

# Deep Learning: Sequential Models

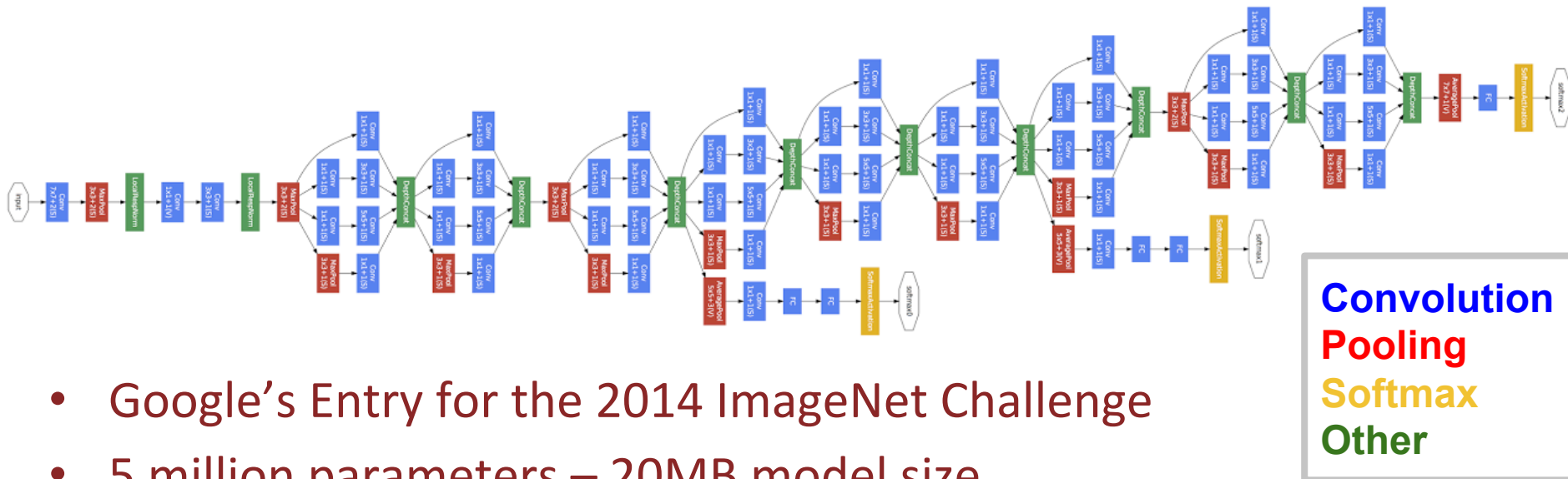
Luciano Barbosa

(Part of the material from Deep Learning for NLP by Nils Reimers)

# Motivation for Deep Learning

- In theory, feed forward networks with a single hidden layer can compute any function: no need for deep architectures
- However, learning shallow architectures is not always efficient
- Some problems require an exponential number of hidden units

# What does a Deep Network can look like?



# Requirements for Training

- Deal with billions of operations
  - Google had trained some models on up to 16000 cores
- Fast: performance in training time is **crucial**
- Nearly all operations are matrix operations (multiplications, additions)
  - Optimizing matrix multiplication for speed is hard
- Run on multiple CPUs and on GPUs

# Convolutional Neural Networks - CNNs

- Limitations of fully-connected layers
  - Don't take into account proximity
  - Input has fixed size
- But for text
  - Context is very important
  - Input has variable size
- Take into account proximity by learning local patterns in the text
- Deal with inputs of variable size
- Three basic ideas
  - Local filters
  - Shared weights
  - Pooling

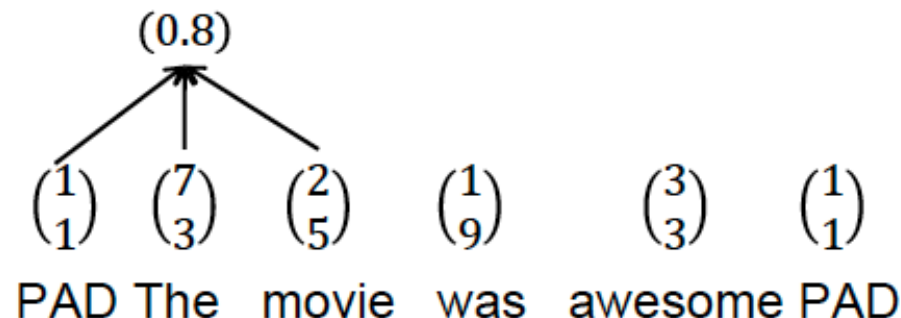
# Single Layer CNN – Local Filter

- 1D filter
- Windows size = 3

Word Vectors:  $w_i \in \mathbb{R}^2$   
 Weight Matrix:  $W \in \mathbb{R}^{1 \times 6}$   
 Bias:  $b \in \mathbb{R}$

} Local  
Filter

$$\text{output} = \tanh \left( W \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} + b \right)$$



# Single Layer CNN – Local Filter

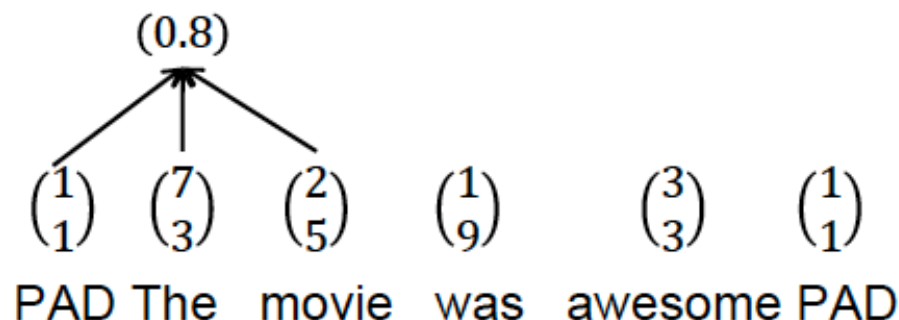
- 1D filter
- Windows size = 3

Word Vectors:  $w_i \in \mathbb{R}^2$   
 Weight Matrix:  $W \in \mathbb{R}^{1 \times 6}$   
 Bias:  $b \in \mathbb{R}$

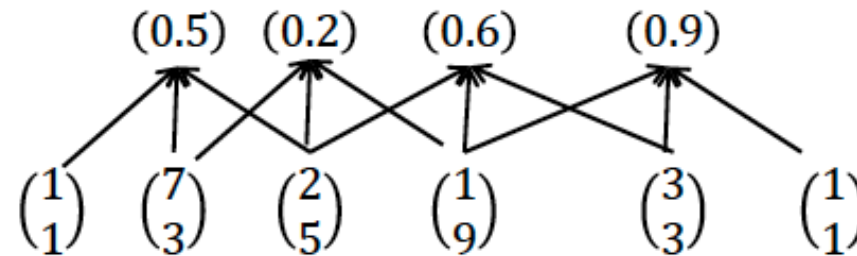
Local Filter

$$\text{than} \begin{bmatrix} \text{---} W \text{---} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 7 \\ 3 \\ 2 \\ 5 \end{bmatrix} + b = 0.8$$

$$\text{output} = \tanh \left( W \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} + b \right)$$



# Single Layer CNN – Shared Weights



PAD The movie was awesome PAD

- For each window use the same weight and bias values (shared weights)
  - Reduce the number of parameters
- This gives us the same number of outputs as the length of the sentence (variable size)
- How to build a fixed size vector?



