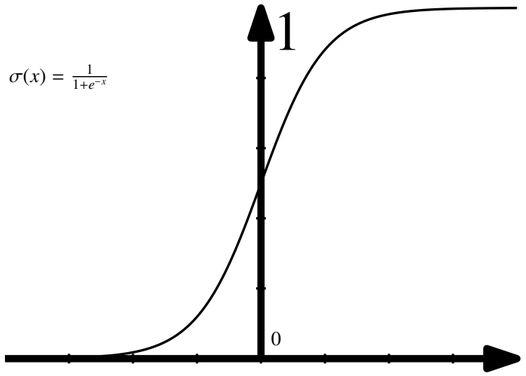


$$h_t = W_x x_t$$

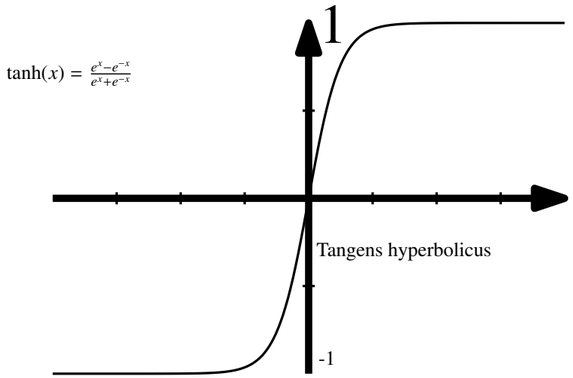
$$o_t = W_o g(h_t)$$

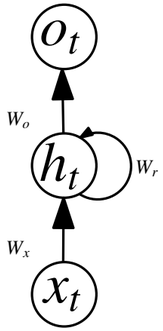


$\sigma(x) = \frac{1}{1+e^{-x}}$



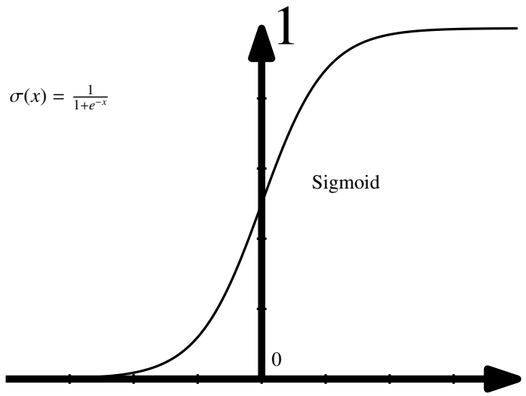
Recurrent Neural Networks (RNNs)



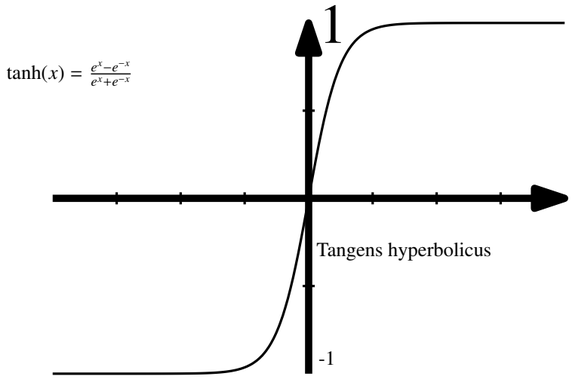


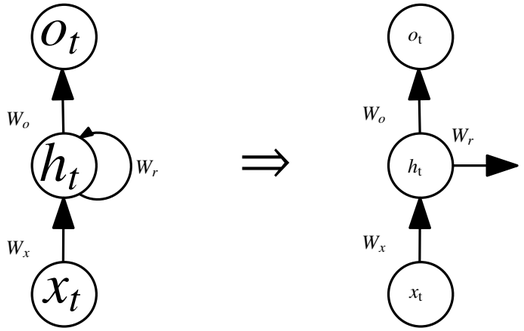
$$h_t = W_x x_t + W_r g(h_{t-1})$$

$$o_t = W_o g(h_t)$$



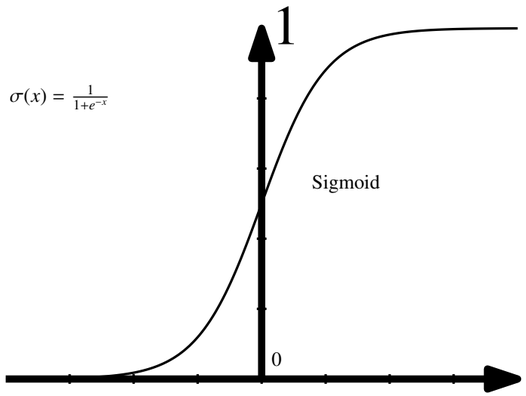
Recurrent Neural Networks (RNNs)



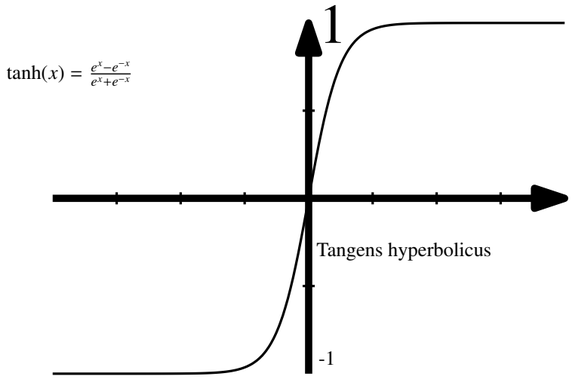


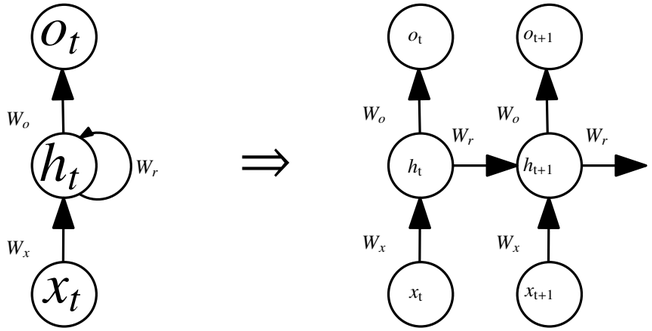
$$h_t = W_x x_t + W_r g(h_{t-1})$$

$$o_t = W_o g(h_t)$$



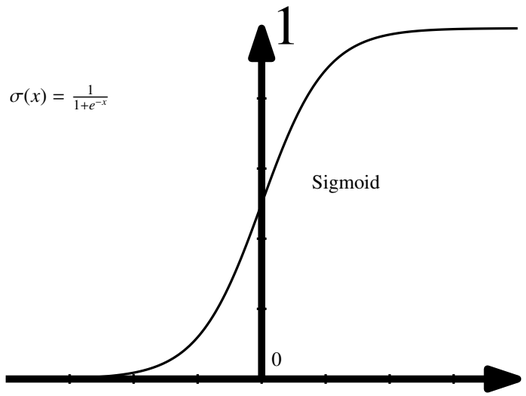
Recurrent Neural Networks (RNNs)



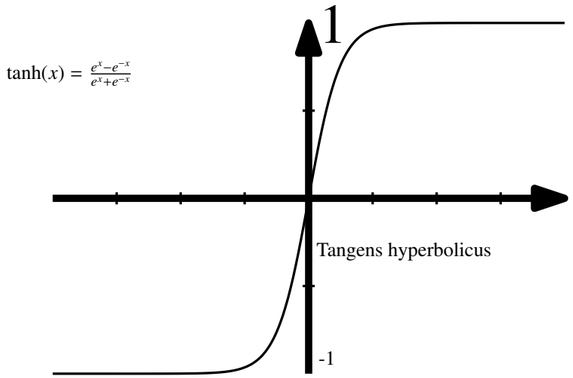


$$h_t = W_x x_t + W_r g(h_{t-1})$$

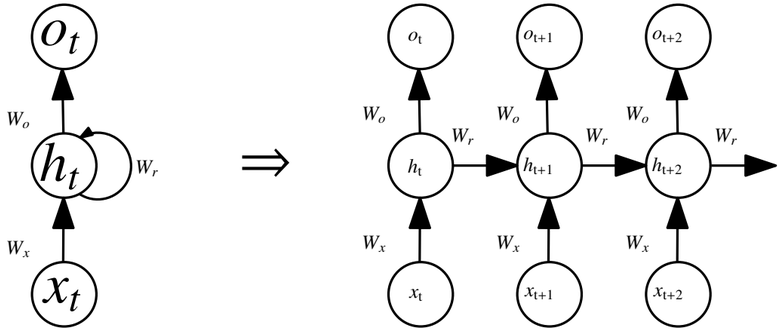
$$o_t = W_o g(h_t)$$



Recurrent Neural Networks (RNNs)

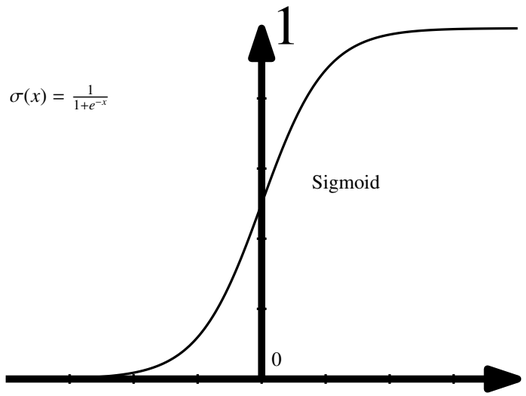


$$x_n = \begin{bmatrix} x_{n-1} \\ \vdots \\ x_{n-t} \end{bmatrix}$$

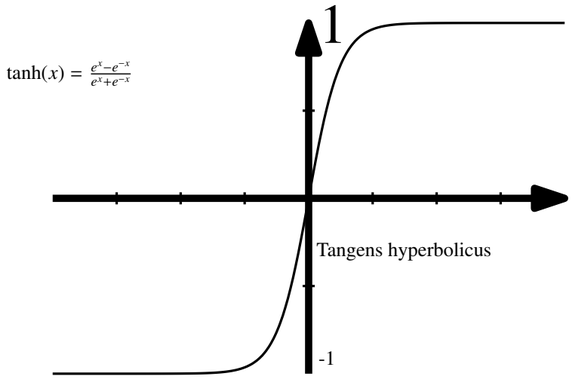


$$h_t = W_x x_t + W_r g(h_{t-1})$$

$$o_t = W_o g(h_t)$$

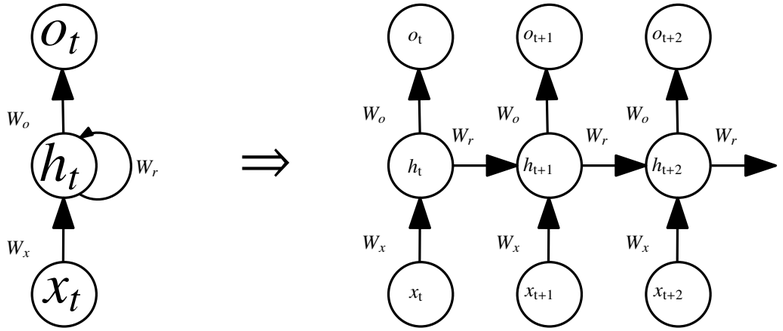


Recurrent Neural Networks (RNNs)



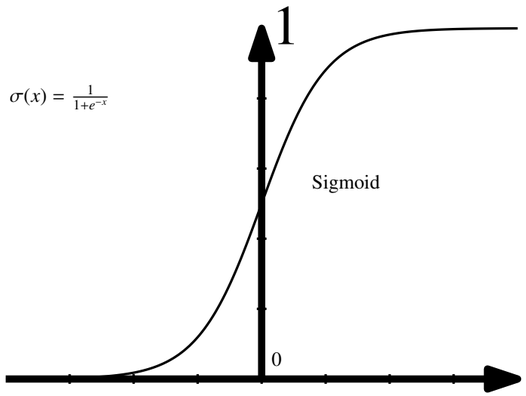
$$x_n = \begin{bmatrix} x_{n-1} \\ \vdots \\ x_{n-t} \end{bmatrix}$$

$$x_{n-t} \rightarrow x_{n-t+1} \rightarrow \ldots \rightarrow x_n$$



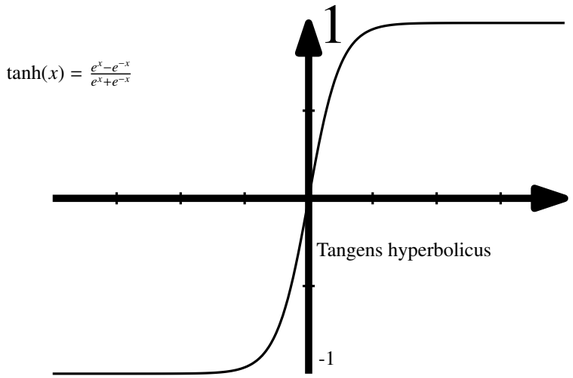
$$h_t = W_x x_t + W_r g(h_{t-1})$$

$$o_t = W_o g(h_t)$$



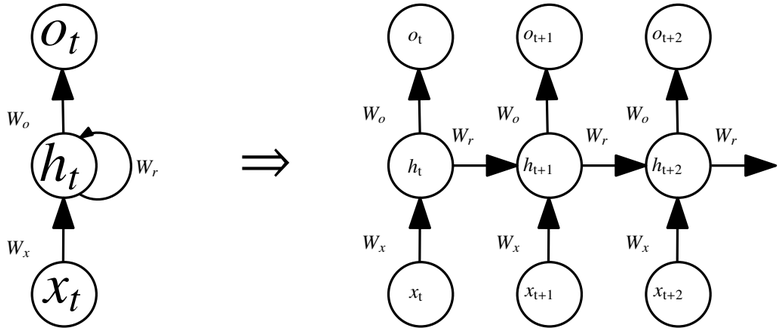
Recurrent Neural Networks (RNNs)

- Hidden state at previous time step flows to the next time step



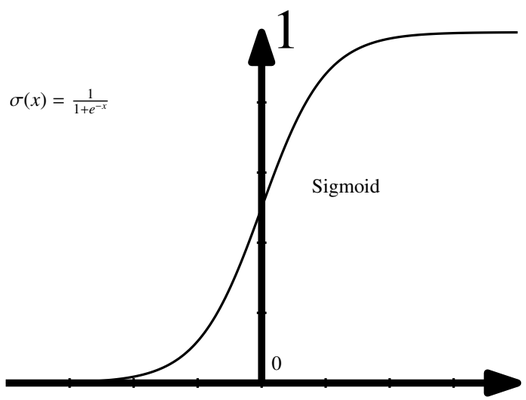
$$x_n = \begin{bmatrix} x_{n-1} \\ \vdots \\ x_{n-t} \end{bmatrix} \qquad o = x_{n+1}$$

$$x_{n-t} \rightarrow x_{n-t+1} \rightarrow \ldots \rightarrow x_n \quad ?$$



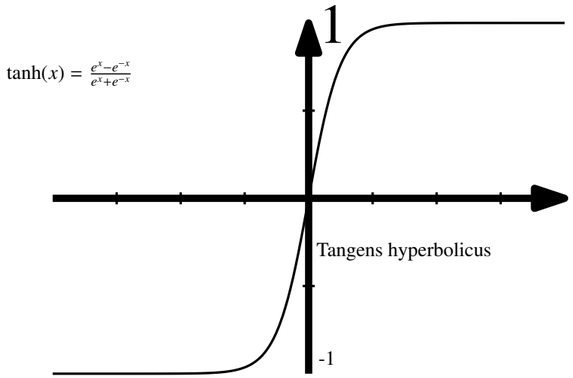
$$h_t = W_x x_t + W_r g(h_{t-1})$$

$$o_t = W_o g(h_t)$$



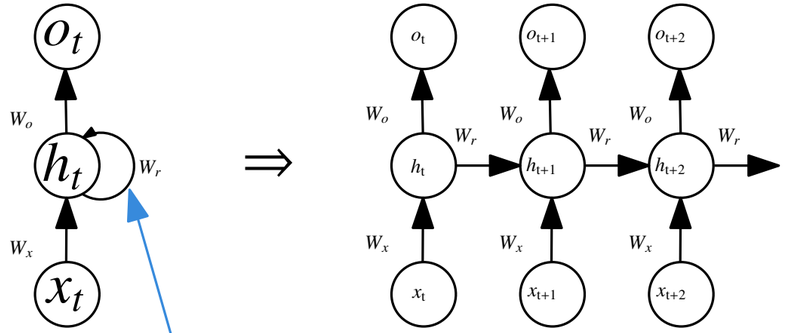
Recurrent Neural Networks (RNNs)