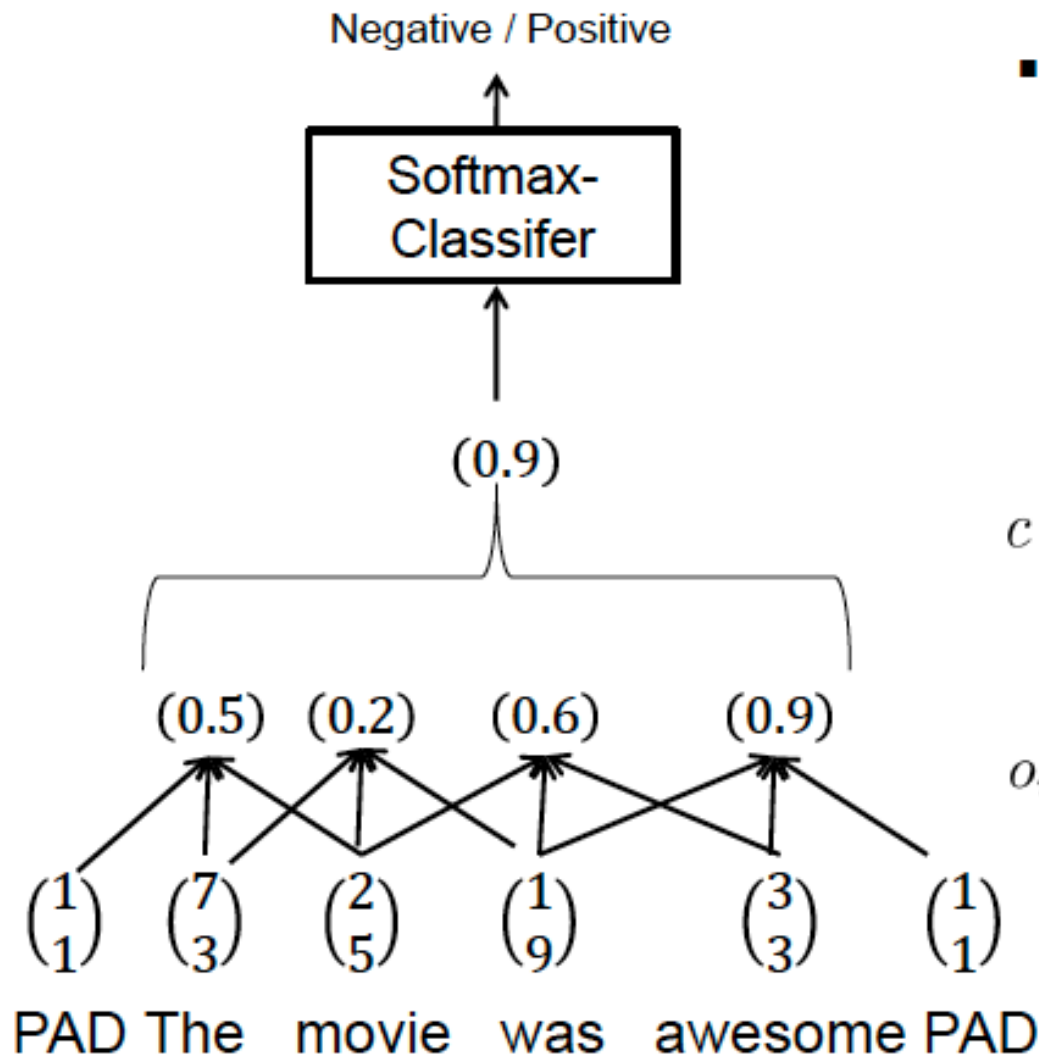
# Pooling Layer

- Idea: to capture the most important activation
- Example: max-pooling layer
- Let  $o_1, o_2, \dots \in \mathbb{R}$  denote the output values of the filter
- Compute a max pooling layer:

$$c = \max_i(o_i) \in \mathbb{R}$$

- Deals with fixed-size output: dimension of  $o_j$
- Max-pooling most common in NLP. In Computer Vision, min-pooling and mean-pooling also common.

# Max Pooling

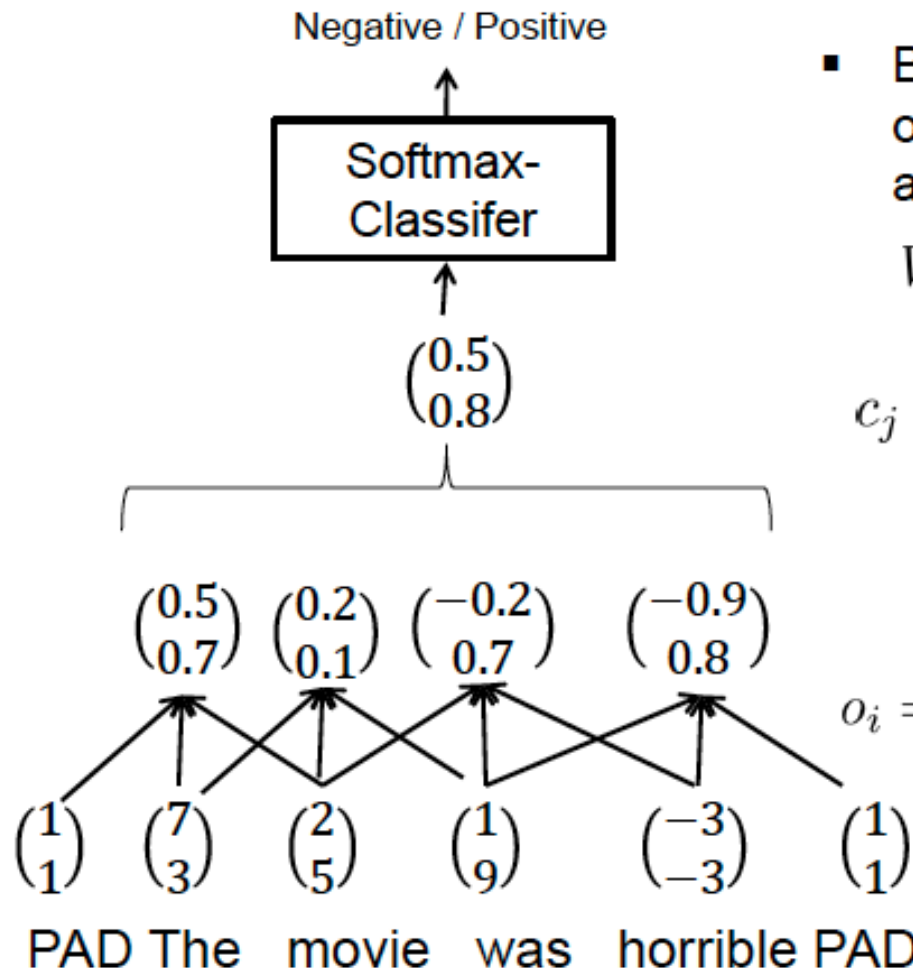


- You can train this like any other Neural Network

$$c = \max_i(o_i)$$

$$o_i = \tanh \left( W \begin{pmatrix} w_{i-1} \\ w_i \\ w_{i+1} \end{pmatrix} + b \right)$$

# Increasing Number of Filters: K=2



- By changing the dimensionality of the weight matrix, we can add further filters:

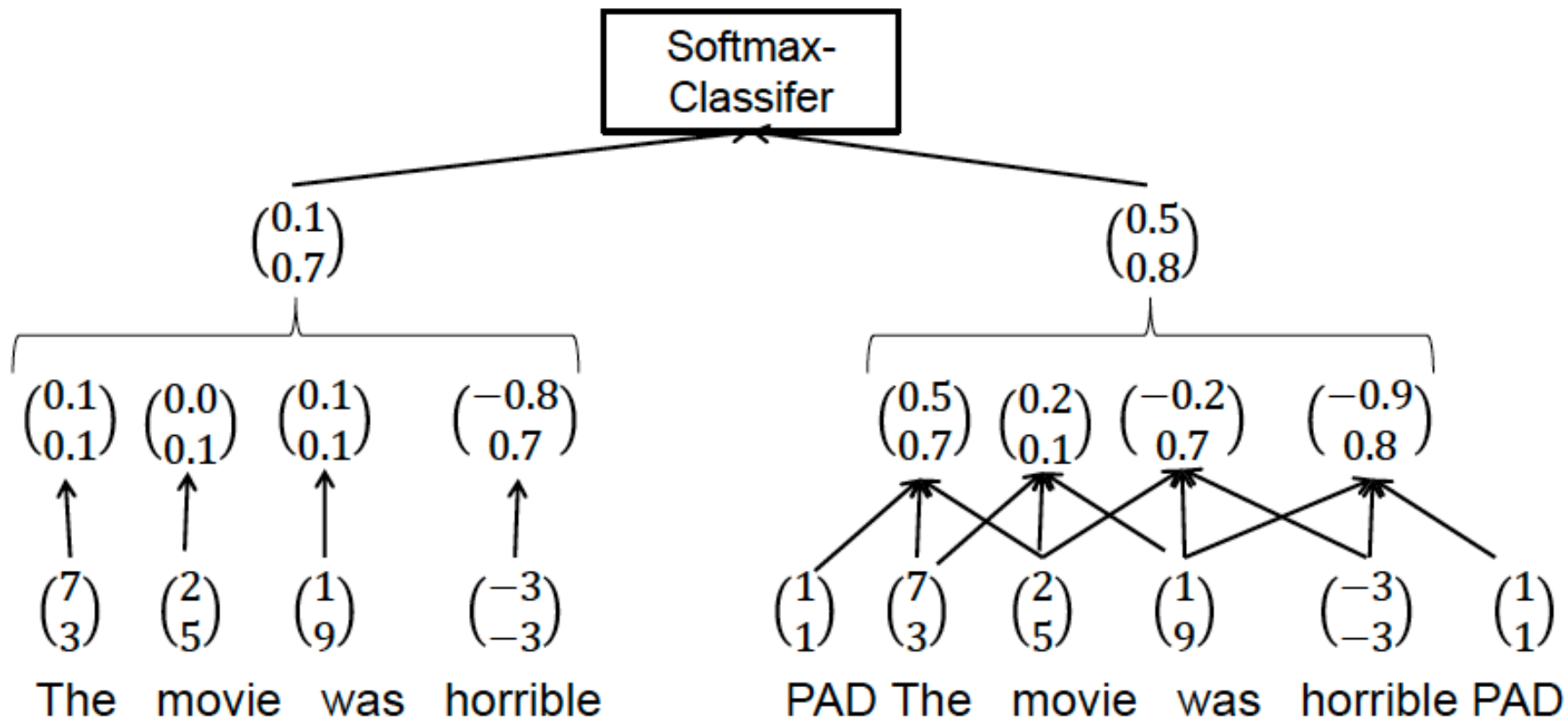
$$W \in \mathbb{R}^{k \times 6}$$

$$c_j = \max_i(o_{i,j}) \text{ for } 0 < j < k$$

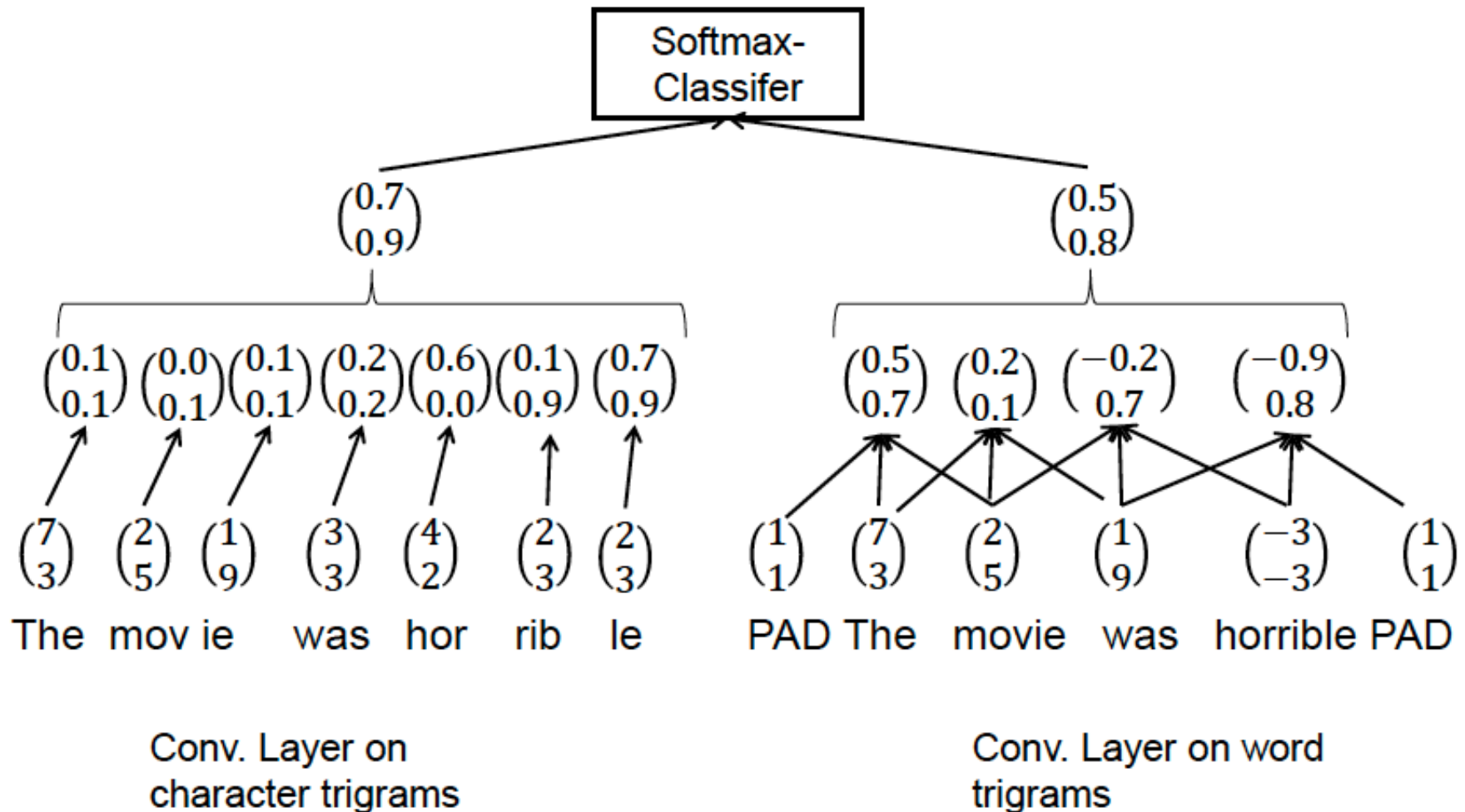
$$o_i = \tanh \left( W \begin{pmatrix} w_{i-1} \\ w_i \\ w_{i+1} \end{pmatrix} + b \right) \in \mathbb{R}^k$$

# Increasing Windows Size

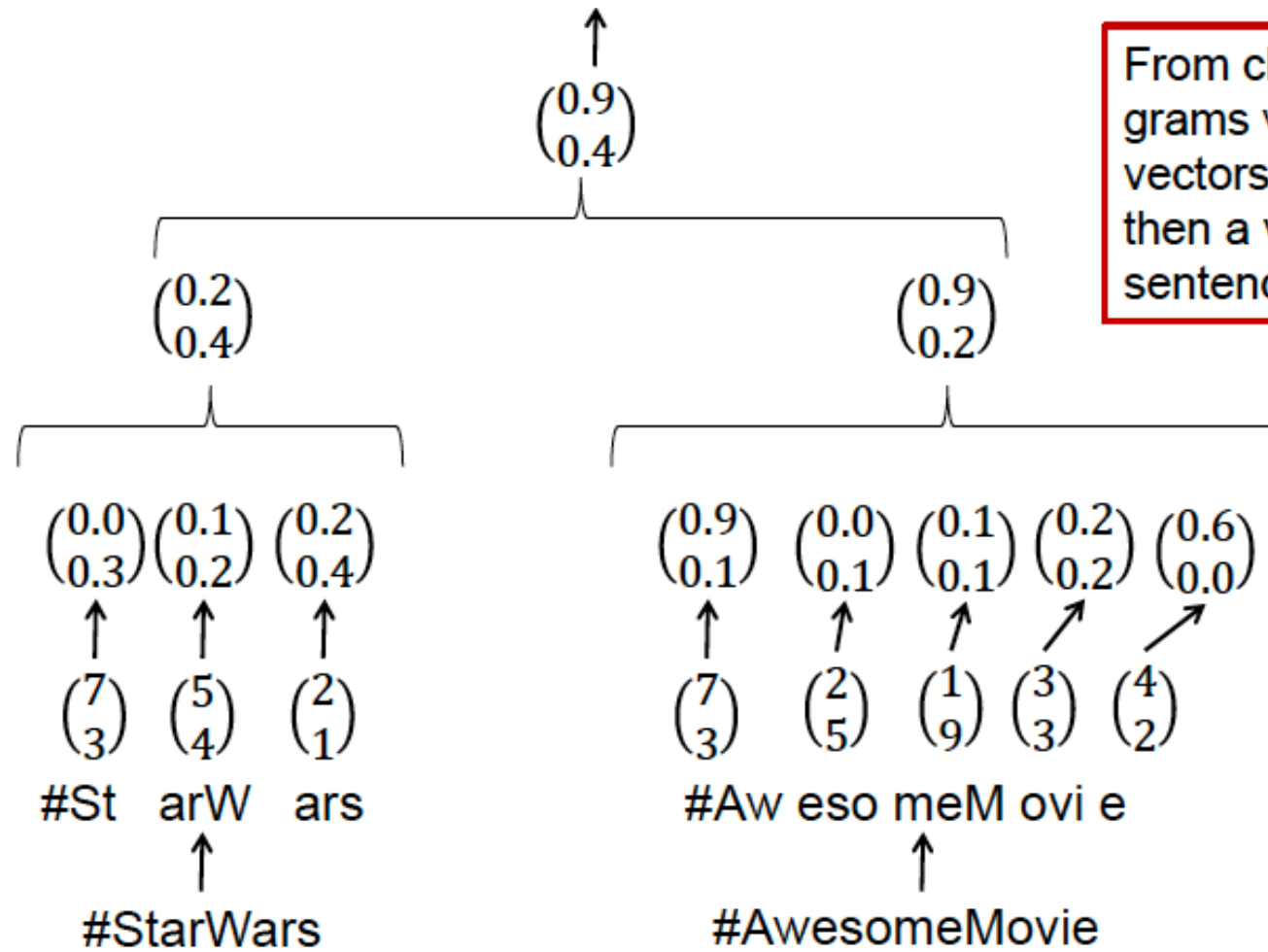
- Combining output of unigrams and trigrams



# Different granularity



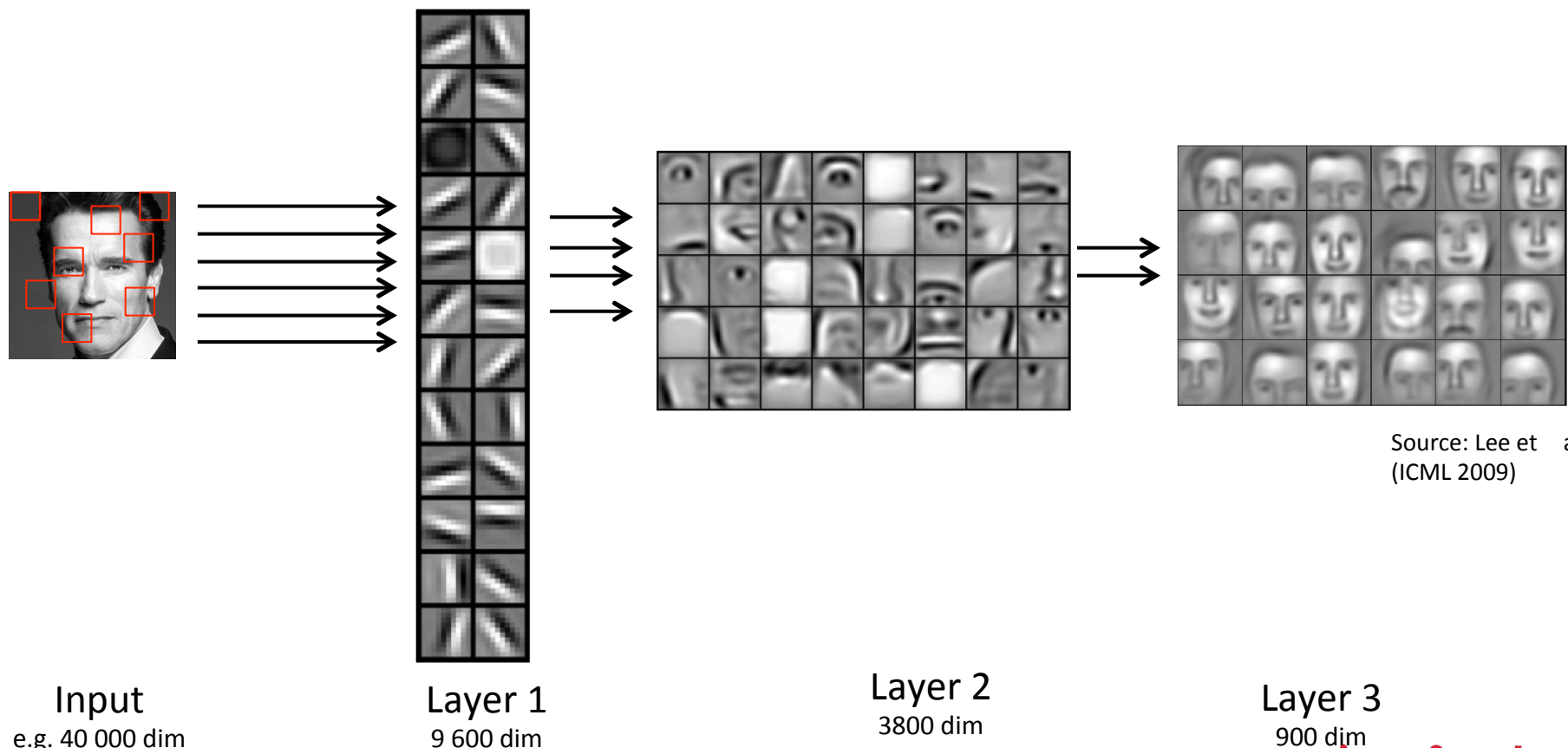
# Stacking Convolutional Layers



From character n-grams we could derive vectors for words and then a vector for the sentence

# Stacked Convolutional Layers

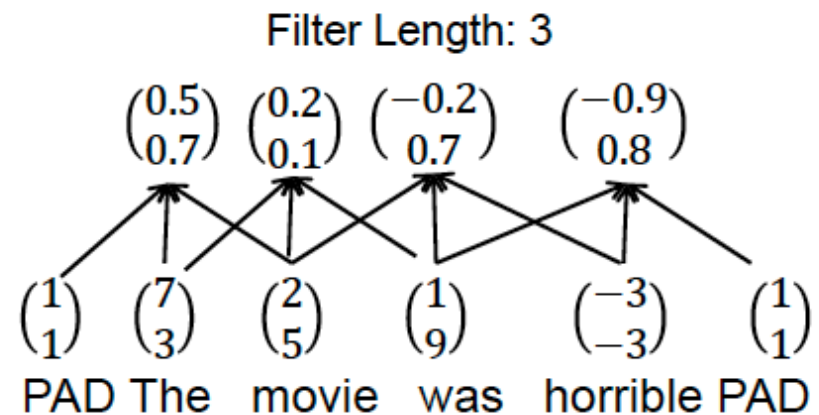
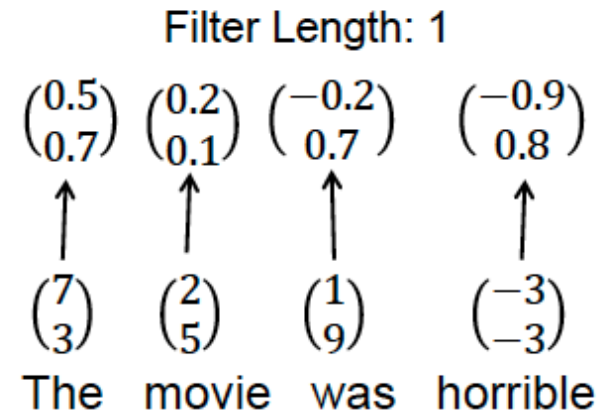
- Computer vision uses stacked convolutional layers to derive from simple representations high level representations



Source: Lee et al.  
(ICML 2009)

# Terminology: Filter Length

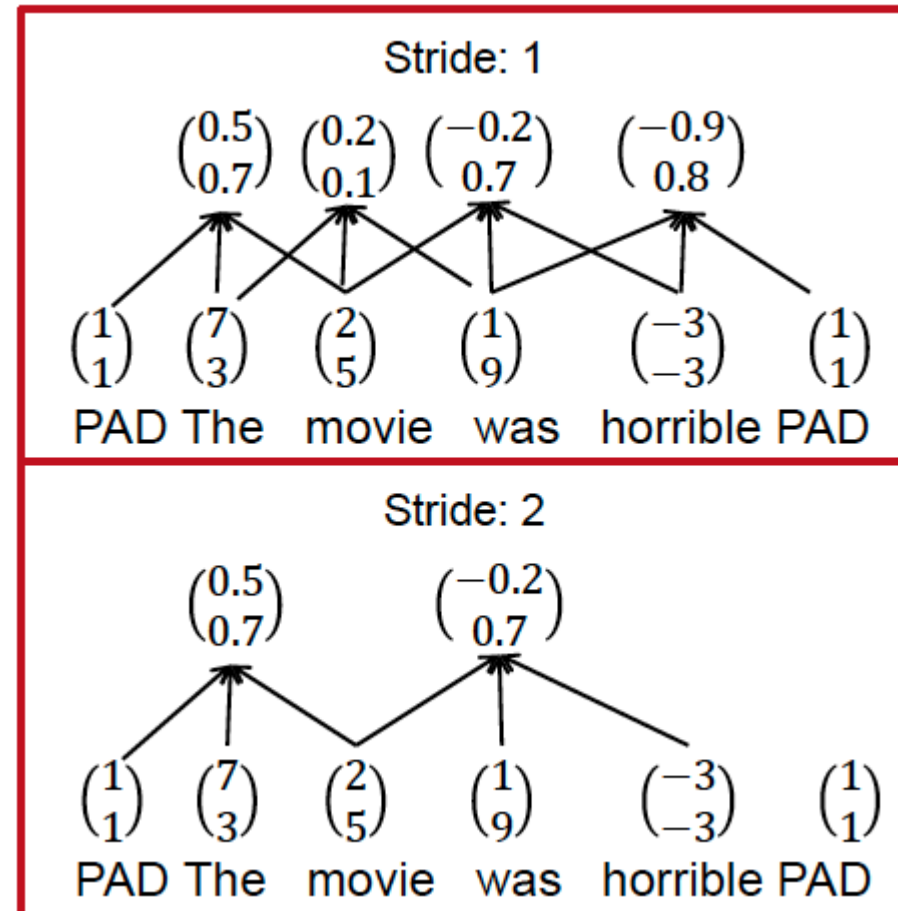
- The filter length is the extension of each filter
- Context window of size  $n$





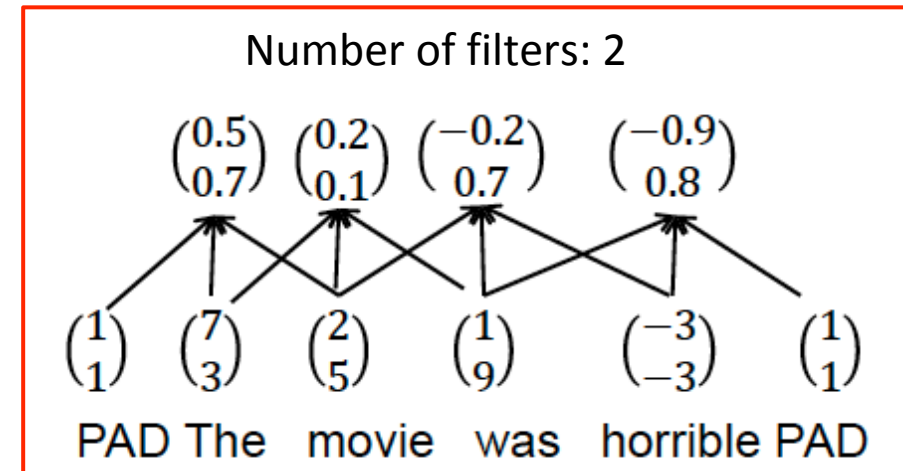
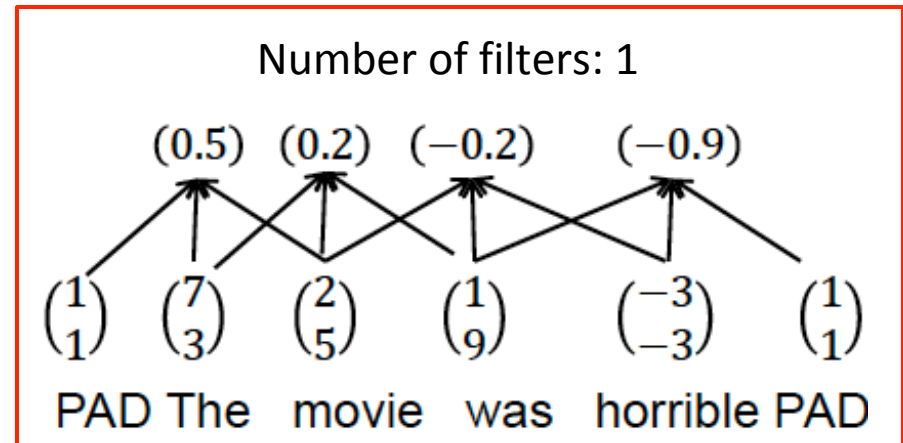
# Terminology: Stride

- The *stride* specifies the step size we move across a sentence
- In NLP: typically stride=1
- In computer vision: Other values can be used