- Hidden state at previous time step flows to the next time step



$$x_n = \begin{bmatrix} x_{n-1} \\ \vdots \\ x_{n-t} \end{bmatrix} \qquad o = x_{n+1}$$

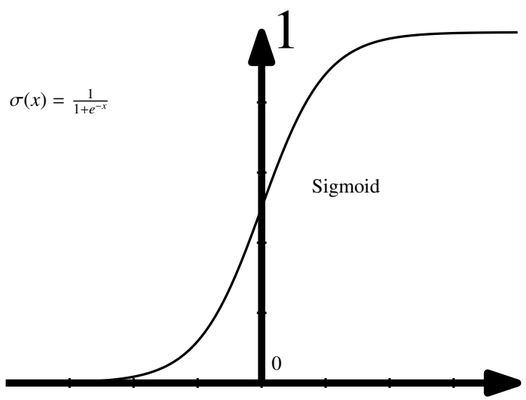
$$x_{n-t} \rightarrow x_{n-t+1} \rightarrow \ldots \rightarrow x_n \quad ?$$



$$h_t = W_x x_t + W_r g(h_{t-1})$$

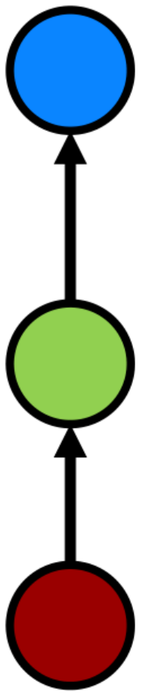
$$o_t = W_o g(h_t)$$

- transport of information and memory



Flexibility of RNNs

one to one



vanilla neural network: image categorization

apple, healthy

tomato, diseased

grapevine, diseased

Flexibility of RNNs

one to many

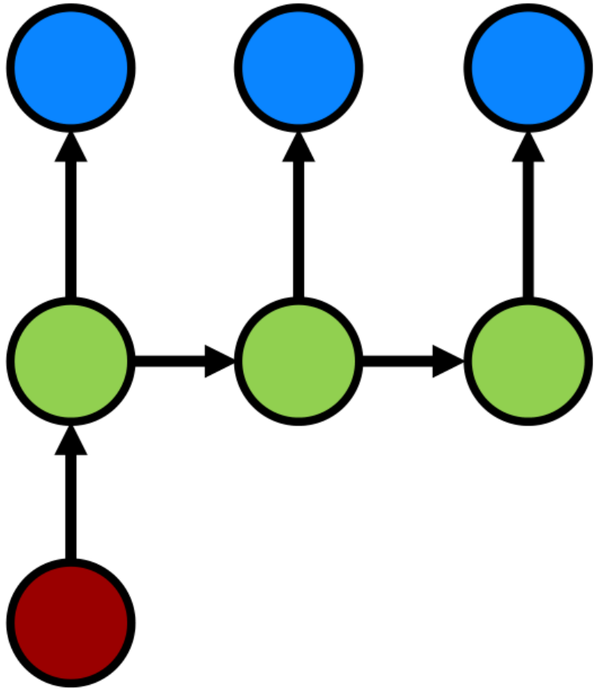
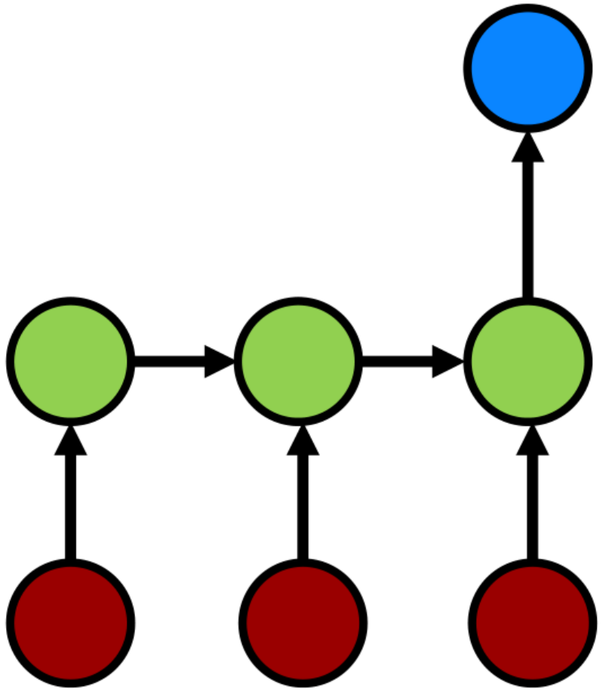


image captioning: image to text

„There are many berries in the image, some are not ripe.“

Flexibility of RNNs

many to one



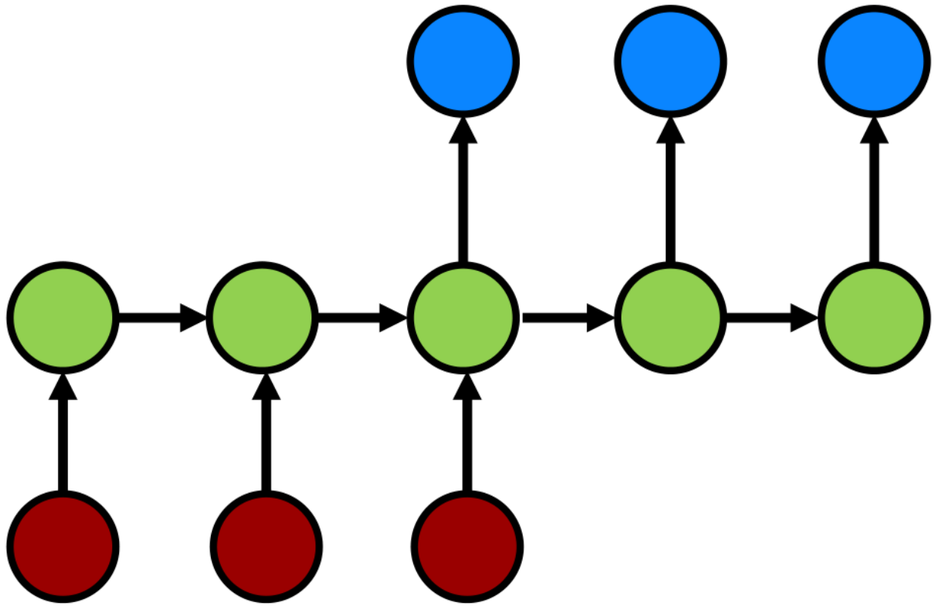
nowcasting: time-series to property

drought stress

rainfall of the last days

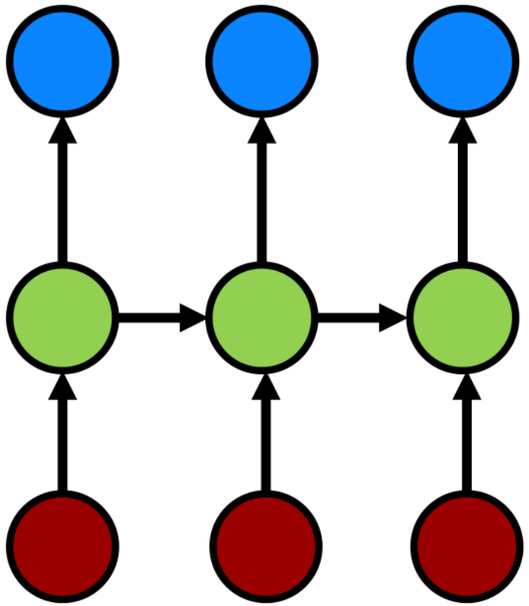
Flexibility of RNNs

many to many



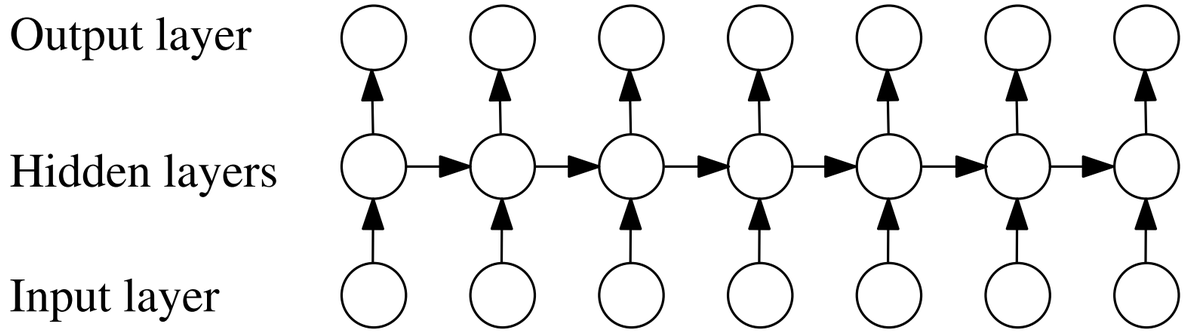
forecasting: time-series to time-series

many to many



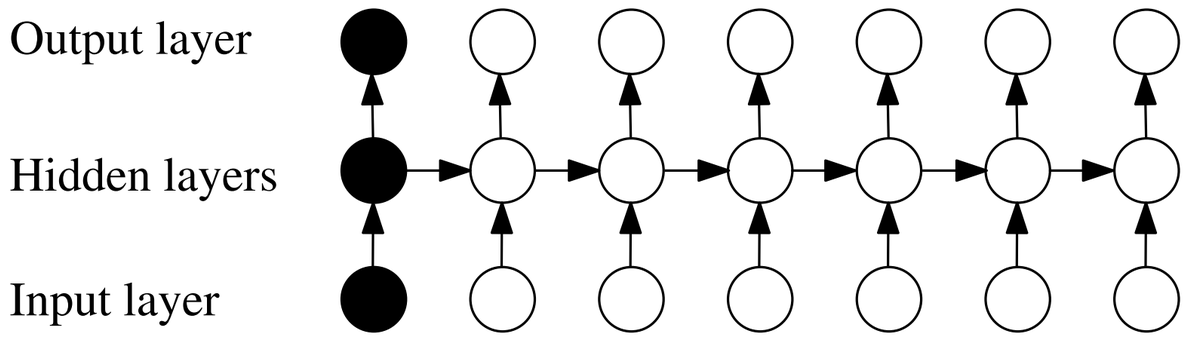
monitoring: time-series to properties

Vanishing gradient



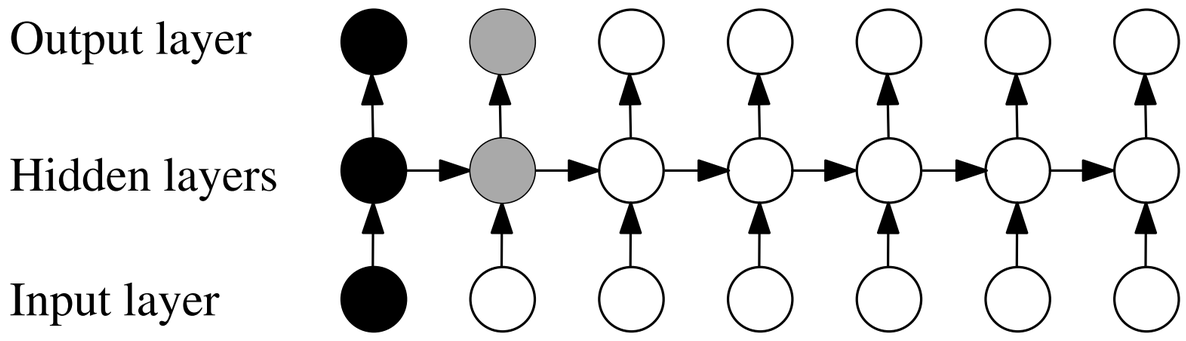
Vanishing gradient

New inputs override activation of hidden layers



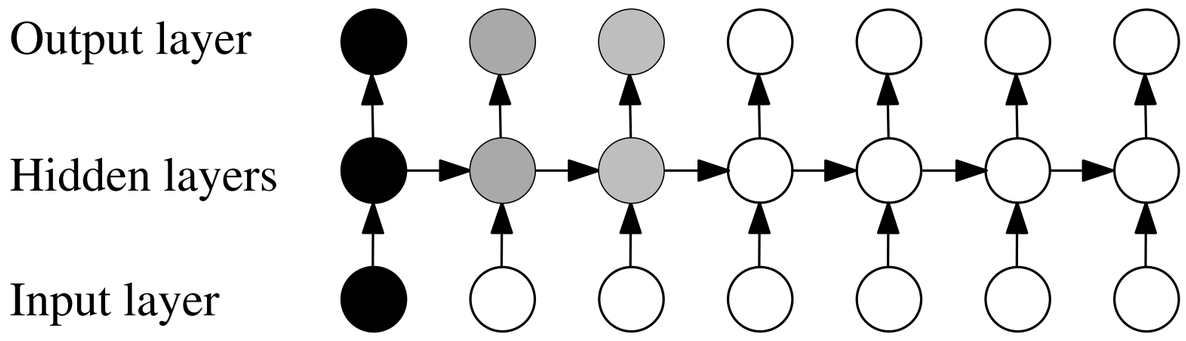
Vanishing gradient

New inputs override activation of hidden layers



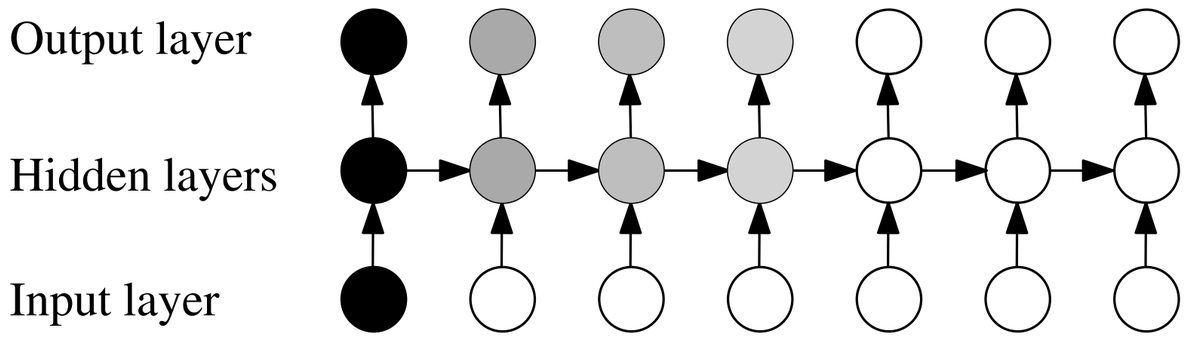
Vanishing gradient

New inputs override activation of hidden layers



Vanishing gradient

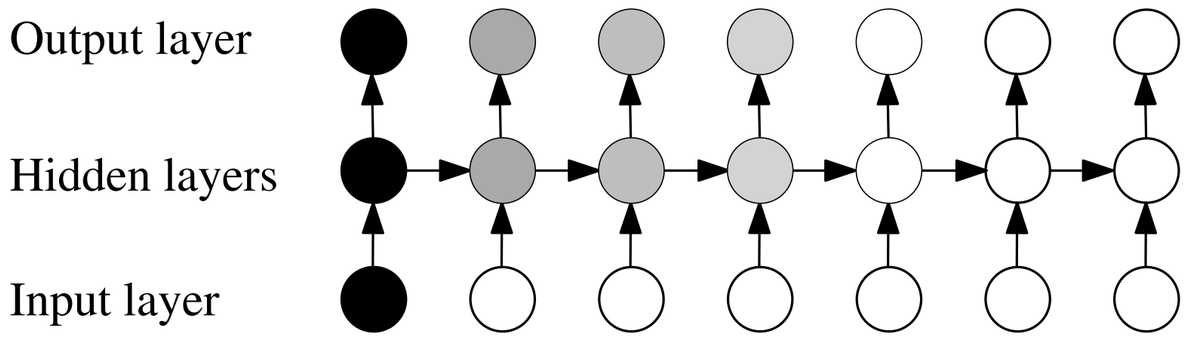
New inputs override activation of hidden layers