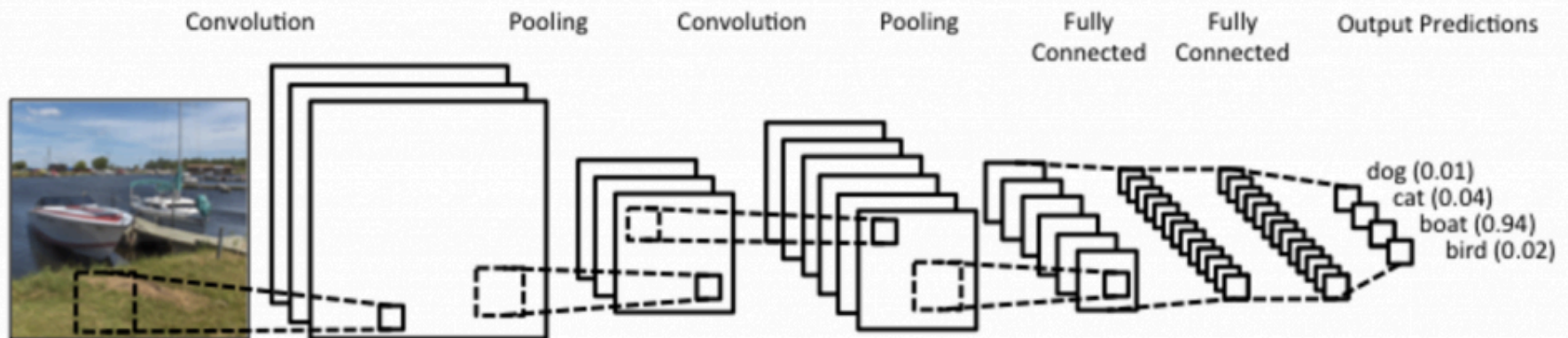
# Terminology: Number of Filters

- Size of the CNN's output vector



# Applications of CNNs

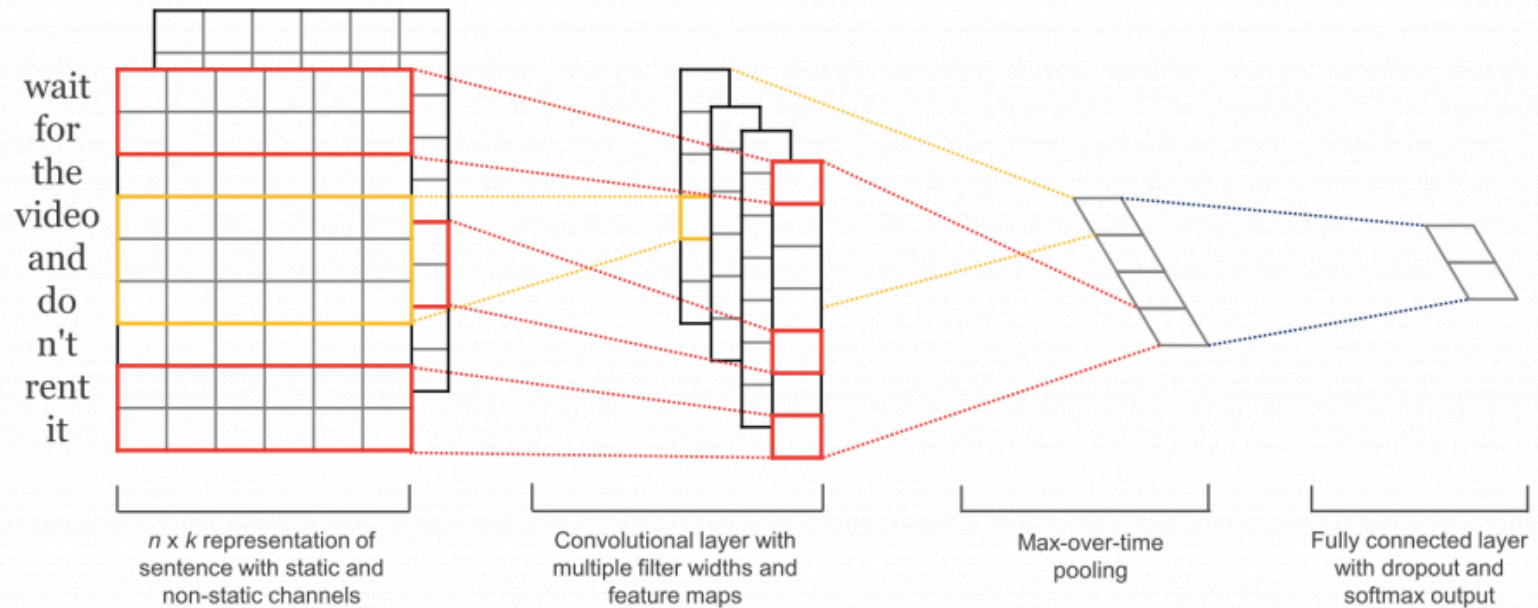
- Object detection



Img-Source: <http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/>

# Applications of CNNs

- Sentiment analysis



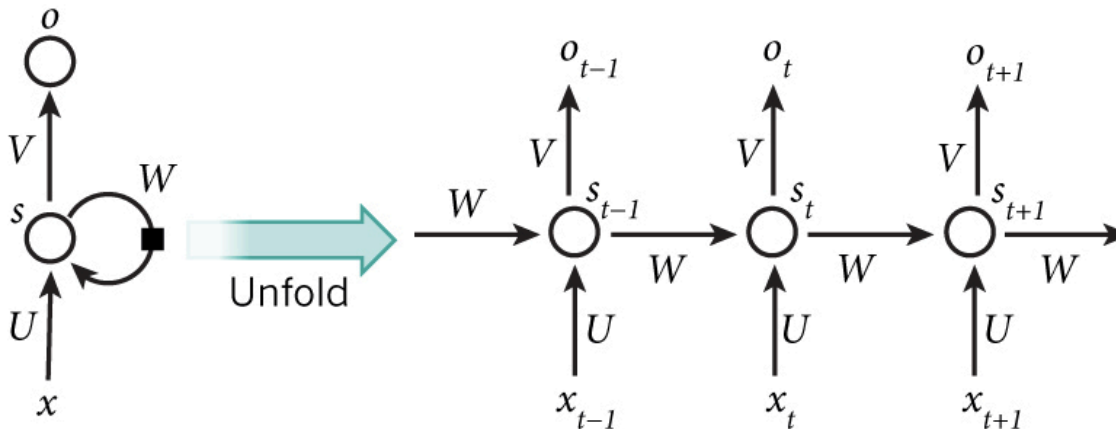
Img-Source: <https://arxiv.org/abs/1408.5882>

# Applications of CNNs

- Video analysis
- Web search
- Drug discovery
- Playing go

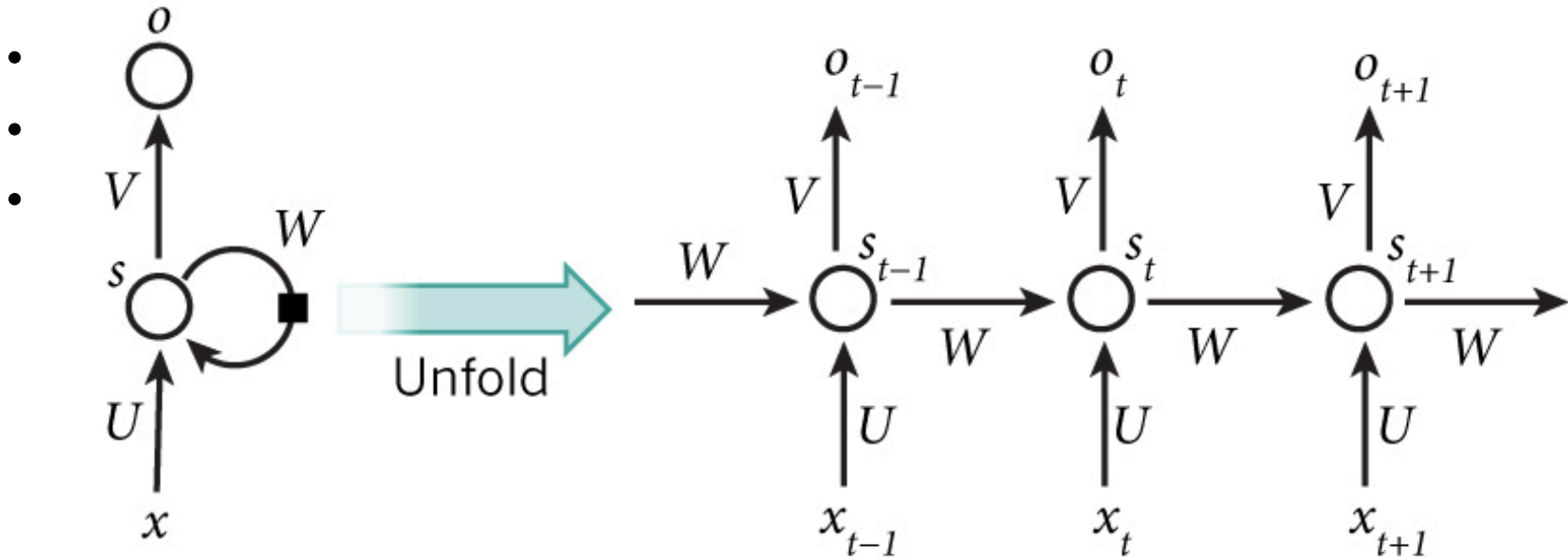
# Recurrent Neural Networks

- Traditional NNs: don't take into account proximity
- Used for sequential information



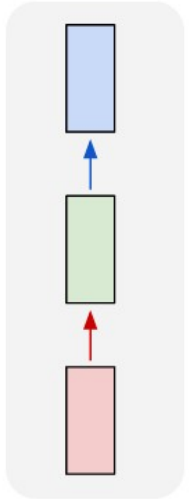
- Perform the same task of every element of a sequence
- Save the memory of what has been calculated
- Shared weights: share the same parameters across all steps