Vanishing gradient

New inputs override activation of hidden layers

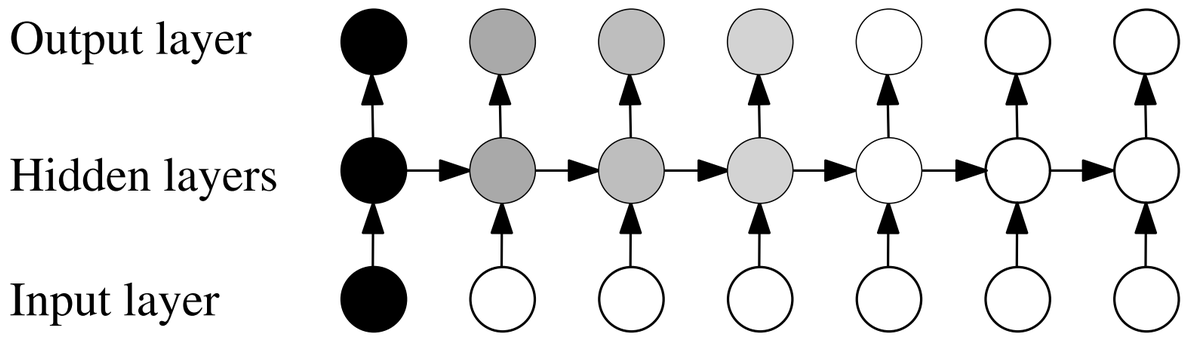
⇒ Network forgets first inputs



Vanishing gradient

New inputs override activation of hidden layers

⇒ Network forgets first inputs

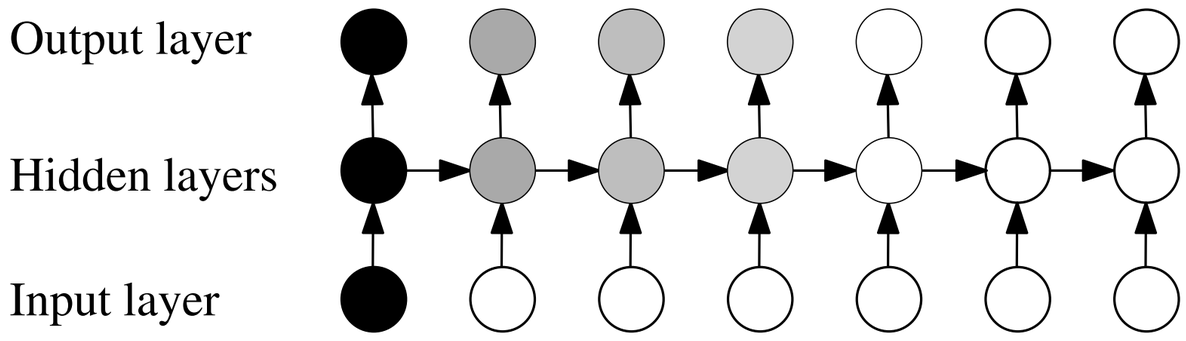


Poorly mappable long-term memory
⇒ Patterns with long dependencies not learnable

Vanishing gradient

New inputs override activation of hidden layers

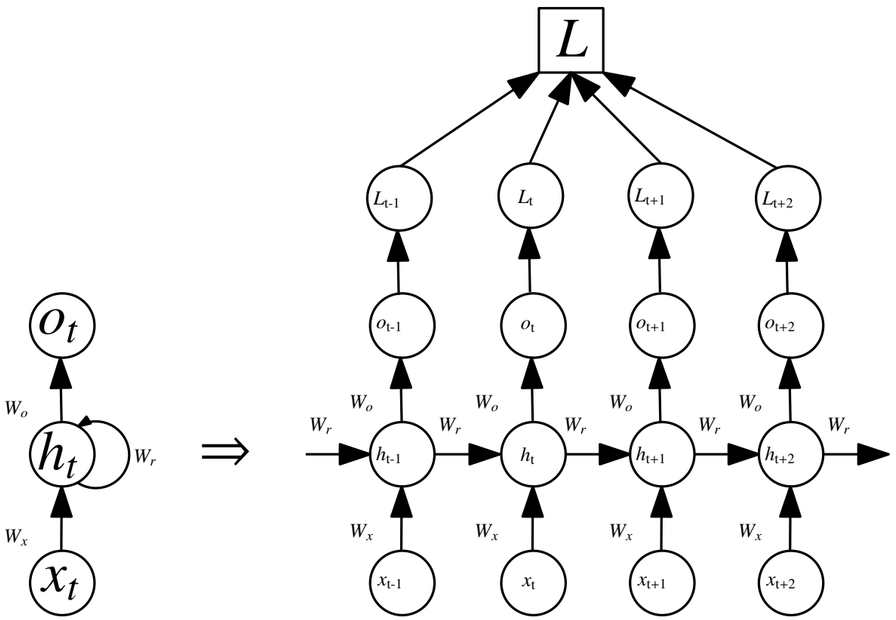
⇒ Network forgets first inputs



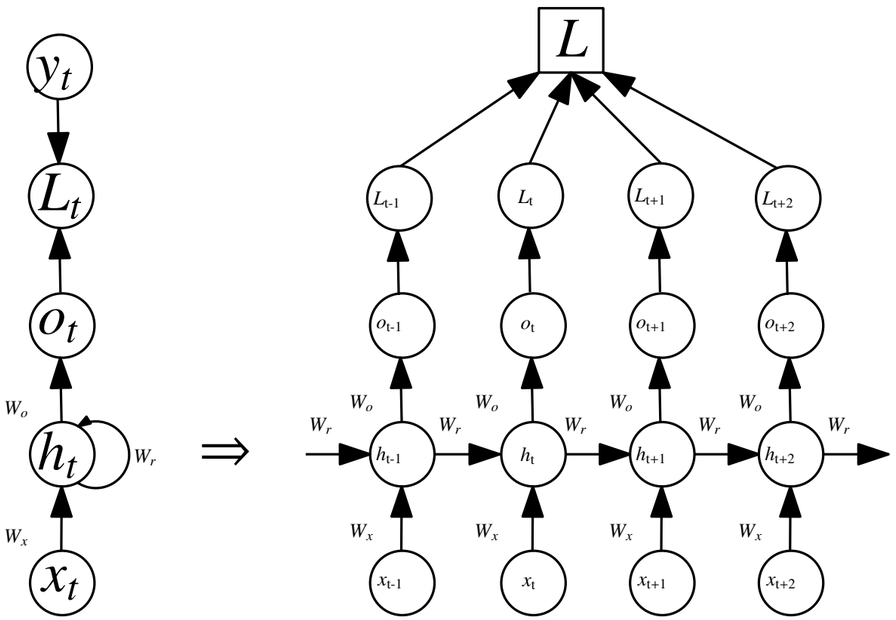
Poorly mappable long-term memory
⇒ Patterns with long dependencies not learnable

Cause ⇒ **vanishing gradient**

Problem with training of RNNs



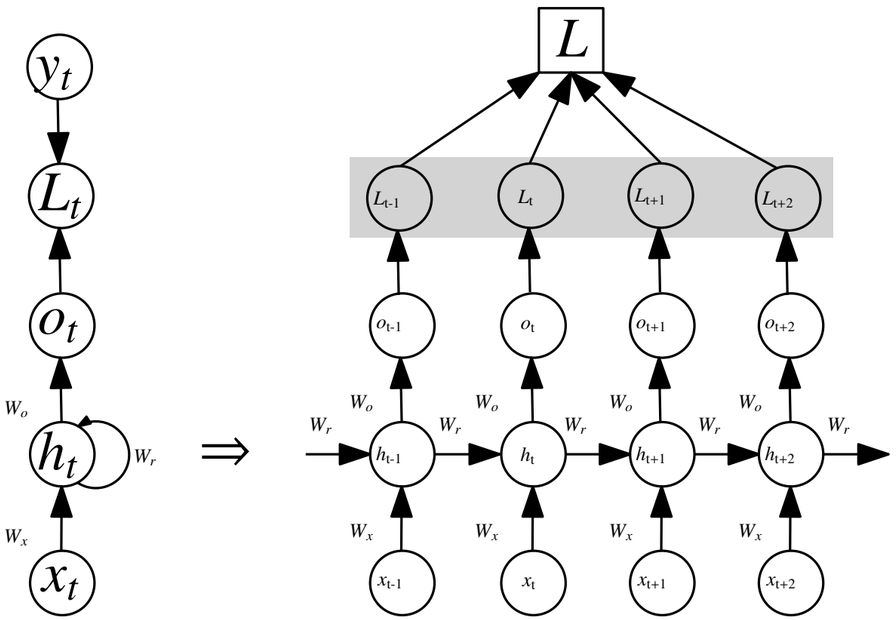
Problem with training of RNNs



Problem with training of RNNs

$$\frac{\partial L}{\partial W_r} = \sum_{t=1}^{s} \frac{\partial L_t}{\partial W_r}$$

$$h_t = W_x x_t + W_r g(h_{t-1})$$
$$o_t = W_o g(h_t)$$

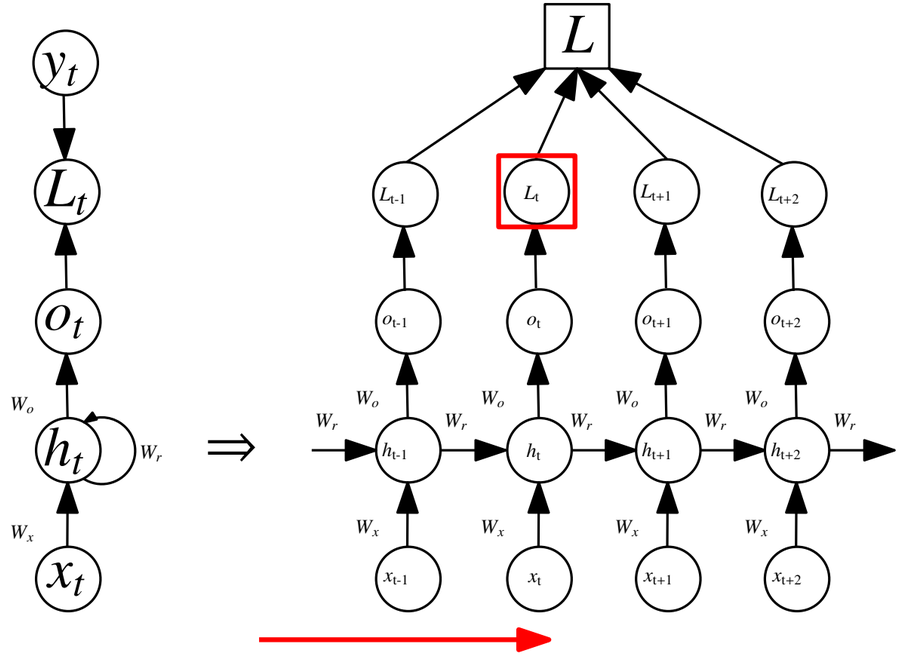


Problem with training of RNNs

$$\frac{\partial L}{\partial W_r} = \sum_{t=1}^{s} \frac{\partial L_t}{\partial W_r}$$

$$h_t = W_x x_t + W_r g(h_{t-1})$$
$$o_t = W_o g(h_t)$$

$$\boxed{\frac{\partial L_t}{\partial W_r}} = \sum_{k=1}^{t} \frac{\partial L_t}{\partial o_t} \frac{\partial o_t}{\partial h_t} \frac{\partial h_t}{\partial h_k} \frac{\partial h_k}{\partial w_r}$$

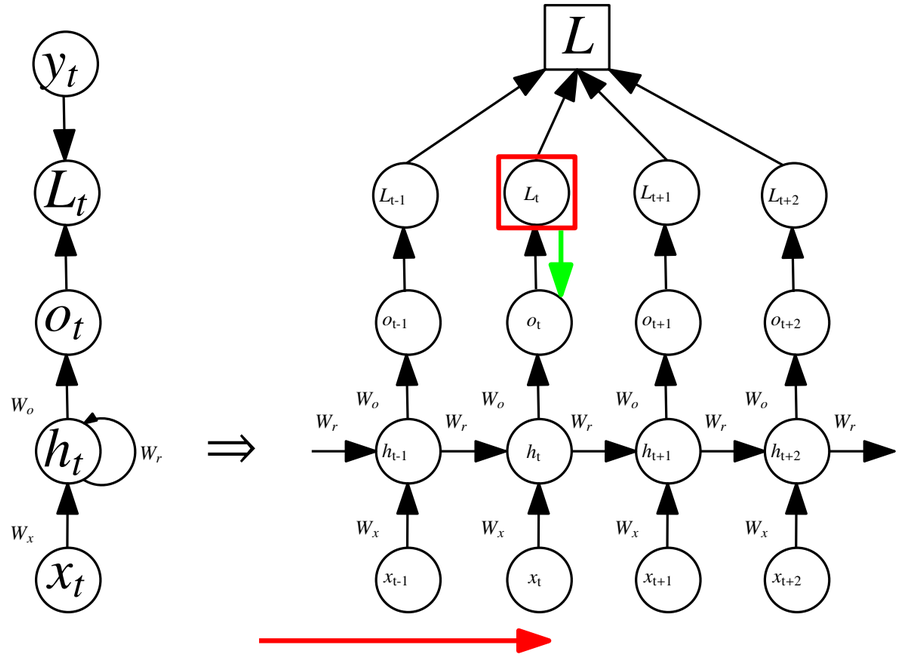


Problem with training of RNNs

$$\frac{\partial L}{\partial W_r} = \sum_{t=1}^{s} \frac{\partial L_t}{\partial W_r}$$

$$h_t = W_x x_t + W_r g(h_{t-1})$$
$$o_t = W_o g(h_t)$$

$$\boxed{\frac{\partial L_t}{\partial W_r}} = \sum_{k=1}^{t} \boxed{\frac{\partial L_t}{\partial o_t}} \frac{\partial o_t}{\partial h_t} \frac{\partial h_t}{\partial h_k} \frac{\partial h_k}{\partial w_r}$$