# Unfolded RNN

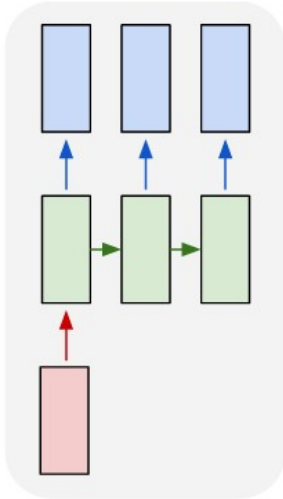


# Topologies of RNNs

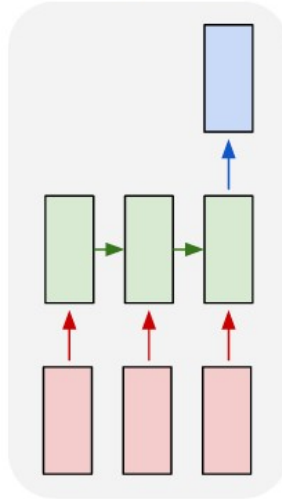
one to one



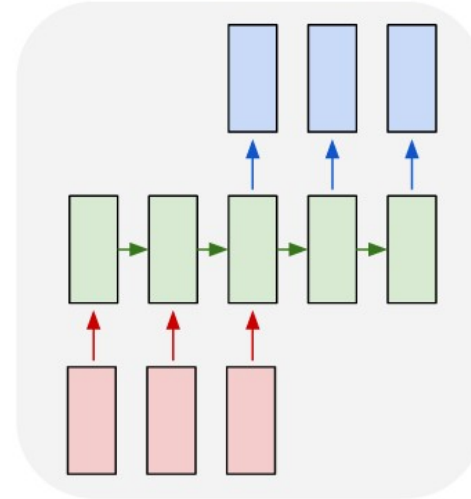
one to many



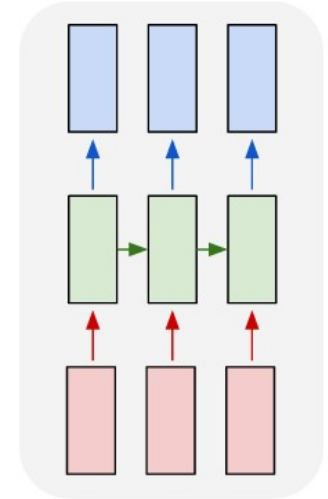
many to one



many to many



many to many



- 1) Fixed-size input and output: common Neural Network (e.g. feed forward network)
- 2) Sequence output: image captioning
- 3) Sequence input: sentiment classification
- 4) Sequence input and output: machine translation
- 5) Synced sequence input and output: frame classification of a video

Img Source: <http://karpathy.github.io/2015/05/21/rnn-effectiveness/>



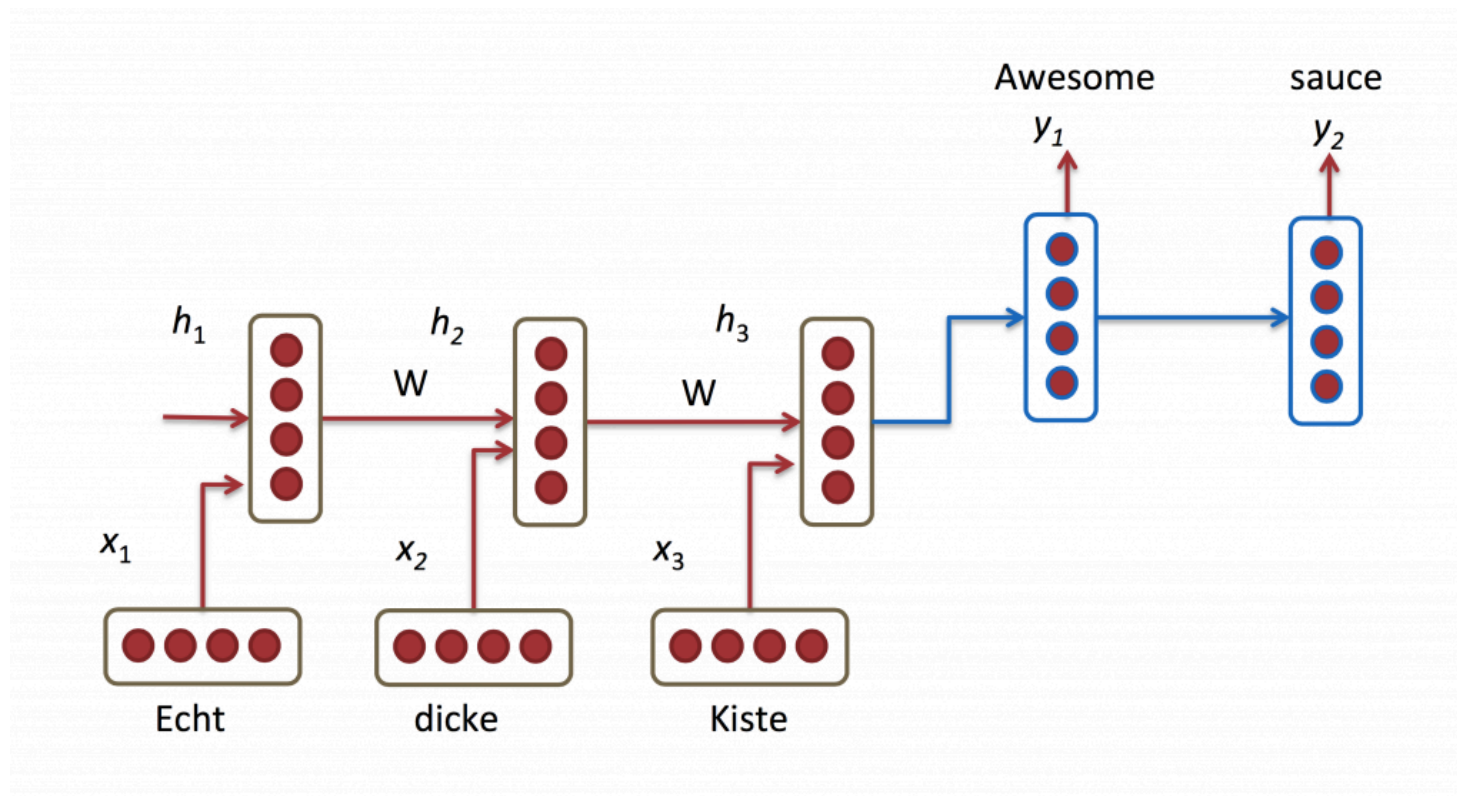
# Applications of RNN

- Language modeling and text generation
  - Example: trained from essays in economic topics

*"The surprised in investors weren't going to raise money. I'm not the company with the time there are all interesting quickly, don't have to get off the same programmers. There's a super-angel round fundraising, why do you can do. If you have a different physical investment are become in people who reduced in a startup with the way to argument the acquirer could see them just that you're also the founders will part of users' affords that and an alternation to the idea. [2] Don't work at first member to see the way kids will seem in advance of a bad successful startup. And if you have to act the big company too."*

# Applications of RNN

- Machine translation



# Applications of RNN



"man in black shirt is playing guitar."



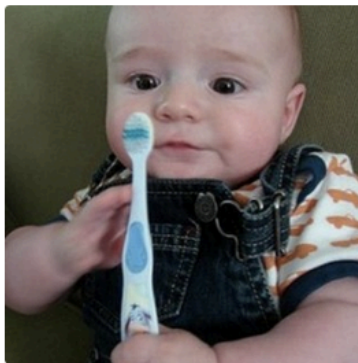
"construction worker in orange safety vest is working on road."



"two young girls are playing with lego toy."



"boy is doing backflip on wakeboard."



"a young boy is holding a baseball bat."



"a cat is sitting on a couch with a remote control."

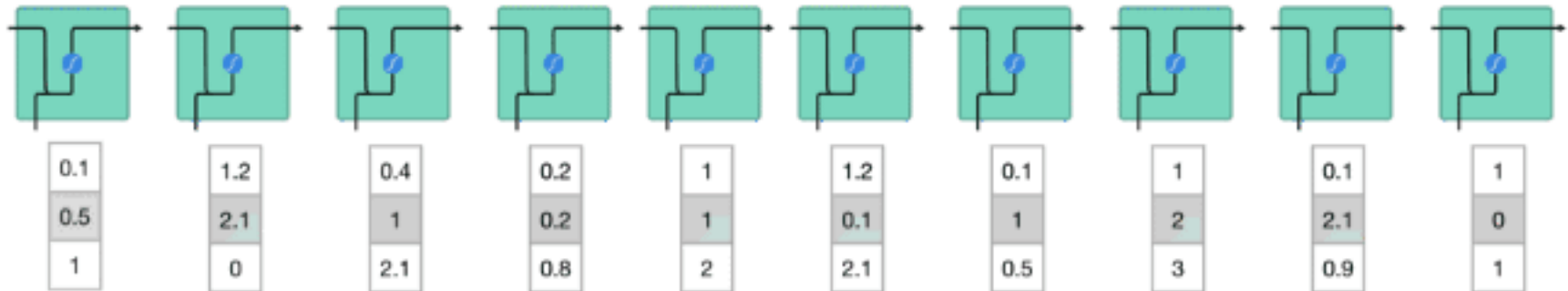


"a woman holding a teddy bear in front of a mirror."

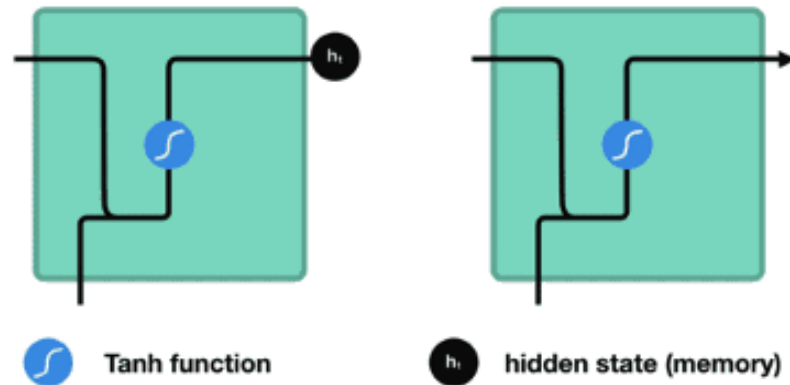


"a horse is standing in the middle of a road."

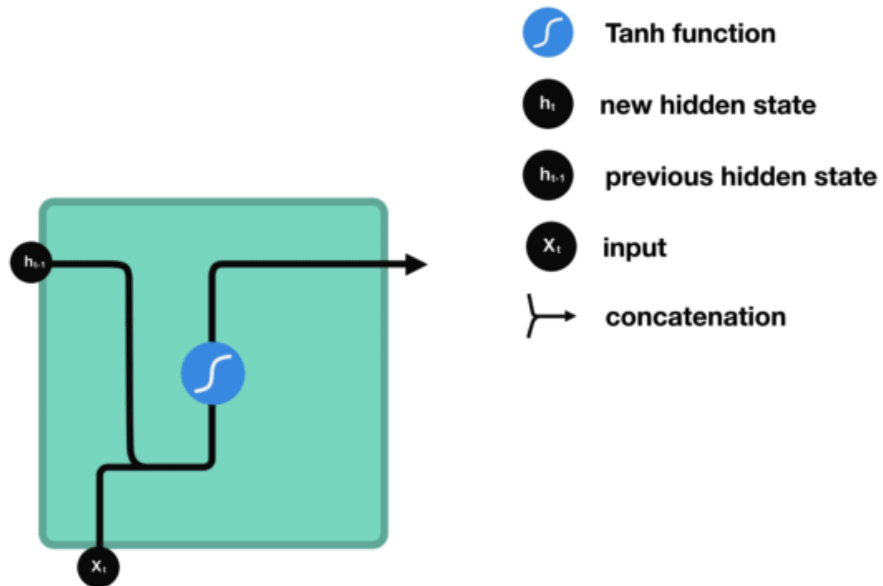
# RNN



# RNN

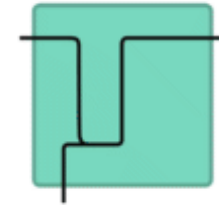
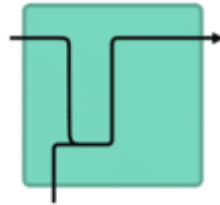
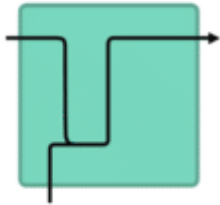


# RNN Cell



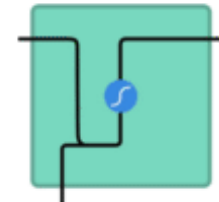
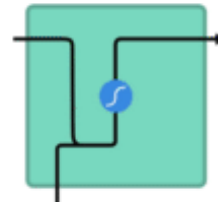
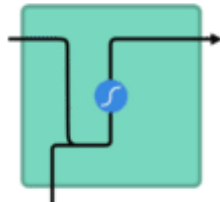
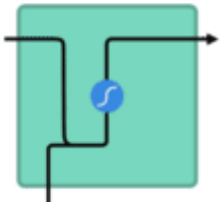
# RNN: Why Use Tanh

5
0.01
-0.5



Without tanh

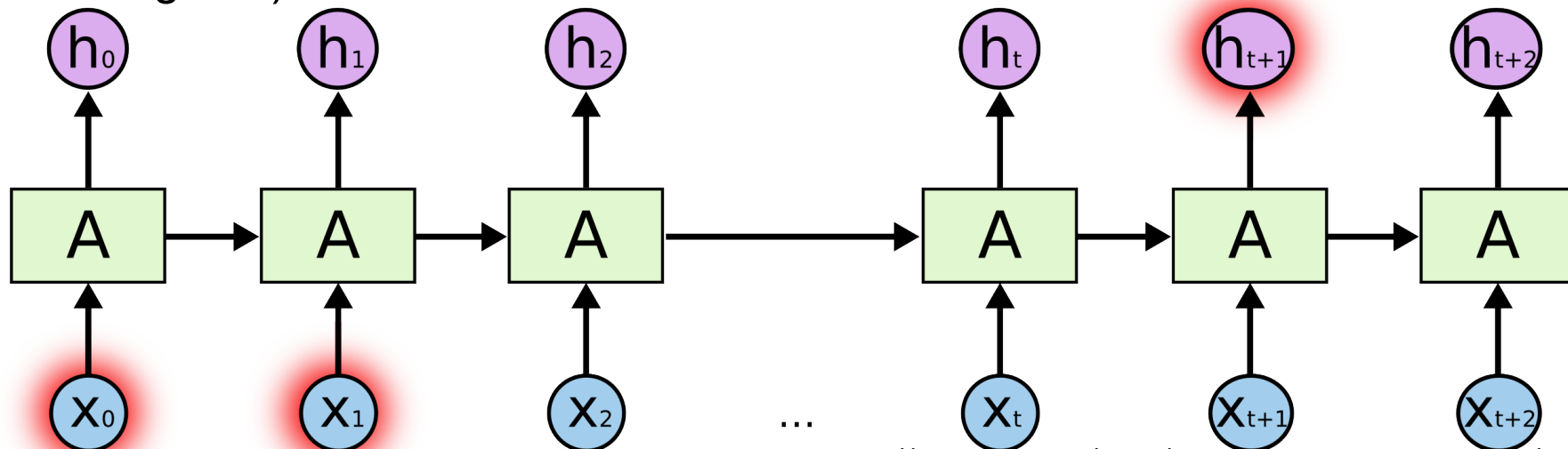
5
0.01
-0.5



With tanh

# Long-Short-Term Memory (LSTM)

- Long-term dependencies:  
*I grew up in France and lived there until I was 18. Therefore I speak fluent ???*
- Original RNN is unable to learn long term dependencies
  - Issue: More recent input data has higher influence on the output
- Long-Short-Term Memory (LSTM) models solve this problem (cell and gates)



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



# LSTM Intuition

Customers Review 2,491



Thanos

September 2018

Verified Purchase

**Amazing!** This box of cereal gave me a perfectly balanced breakfast, as all things should be. I only ate half of it but will definitely be buying again!



A Box of Cereal  
**\$3.99**



Keep relevant information

Customers Review 2,491



Thanos

September 2018

Verified Purchase

**Amazing!** This box of cereal gave me a perfectly balanced breakfast, as all things should be. I only ate half of it but will definitely be buying again!



A Box of Cereal  
**\$3.99**



img-source:

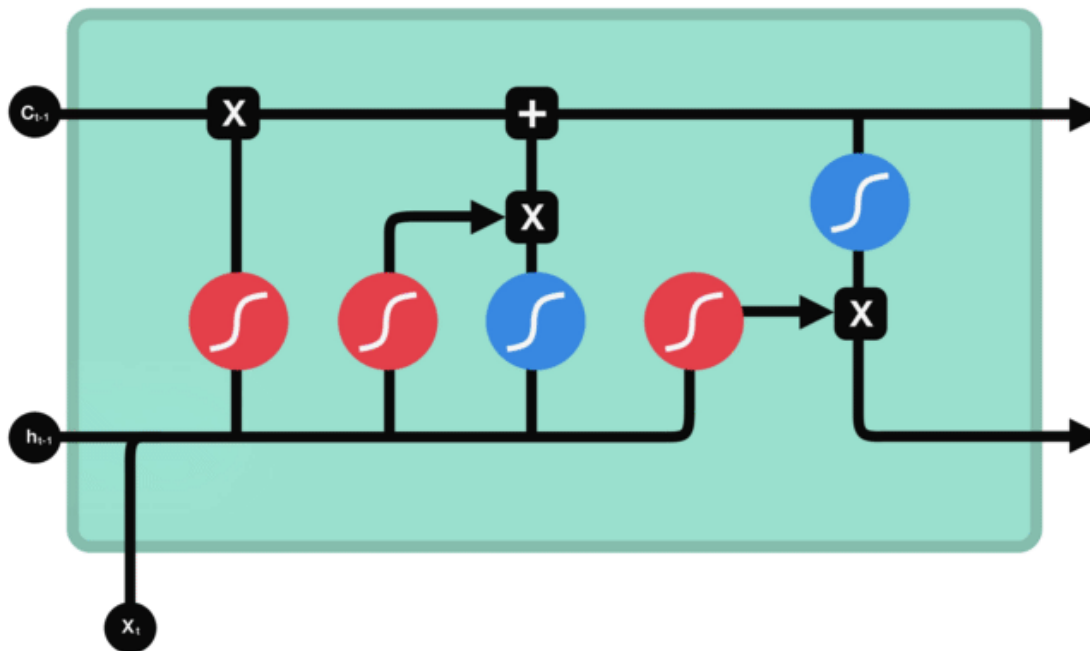
<https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>