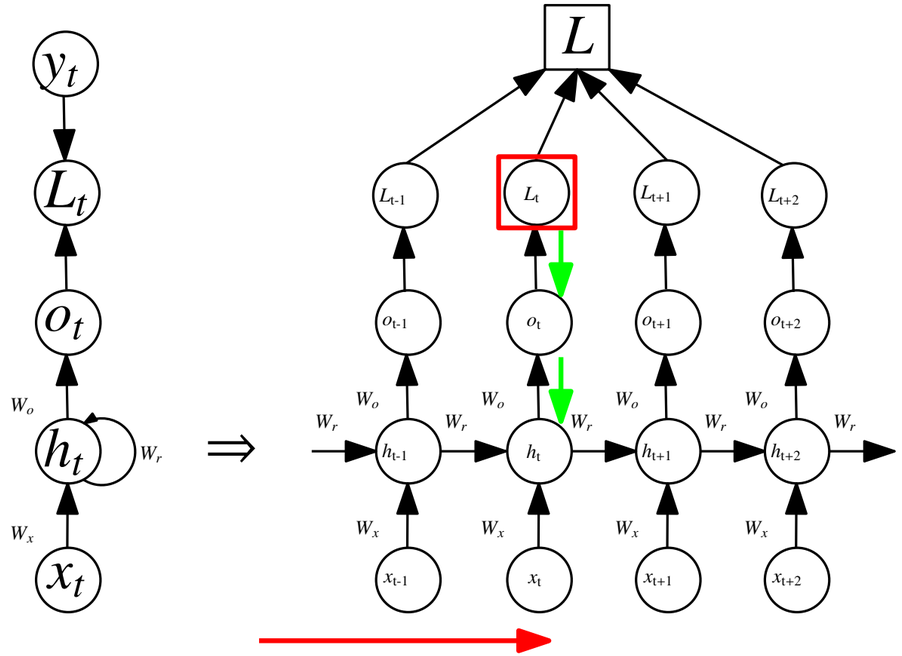


Problem with training of RNNs

$$\frac{\partial L}{\partial W_r} = \sum_{t=1}^{s} \frac{\partial L_t}{\partial W_r}$$

$$h_t = W_x x_t + W_r g(h_{t-1})$$
$$o_t = W_o g(h_t)$$

$$\boxed{\frac{\partial L_t}{\partial W_r}} = \sum_{k=1}^{t} \boxed{\frac{\partial L_t}{\partial o_t}} \boxed{\frac{\partial o_t}{\partial h_t}} \frac{\partial h_t}{\partial h_k} \frac{\partial h_k}{\partial w_r}$$

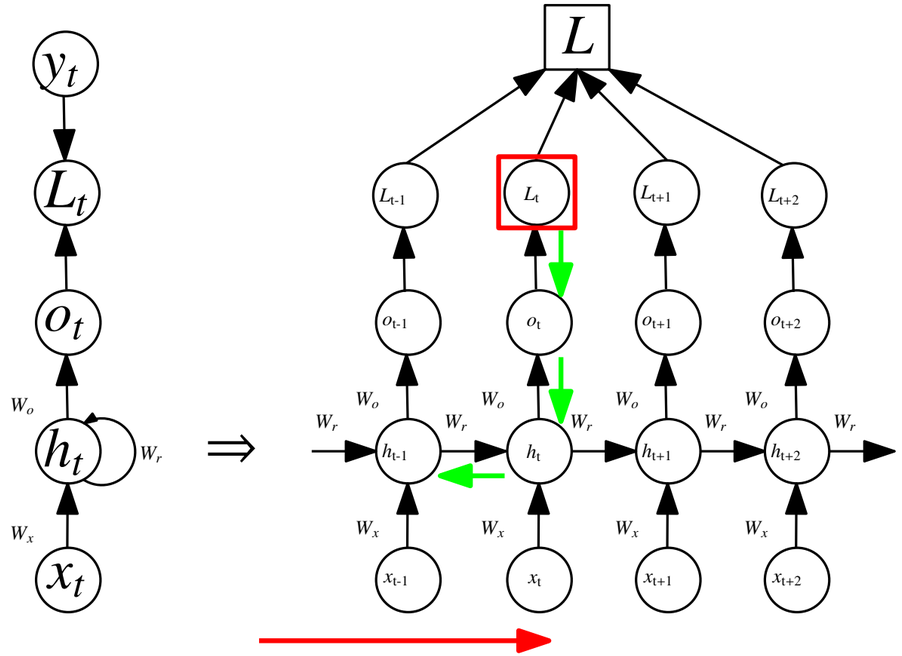


Problem with training of RNNs

$$\frac{\partial L}{\partial W_r} = \sum_{t=1}^{s} \frac{\partial L_t}{\partial W_r}$$

$$h_t = W_x x_t + W_r g(h_{t-1})$$
$$o_t = W_o g(h_t)$$

$$\boxed{\frac{\partial L_t}{\partial W_r}} = \sum_{k=1}^{t} \boxed{\frac{\partial L_t}{\partial o_t}}\boxed{\frac{\partial o_t}{\partial h_t}}\boxed{\frac{\partial h_t}{\partial h_k}}\frac{\partial h_k}{\partial w_r}$$

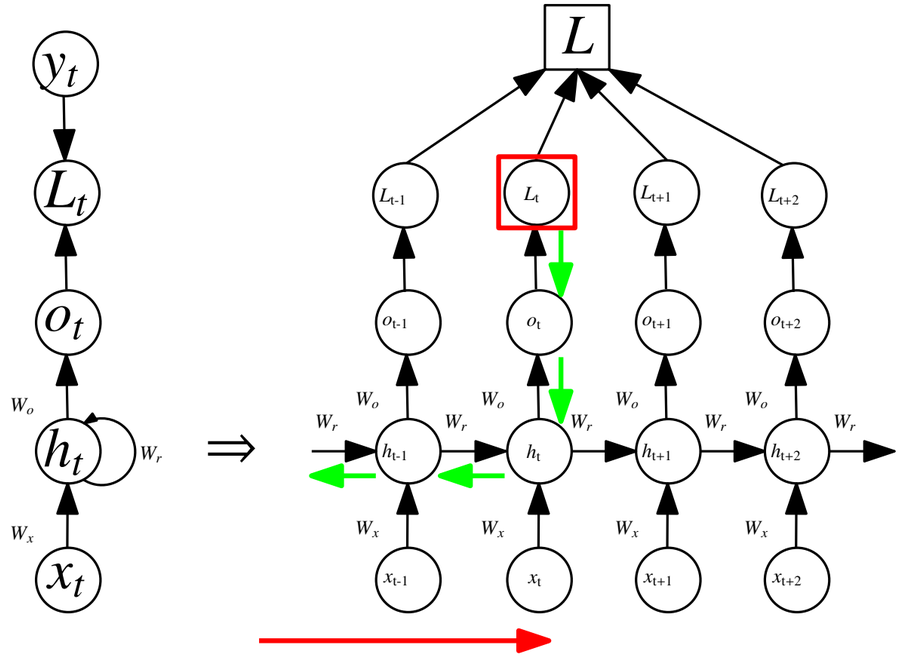


Problem with training of RNNs

$$\frac{\partial L}{\partial W_r} = \sum_{t=1}^{s} \frac{\partial L_t}{\partial W_r}$$

$$h_t = W_x x_t + W_r g(h_{t-1})$$
$$o_t = W_o g(h_t)$$

$$\boxed{\frac{\partial L_t}{\partial W_r}} = \sum_{k=1}^{t} \boxed{\frac{\partial L_t}{\partial o_t}}\boxed{\frac{\partial o_t}{\partial h_t}}\boxed{\frac{\partial h_t}{\partial h_k}}\boxed{\frac{\partial h_k}{\partial w_r}}$$



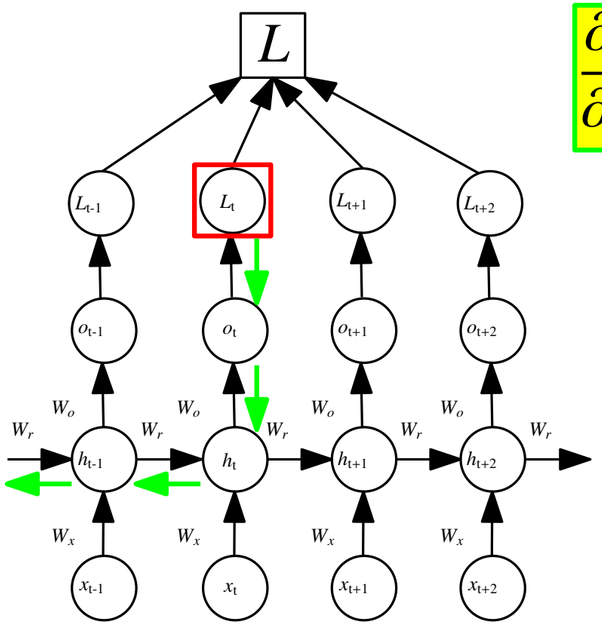
Problem with training of RNNs

$$\frac{\partial L}{\partial W_r} = \sum_{t=1}^{s} \frac{\partial L_t}{\partial W_r}$$

$$h_t = W_x x_t + W_r g(h_{t-1})$$
$$o_t = W_o g(h_t)$$

$$\boxed{\frac{\partial L_t}{\partial W_r}} = \sum_{k=1}^{t} \boxed{\frac{\partial L_t}{\partial o_t}}\boxed{\frac{\partial o_t}{\partial h_t}}\boxed{\frac{\partial h_t}{\partial h_k}}\boxed{\frac{\partial h_k}{\partial w_r}}$$

$$\boxed{\frac{\partial h_t}{\partial h_k} = \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} = \prod_{i=k+1}^{t} W_r g'(h_{i-1})}$$

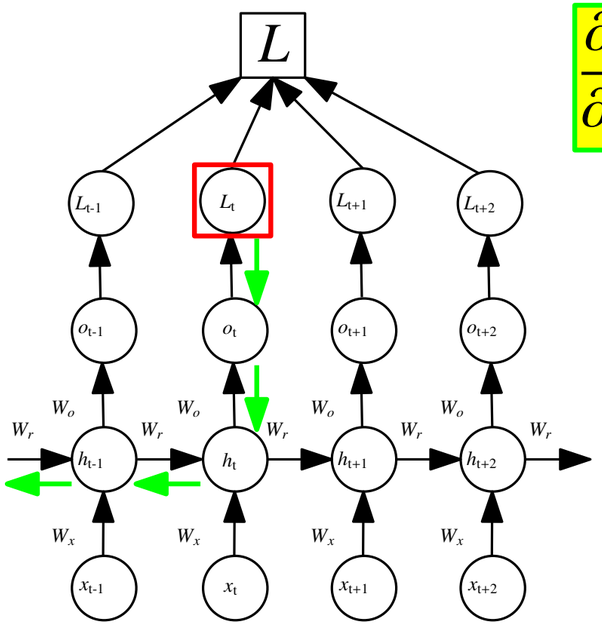


Problem with training of RNNs

$$\frac{\partial L}{\partial W_r} = \sum_{t=1}^{s} \frac{\partial L_t}{\partial W_r}$$

$$h_t = W_x x_t + W_r g(h_{t-1})$$
$$o_t = W_o g(h_t)$$

$$\boxed{\frac{\partial L_t}{\partial W_r}} = \sum_{k=1}^{t} \boxed{\frac{\partial L_t}{\partial o_t}}\boxed{\frac{\partial o_t}{\partial h_t}}\boxed{\frac{\partial h_t}{\partial h_k}}\boxed{\frac{\partial h_k}{\partial w_r}}$$



$$\boxed{\frac{\partial h_t}{\partial h_k} = \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} = \prod_{i=k+1}^{t} W_r g'(h_{i-1})}$$

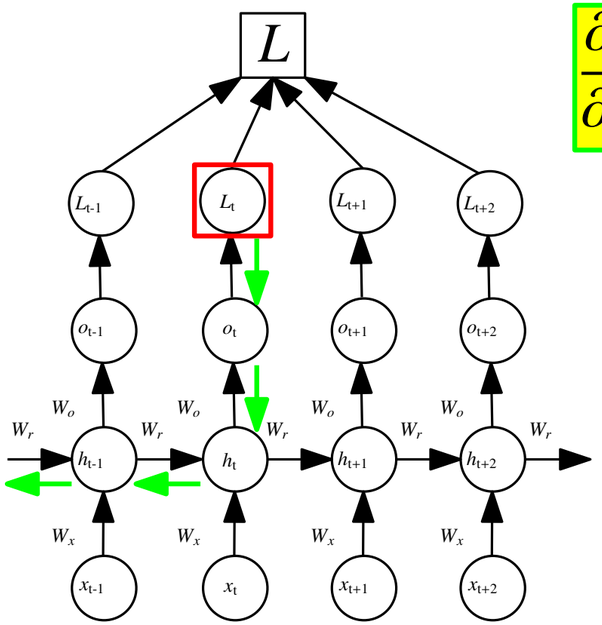
$$\frac{\partial h_i}{\partial h_{i-1}} \leq \|W_r\|\|g'(h_{i-1})\| \leq \gamma_{W_r}\gamma_g$$

Problem with training of RNNs

$$\frac{\partial L}{\partial W_r} = \sum_{t=1}^{s} \frac{\partial L_t}{\partial W_r}$$

$$h_t = W_x x_t + W_r g(h_{t-1})$$
$$o_t = W_o g(h_t)$$

$$\boxed{\frac{\partial L_t}{\partial W_r}} = \sum_{k=1}^{t} \boxed{\frac{\partial L_t}{\partial o_t}}\boxed{\frac{\partial o_t}{\partial h_t}}\boxed{\frac{\partial h_t}{\partial h_k}}\boxed{\frac{\partial h_k}{\partial w_r}}$$



$$\boxed{\frac{\partial h_t}{\partial h_k} = \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} = \prod_{i=k+1}^{t} W_r g'(h_{i-1})}$$

$$\frac{\partial h_i}{\partial h_{i-1}} \leq \|W_r\| \|g'(h_{i-1})\| \leq \gamma_{W_r} \gamma_g$$

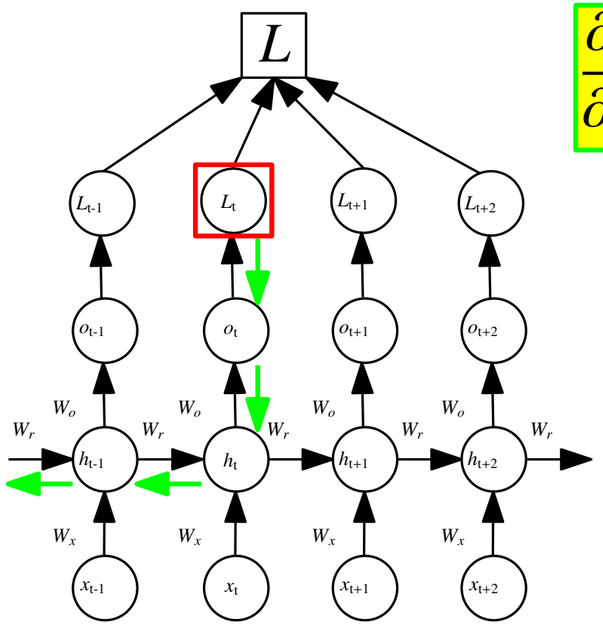
$$\frac{\partial h_t}{\partial h_k} \leq (\gamma_{W_r} \gamma_g)^{t-k}$$

Problem with training of RNNs

$$\frac{\partial L}{\partial W_r} = \sum_{t=1}^{s} \frac{\partial L_t}{\partial W_r}$$

$$h_t = W_x x_t + W_r g(h_{t-1})$$
$$o_t = W_o g(h_t)$$

$$\boxed{\frac{\partial L_t}{\partial W_r}} = \sum_{k=1}^{t} \boxed{\frac{\partial L_t}{\partial o_t}}\boxed{\frac{\partial o_t}{\partial h_t}}\boxed{\frac{\partial h_t}{\partial h_k}}\boxed{\frac{\partial h_k}{\partial w_r}}$$



$$\boxed{\frac{\partial h_t}{\partial h_k} = \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} = \prod_{i=k+1}^{t} W_r g'(h_{i-1})}$$

$$\frac{\partial h_i}{\partial h_{i-1}} \leq \|W_r\| \|g'(h_{i-1})\| \leq \gamma_{W_r} \gamma_g$$

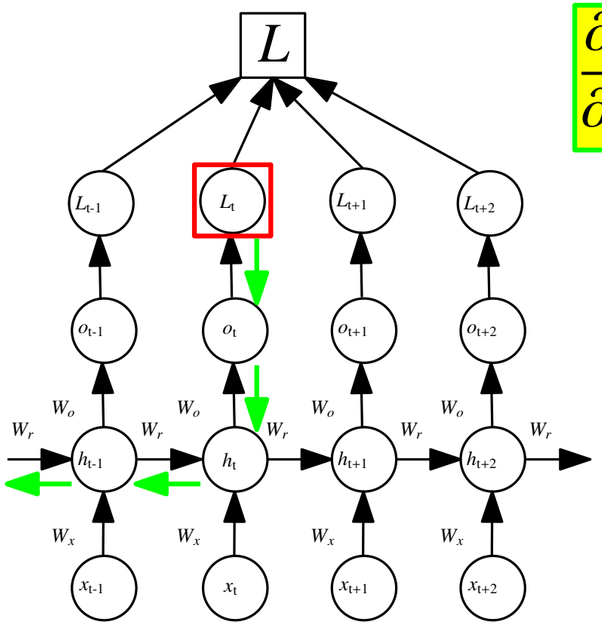
$$\frac{\partial h_t}{\partial h_k} \leq (\gamma_{W_r} \gamma_g)^{\boxed{t-k}}$$

Problem with training of RNNs

$$\frac{\partial L}{\partial W_r} = \sum_{t=1}^{s} \frac{\partial L_t}{\partial W_r}$$

$$h_t = W_x x_t + W_r g(h_{t-1})$$
$$o_t = W_o g(h_t)$$

$$\boxed{\frac{\partial L_t}{\partial W_r}} = \sum_{k=1}^{t} \boxed{\frac{\partial L_t}{\partial o_t}}\boxed{\frac{\partial o_t}{\partial h_t}}\boxed{\frac{\partial h_t}{\partial h_k}}\boxed{\frac{\partial h_k}{\partial w_r}}$$



$$\boxed{\frac{\partial h_t}{\partial h_k} = \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} = \prod_{i=k+1}^{t} W_r g'(h_{i-1})}$$

$$\frac{\partial h_i}{\partial h_{i-1}} \leq \|W_r\|\|g'(h_{i-1})\| \leq \gamma_{W_r}\gamma_g$$

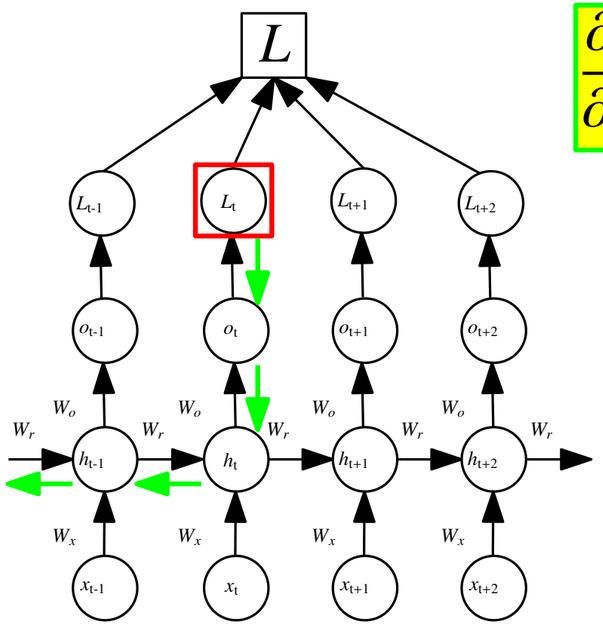
$$\frac{\partial h_t}{\partial h_k} \leq \underbrace{(\gamma_{W_r}\gamma_g)}_{\gamma}{}^{t-k}$$

Problem with training of RNNs

$$\frac{\partial L}{\partial W_r} = \sum_{t=1}^{s} \frac{\partial L_t}{\partial W_r}$$

$$h_t = W_x x_t + W_r g(h_{t-1})$$
$$o_t = W_o g(h_t)$$

$$\boxed{\frac{\partial L_t}{\partial W_r}} = \sum_{k=1}^{t} \boxed{\frac{\partial L_t}{\partial o_t}}\boxed{\frac{\partial o_t}{\partial h_t}}\boxed{\frac{\partial h_t}{\partial h_k}}\boxed{\frac{\partial h_k}{\partial w_r}}$$



$$\boxed{\frac{\partial h_t}{\partial h_k} = \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} = \prod_{i=k+1}^{t} W_r g'(h_{i-1})}$$

$$\frac{\partial h_i}{\partial h_{i-1}} \leq \|W_r\| \|g'(h_{i-1})\| \leq \gamma_{W_r} \gamma_g$$

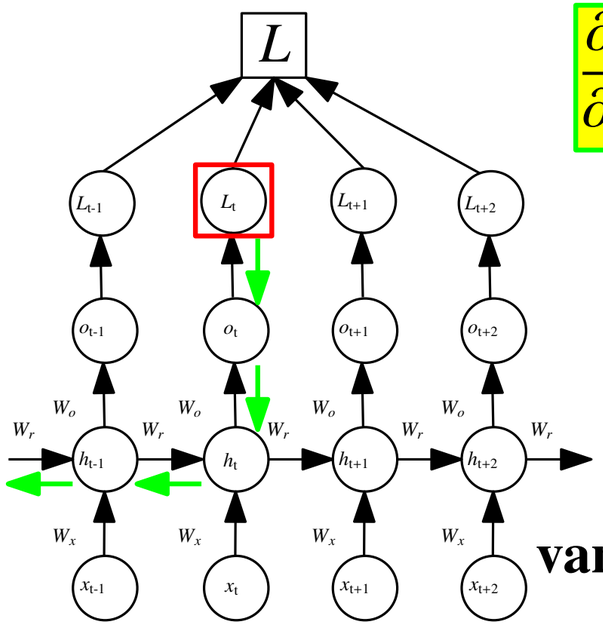
$$\frac{\partial h_t}{\partial h_k} \leq \underbrace{(\gamma_{W_r} \gamma_g)}_{0 < \gamma < 1}{}^{t-k}$$

Problem with training of RNNs

$$\frac{\partial L}{\partial W_r} = \sum_{t=1}^{s} \frac{\partial L_t}{\partial W_r}$$

$$h_t = W_x x_t + W_r g(h_{t-1})$$
$$o_t = W_o g(h_t)$$

$$\boxed{\frac{\partial L_t}{\partial W_r}} = \sum_{k=1}^{t} \boxed{\frac{\partial L_t}{\partial o_t}}\boxed{\frac{\partial o_t}{\partial h_t}}\boxed{\frac{\partial h_t}{\partial h_k}}\boxed{\frac{\partial h_k}{\partial w_r}}$$



$$\boxed{\frac{\partial h_t}{\partial h_k} = \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} = \prod_{i=k+1}^{t} W_r g'(h_{i-1})}$$

$$\frac{\partial h_i}{\partial h_{i-1}} \leq \|W_r\| \|g'(h_{i-1})\| \leq \gamma_{W_r} \gamma_g$$

$$\frac{\partial h_t}{\partial h_k} \leq \underbrace{(\gamma_{W_r} \gamma_g)}_{0 < \gamma < 1}{}^{t-k}$$

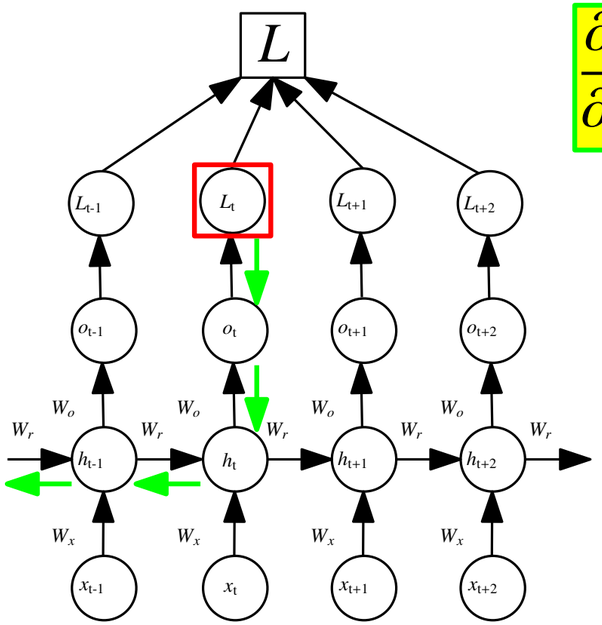
vanishing gradient $\Leftarrow$

Problem with training of RNNs

$$\frac{\partial L}{\partial W_r} = \sum_{t=1}^{s} \frac{\partial L_t}{\partial W_r}$$

$$h_t = W_x x_t + W_r g(h_{t-1})$$
$$o_t = W_o g(h_t)$$

$$\boxed{\frac{\partial L_t}{\partial W_r}} = \sum_{k=1}^{t} \boxed{\frac{\partial L_t}{\partial o_t}}\boxed{\frac{\partial o_t}{\partial h_t}}\boxed{\frac{\partial h_t}{\partial h_k}}\boxed{\frac{\partial h_k}{\partial w_r}}$$



$$\boxed{\frac{\partial h_t}{\partial h_k} = \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} = \prod_{i=k+1}^{t} W_r g'(h_{i-1})}$$

$$\frac{\partial h_i}{\partial h_{i-1}} \leq \|W_r\|\|g'(h_{i-1})\| \leq \gamma_{W_r}\gamma_g$$

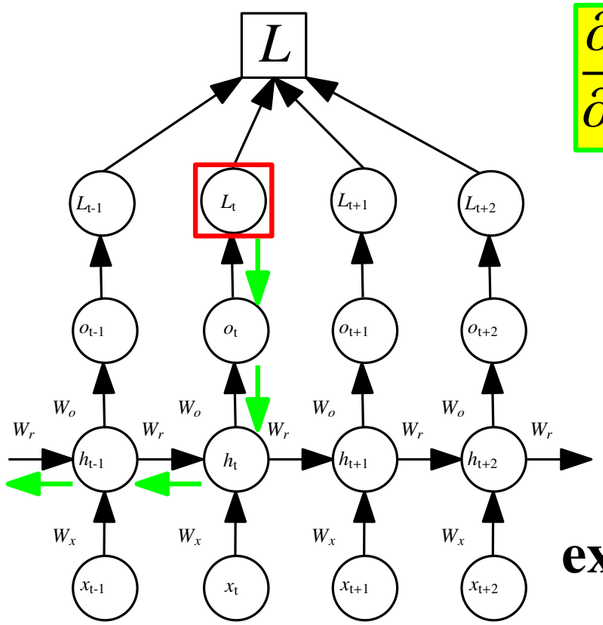
$$\frac{\partial h_t}{\partial h_k} \leq \underbrace{(\gamma_{W_r}\gamma_g)}_{\gamma > 1}{}^{t-k}$$

Problem with training of RNNs

$$\frac{\partial L}{\partial W_r} = \sum_{t=1}^{s} \frac{\partial L_t}{\partial W_r}$$

$$h_t = W_x x_t + W_r g(h_{t-1})$$
$$o_t = W_o g(h_t)$$

$$\boxed{\frac{\partial L_t}{\partial W_r}} = \sum_{k=1}^{t} \boxed{\frac{\partial L_t}{\partial o_t}}\boxed{\frac{\partial o_t}{\partial h_t}}\boxed{\frac{\partial h_t}{\partial h_k}}\boxed{\frac{\partial h_k}{\partial w_r}}$$



$$\boxed{\frac{\partial h_t}{\partial h_k} = \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} = \prod_{i=k+1}^{t} W_r g'(h_{i-1})}$$

$$\frac{\partial h_i}{\partial h_{i-1}} \leq \|W_r\| \|g'(h_{i-1})\| \leq \gamma_{W_r} \gamma_g$$

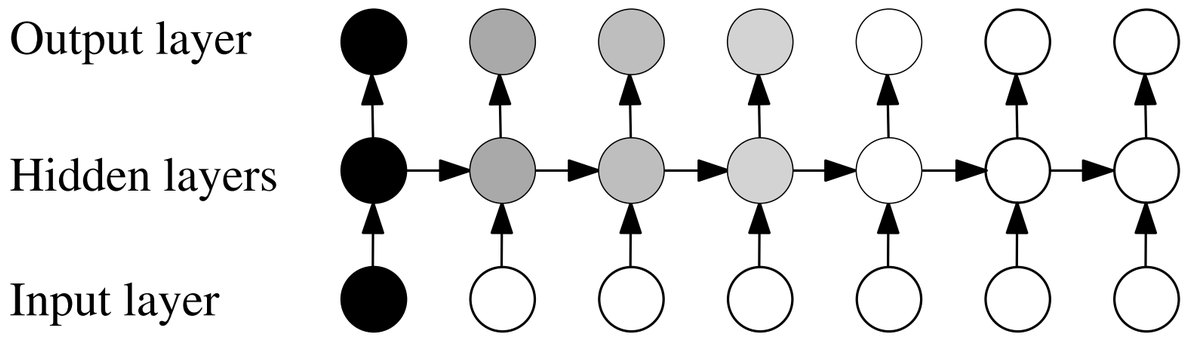
$$\frac{\partial h_t}{\partial h_k} \leq \underbrace{(\gamma_{W_r} \gamma_g)}_{\gamma > 1}{}^{t-k}$$

exploding gradient $\Leftarrow$

Vanishing gradient

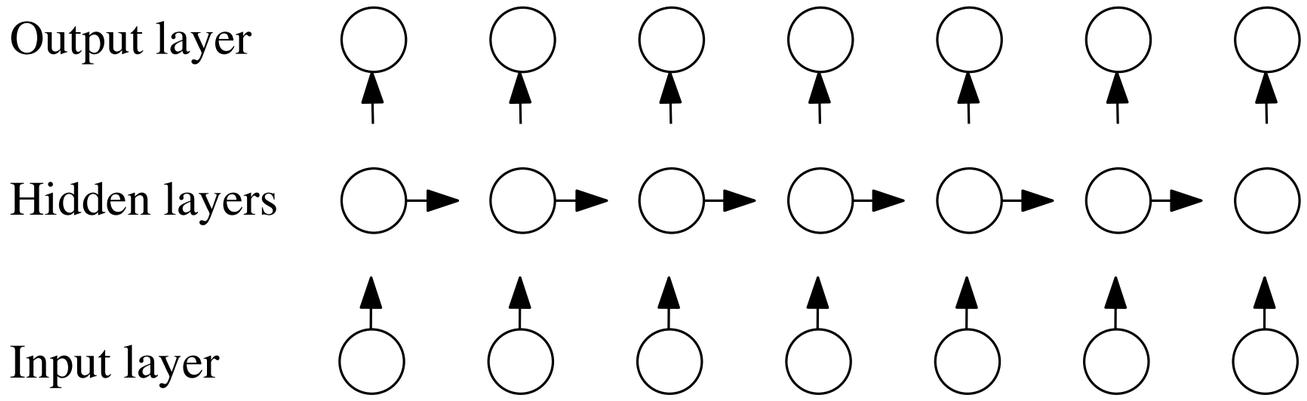
New inputs override activation of hidden layers

⇒ Network forgets first inputs



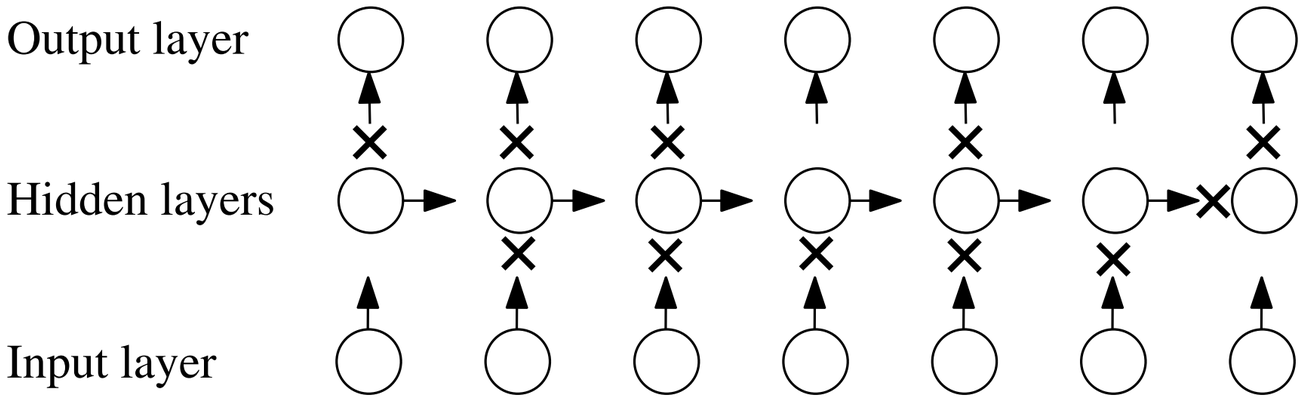
Poorly mappable long-term memory
⇒ Patterns with long dependencies not learnable

Vanishing gradient



Vanishing gradient

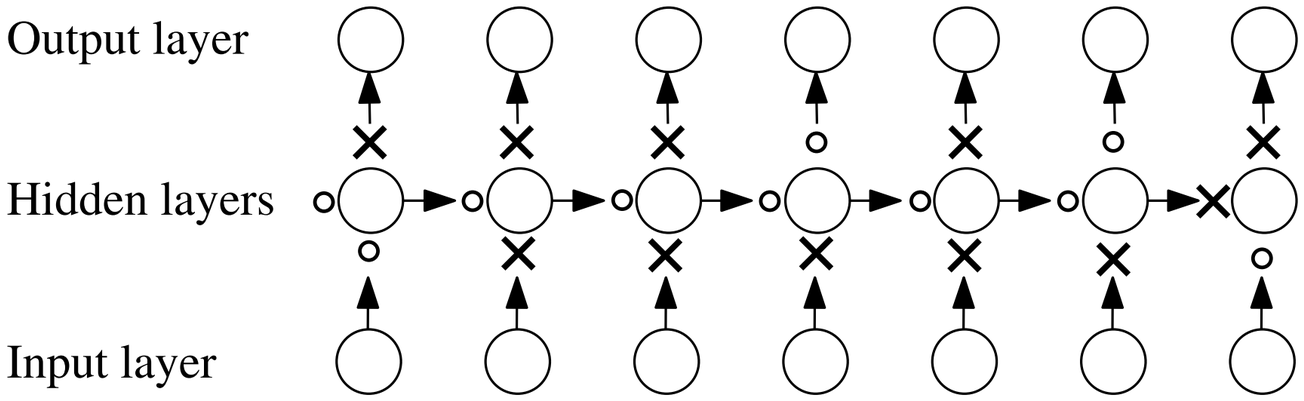
✕ Forget gate



Vanishing gradient

× Forget gate

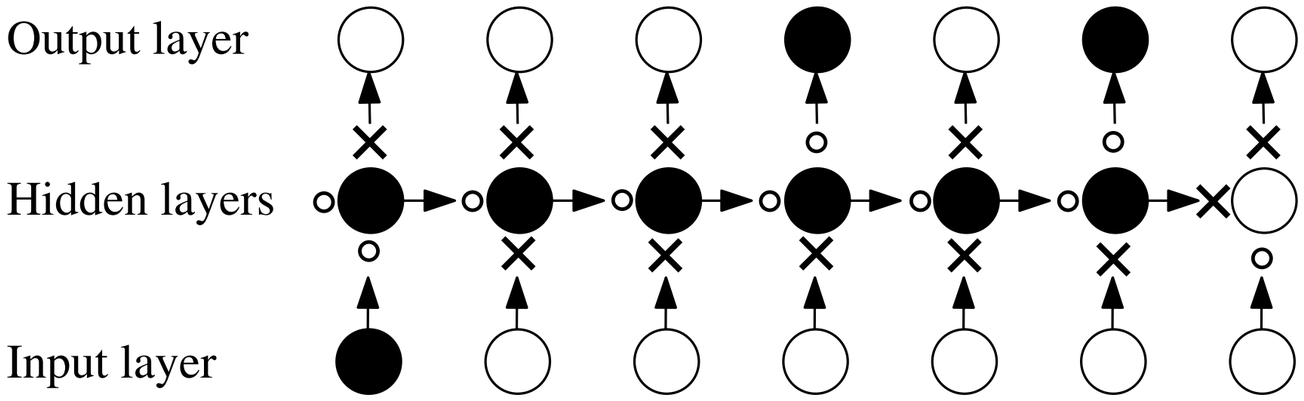
○ Open gate



Vanishing gradient

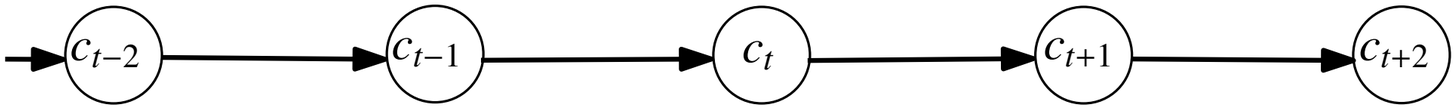
× Forget gate

○ Open gate



Long Short Term Memory (LSTM)

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780

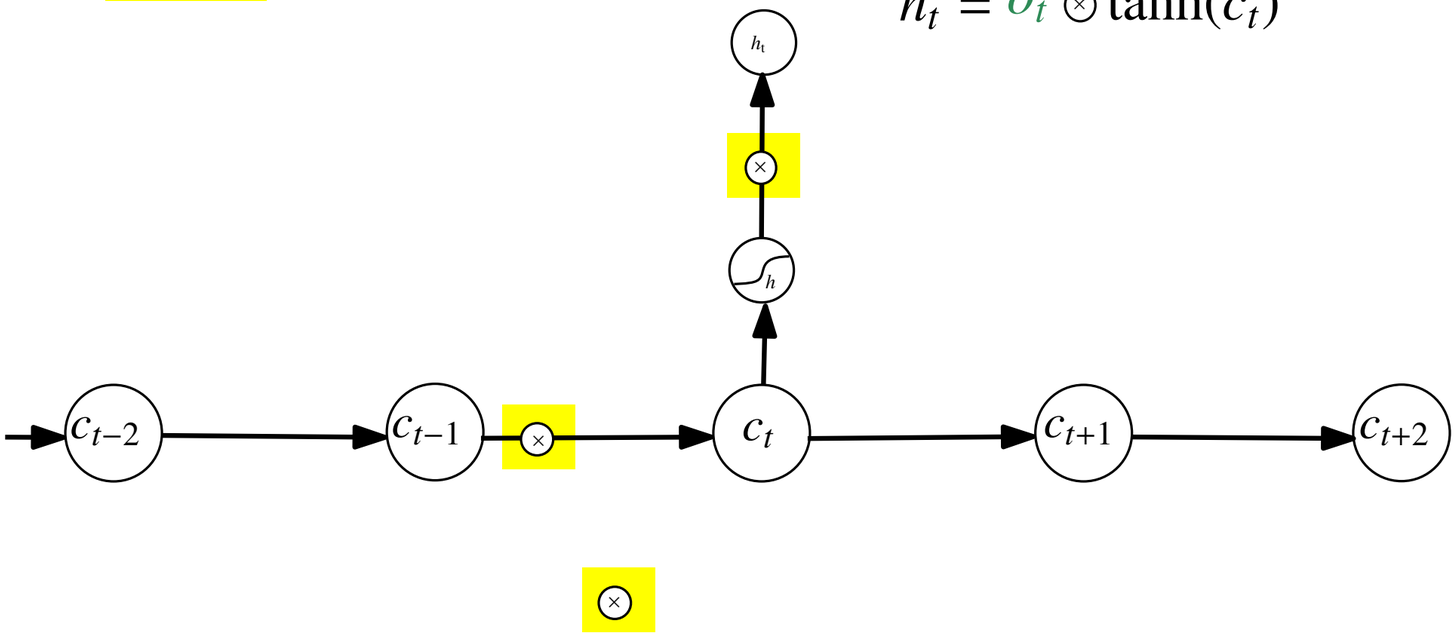


Long Short Term Memory (LSTM)

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780

$\odot$ Gates

$$h_t = o_t \odot \tanh(c_t)$$



Long Short Term Memory (LSTM)

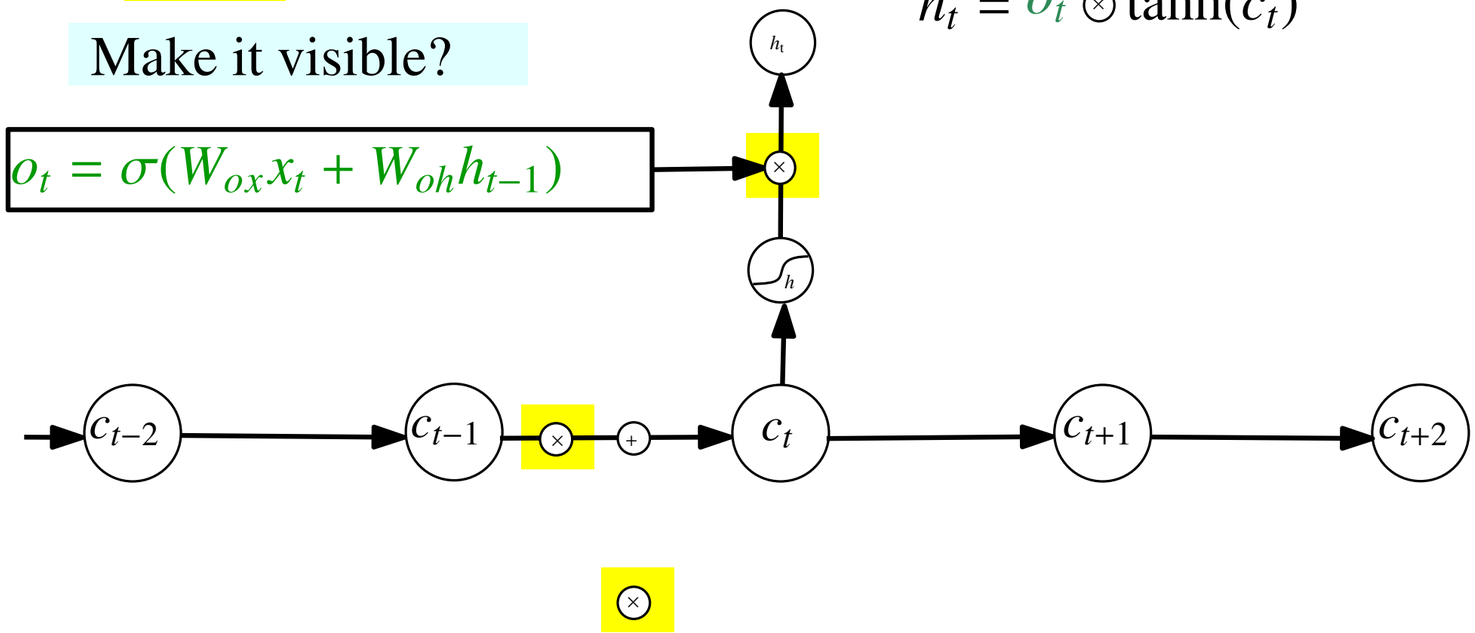
Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780

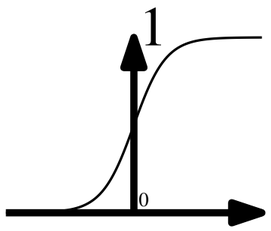
$\otimes$ Gates

Make it visible?

$$h_t = o_t \otimes \tanh(c_t)$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1})$$





$\sigma(x) = \frac{1}{1+e^{-x}}$

Long Short Term Memory (LSTM)

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780

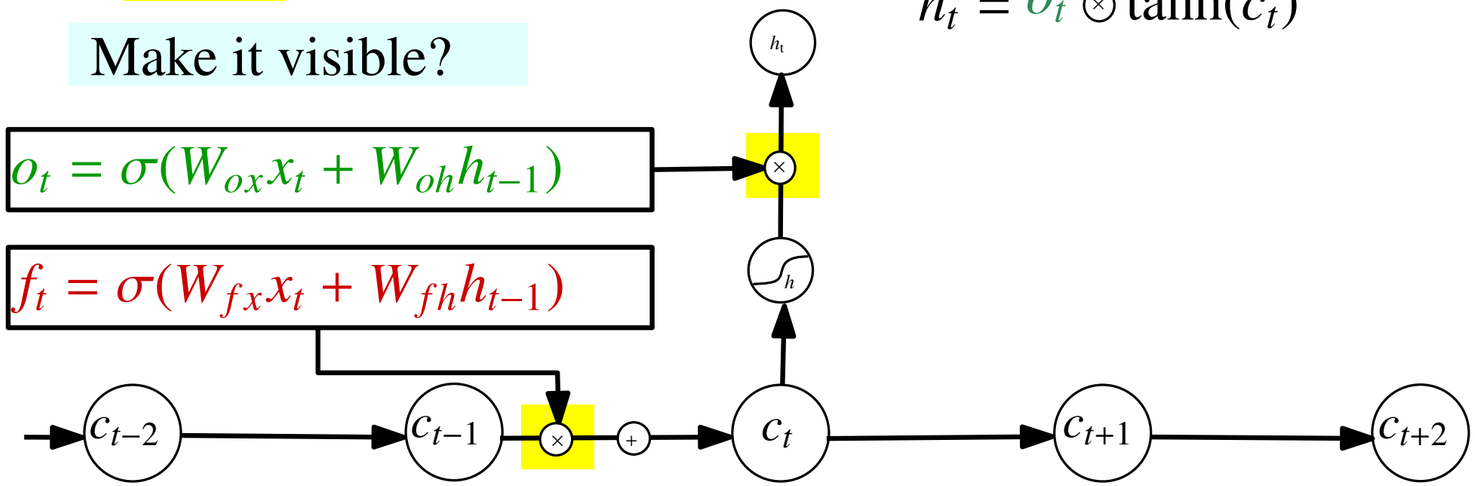
$\otimes$ Gates

Make it visible?

$$h_t = o_t \otimes \tanh(c_t)$$

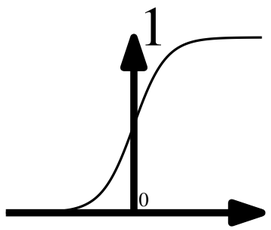
$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1})$$

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1})$$



Forget or store?

$\otimes$



$\sigma(x) = \frac{1}{1+e^{-x}}$

Long Short Term Memory (LSTM)

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780

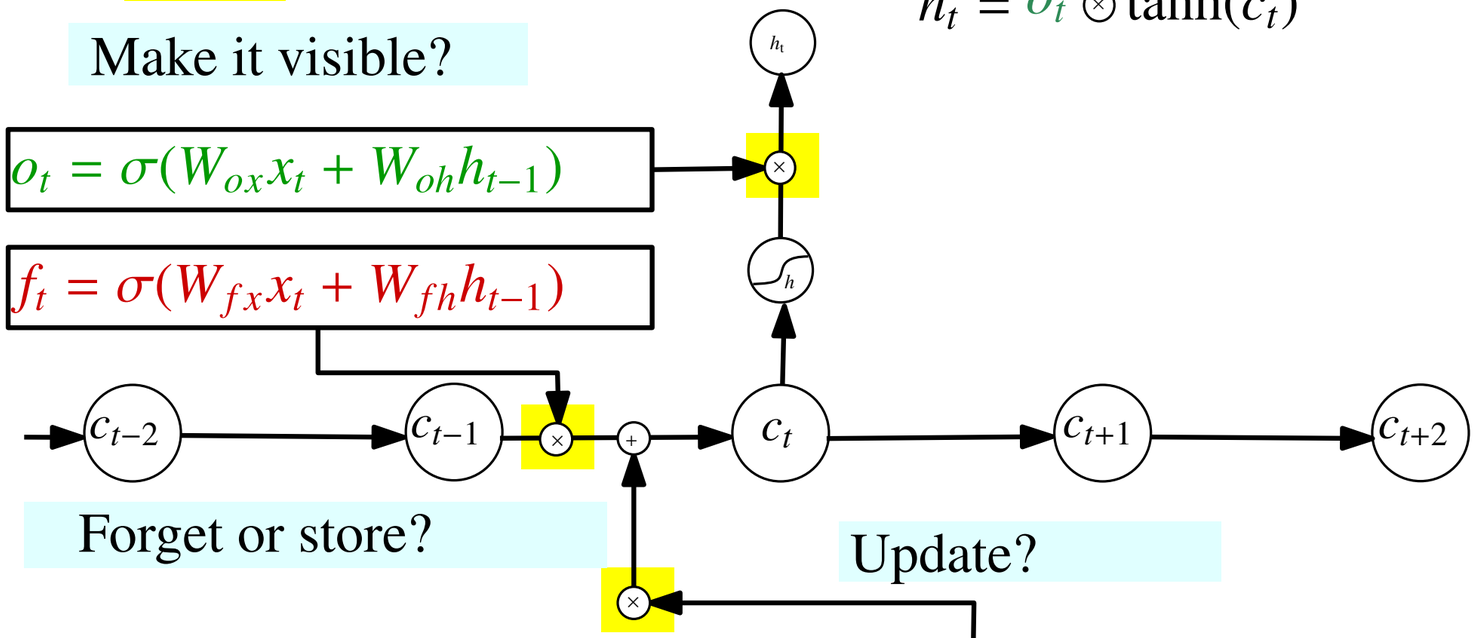
$\otimes$ Gates

Make it visible?

$$h_t = o_t \otimes \tanh(c_t)$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1})$$

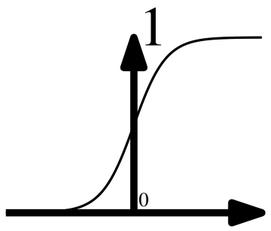
$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1})$$



Forget or store?

Update?

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1})$$



$\sigma(x) = \frac{1}{1+e^{-x}}$

Long Short Term Memory (LSTM)

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780

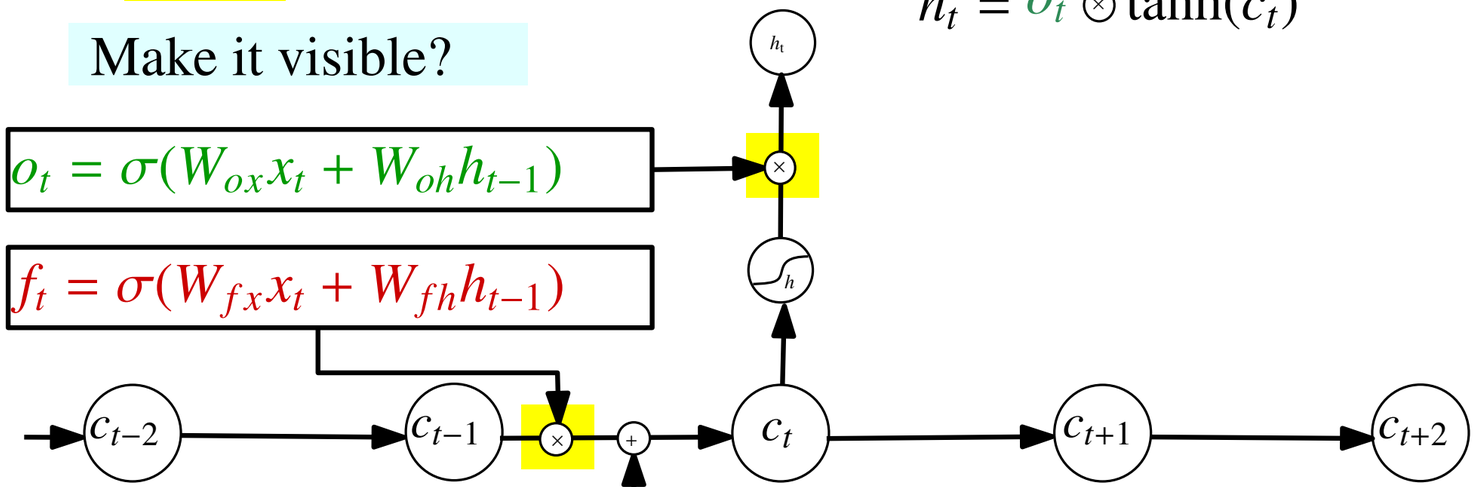
$\otimes$ Gates

Make it visible?

$$h_t = o_t \otimes \tanh(c_t)$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1})$$

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1})$$

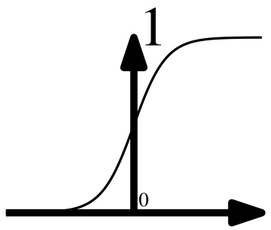


Forget or store?

Update?

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1})$$

$$\hat{c}_t = \tanh(W_{cx}x_t + W_{ch}h_{t-1})$$



$\sigma(x) = \frac{1}{1+e^{-x}}$

Long Short Term Memory (LSTM)

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780

$\otimes$ Gates

Make it visible?

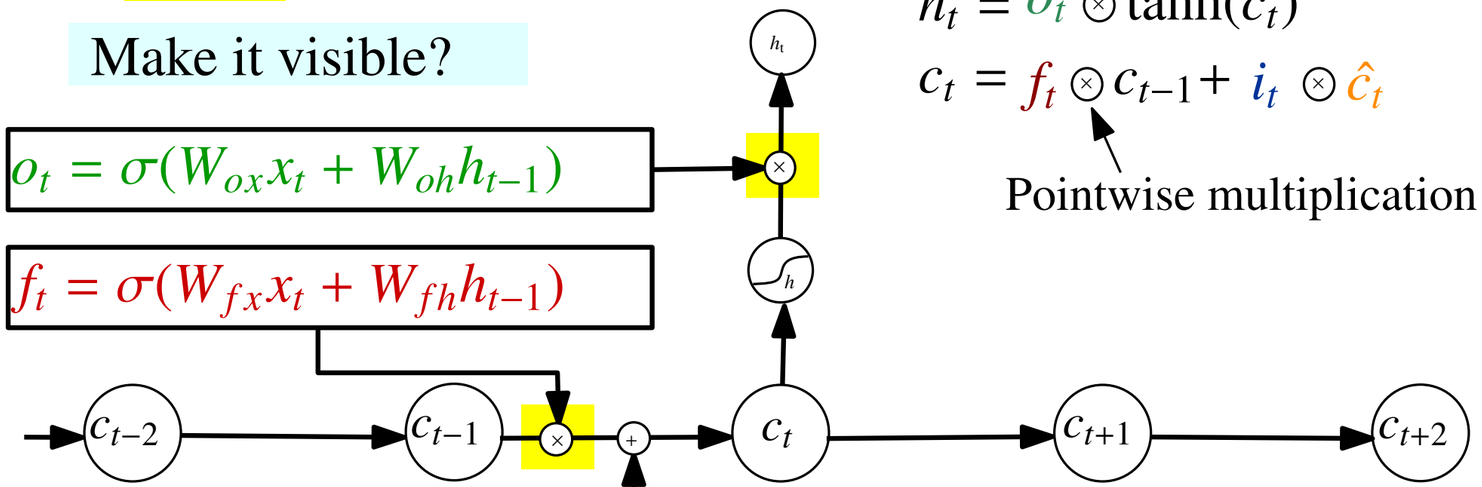
$$h_t = o_t \otimes \tanh(c_t)$$

$$c_t = f_t \otimes c_{t-1} + i_t \otimes \hat{c}_t$$

Pointwise multiplication

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1})$$

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1})$$

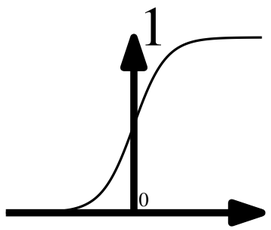


Forget or store?

Update?

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1})$$

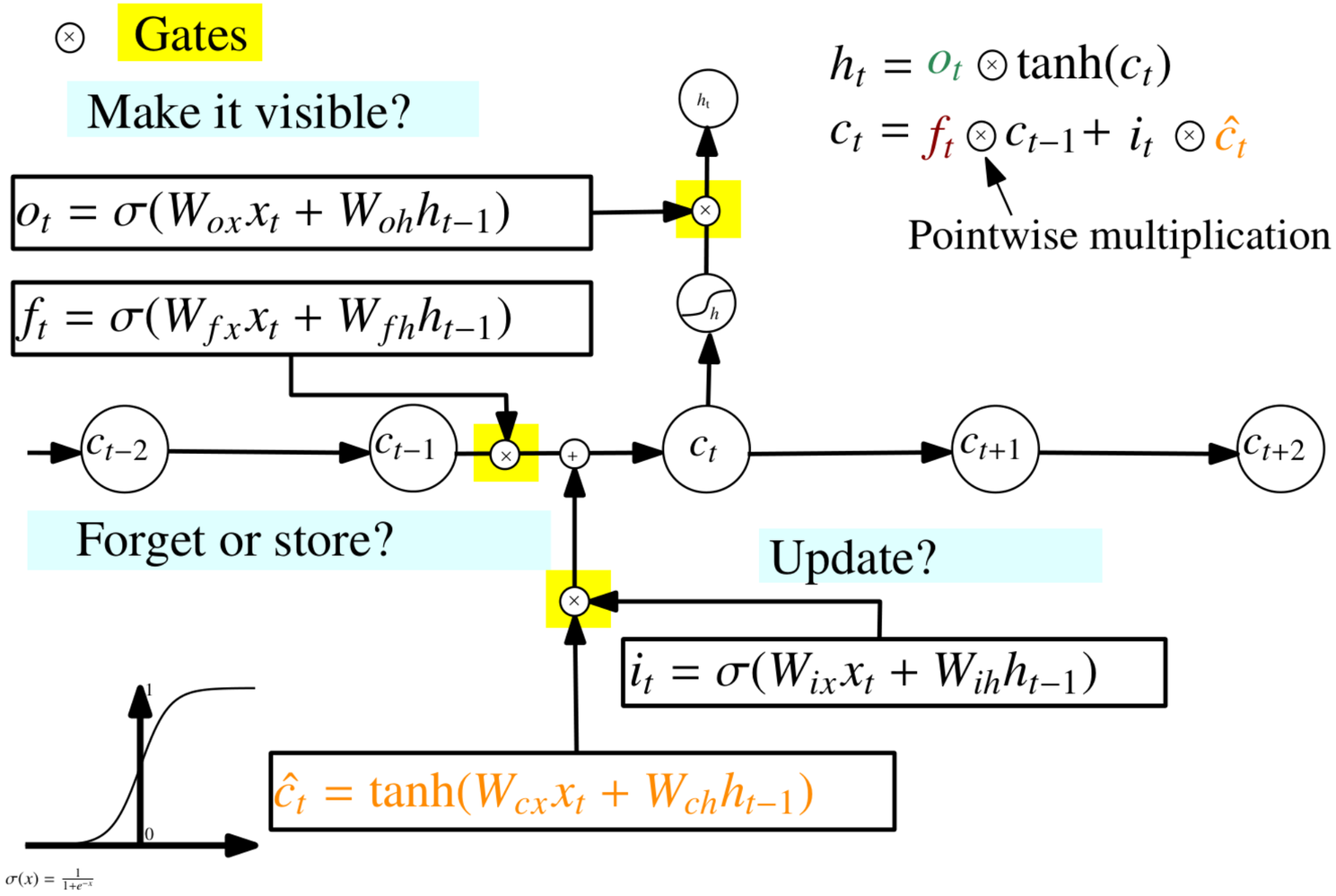
$$\hat{c}_t = \tanh(W_{cx}x_t + W_{ch}h_{t-1})$$



$\sigma(x) = \frac{1}{1+e^{-x}}$

Long Short Term Memory (LSTM)

LSTM cell as simple RNN cell

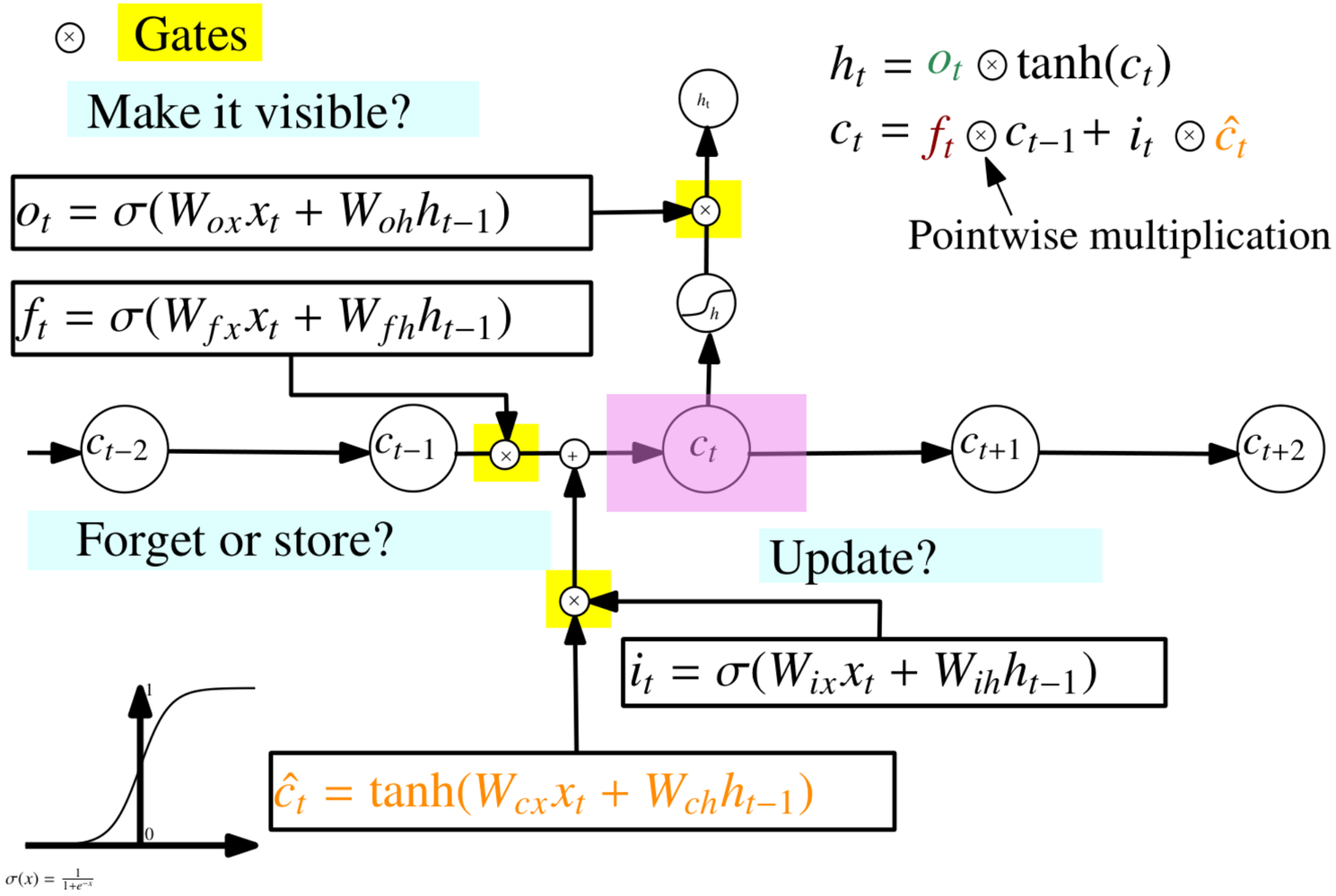


Long Short Term Memory (LSTM)

LSTM cell as simple RNN cell

Further components

- Cell state
- Gates
  - Forget gate
  - Input gate
  - Output gate



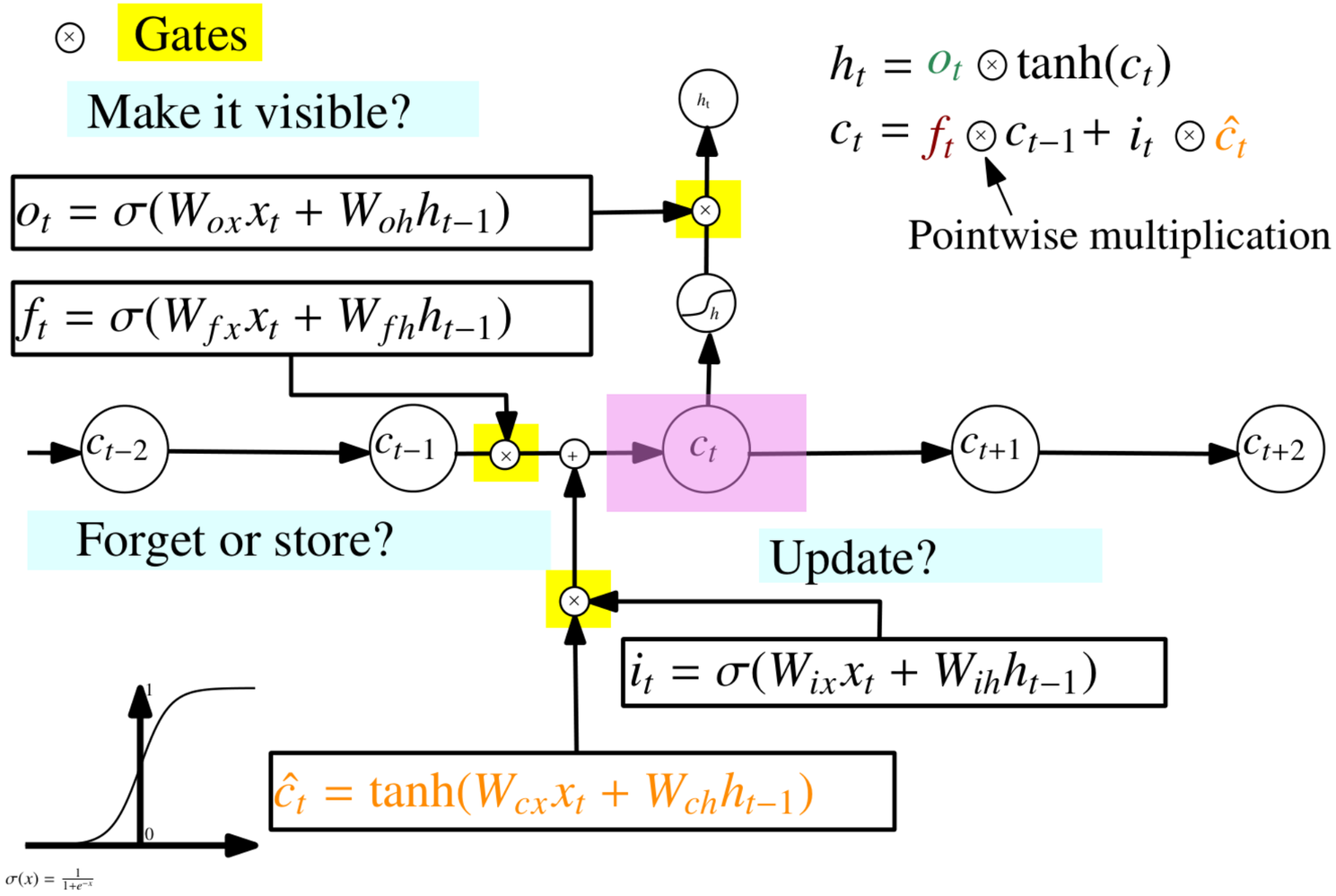
Long Short Term Memory (LSTM)

LSTM cell as simple RNN cell

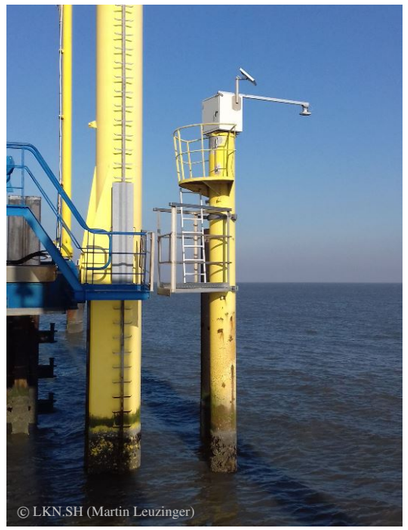
Further components

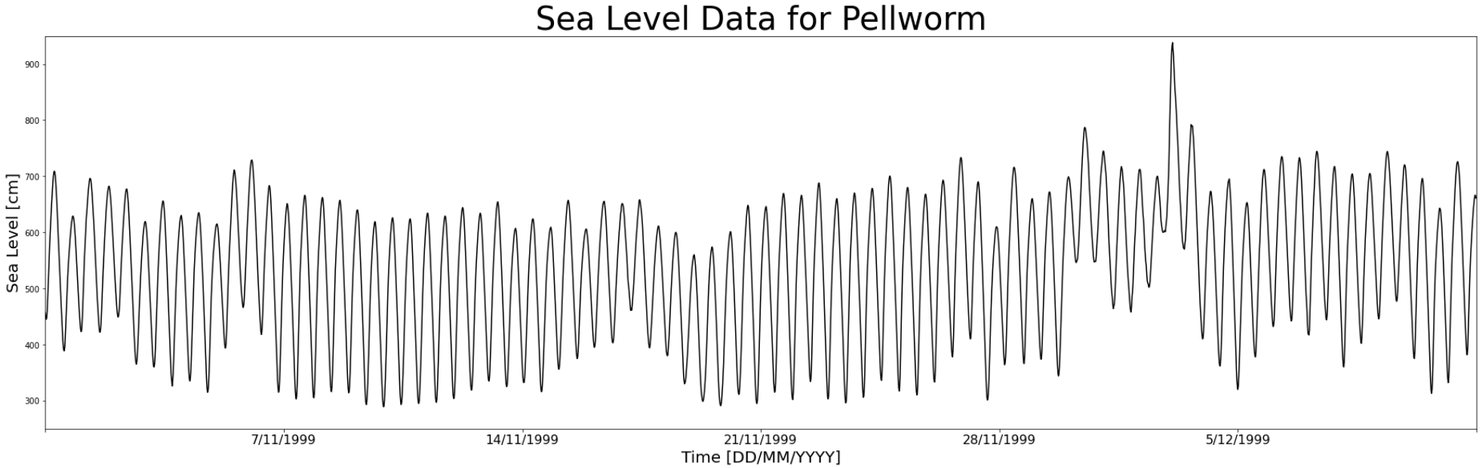
- Cell state
- Gates
  - Forget gate
  - Input gate
  - Output gate

⇒ Gates as filter

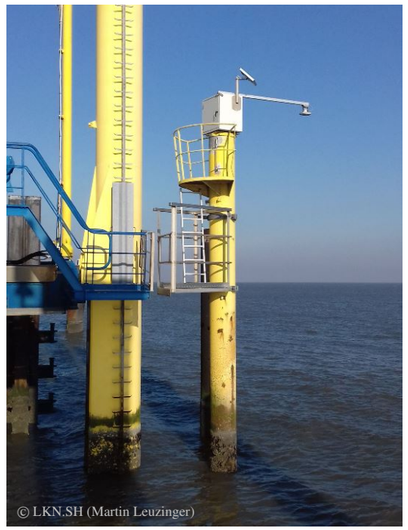


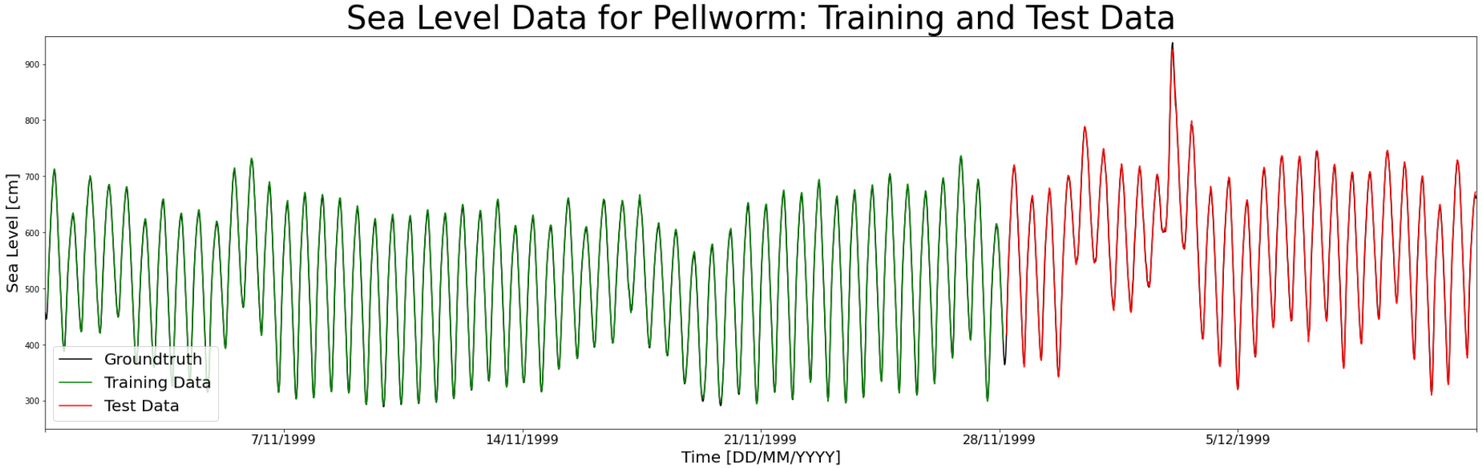
Time-series data





Time-series data





Time-series data

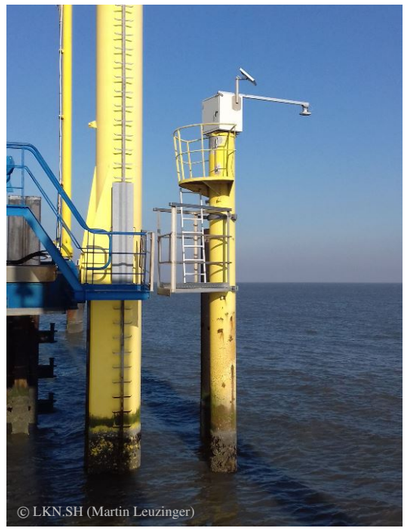
Samstag, 04.12.1999

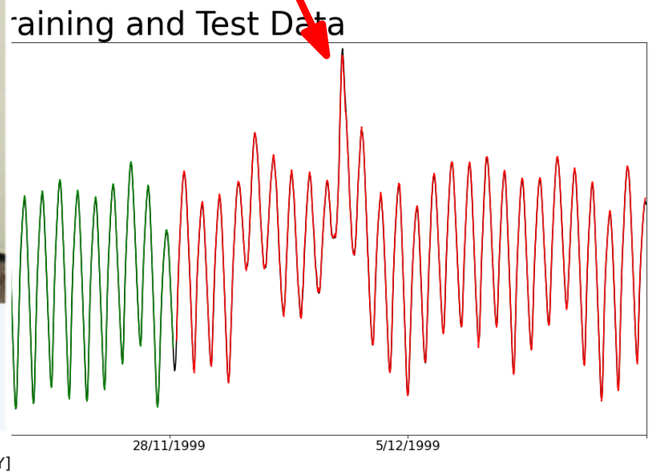
Milliardenschäden nach ANATOL

Orkan fordert mehrere Todesopfer

Orkan ANATOL ist am 3. und 4. Dezember 1999 über Nordeuropa und Norddeutschland gefegt. Mehrere Menschen kamen ums Leben. In Dänemark galt er als schlimmster Orkan des 20. Jahrhunderts und verursachte Milliardenschäden.

ANATOL zerstört auf Sylt einzelne Häuser, wie hier in Kampen. In List wurden Windgeschwindigkeiten von Tempo 180 gemessen. Auf der Nachbarinsel Föhr werden hunderte Bäume umgeweht. Bild: dpa





Summary

- Deep neural network needs to be designed based on application
  - Convolutional neural network suitable to capture spatial information
  - Recurrent neural network suitable to capture temporal information
- Different networks can be combined (Conv-LSTM)