# Forget Gate

- Decide what information to keep from previous step
- Close to 1 -> keep

  **$c_{t-1}$**  previous cell state  
  **$f_t$**  forget gate output



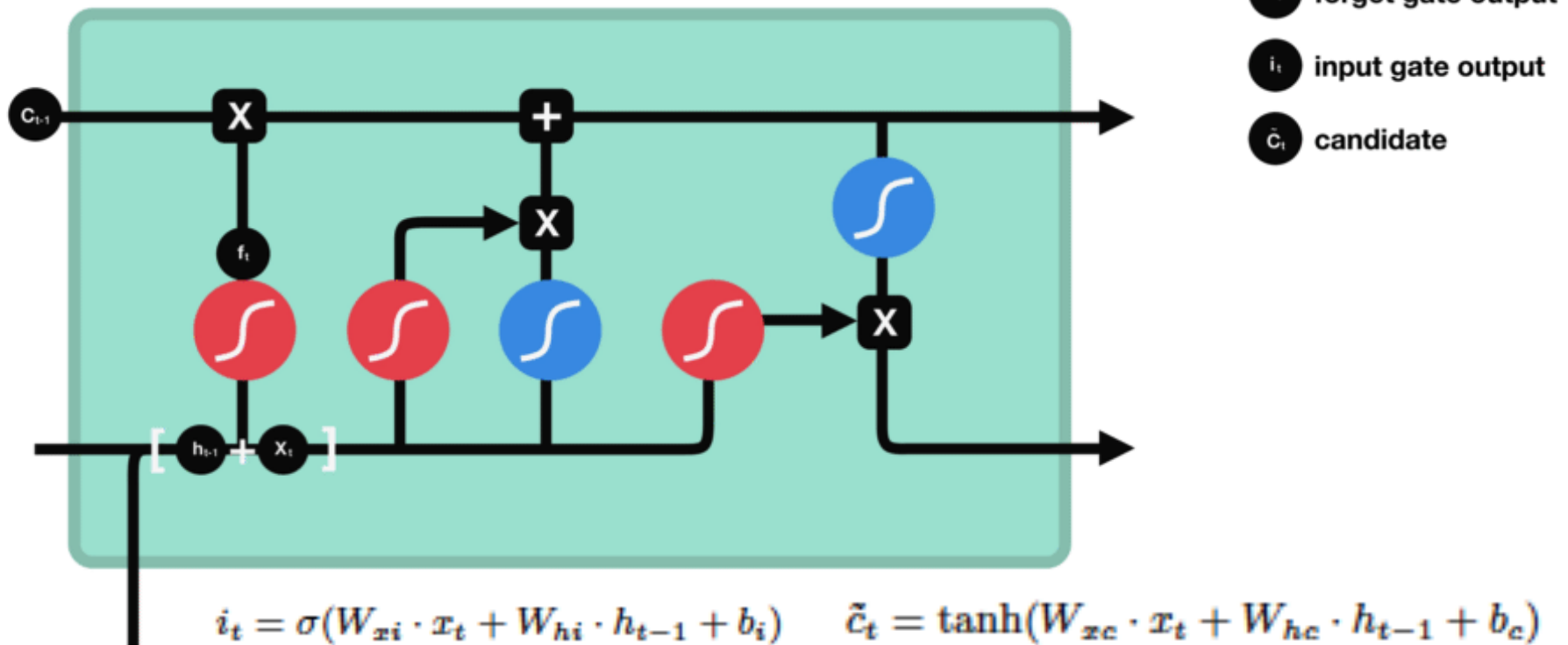
$$f_t = \sigma(W_{xf} \cdot x_t + W_{hf} \cdot h_{t-1} + b_f)$$

img-source:

<https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>

# Input Gate

- Help on updating cell state
- Keep relevant information from current step

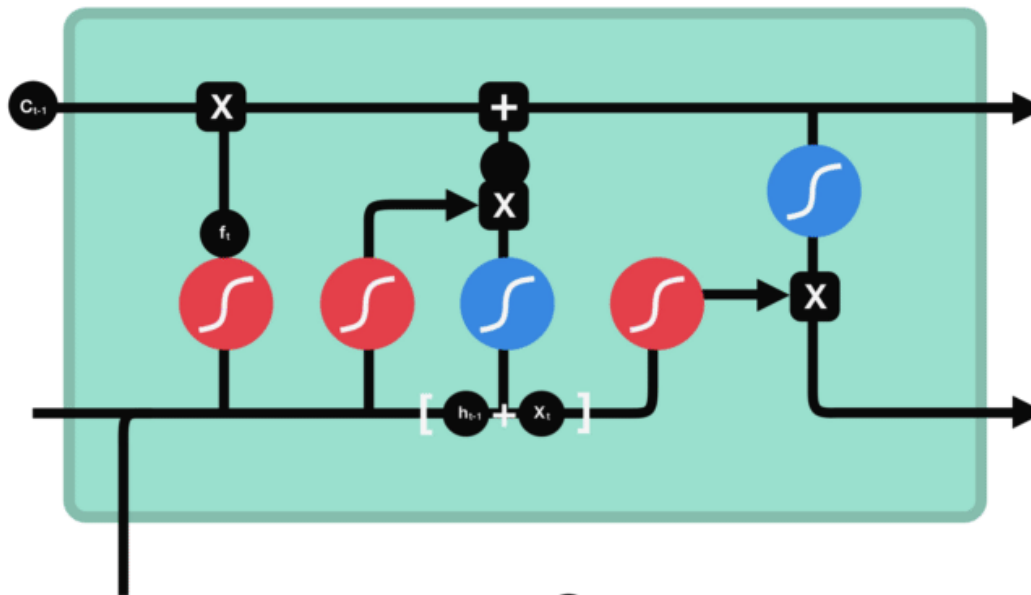


img-source:

<https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>

# Cell State

- Update cell state



- $c_{t-1}$  previous cell state
- $f_t$  forget gate output
- $i_t$  input gate output
- $\tilde{c}_t$  candidate
- $c_t$  new cell state

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t$$

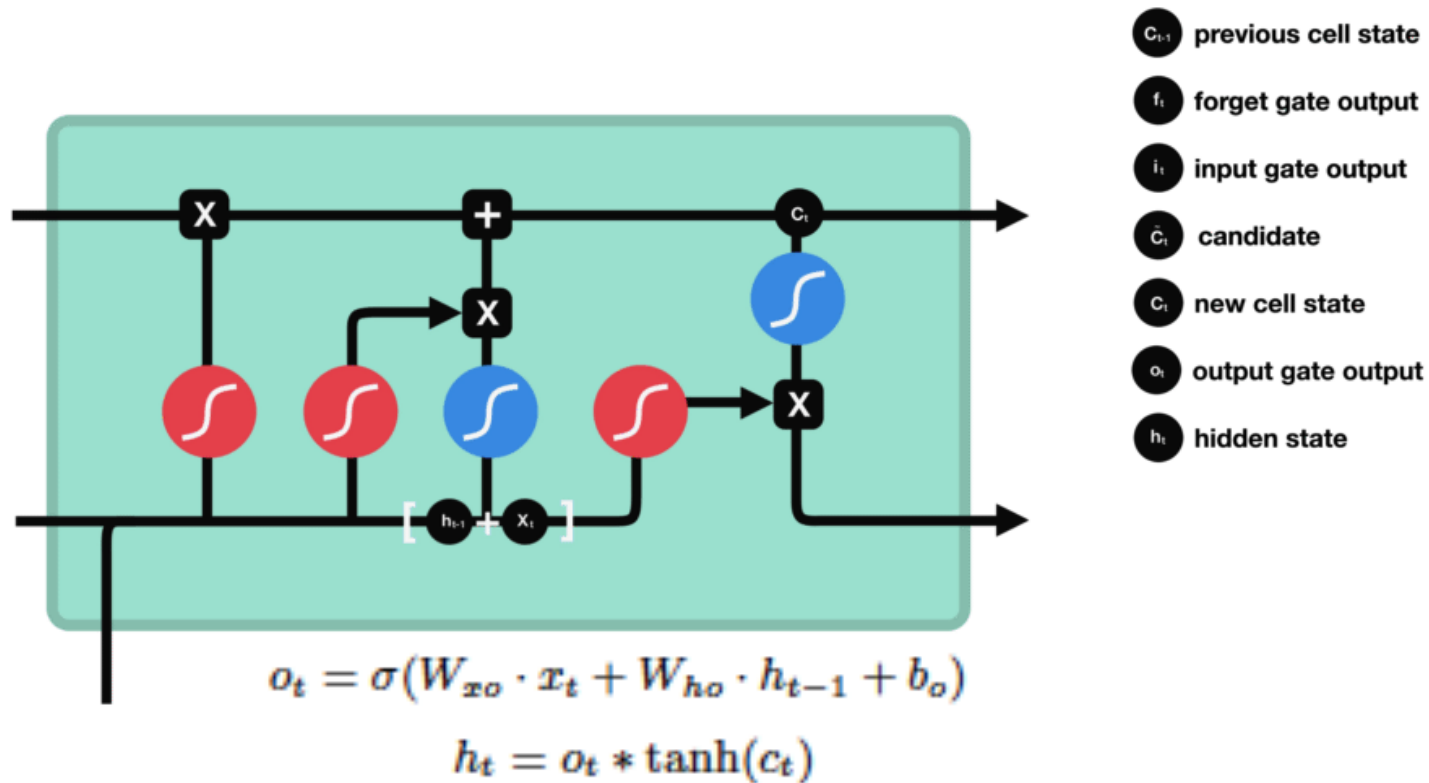
$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t$$

img-source:

<https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>

# Output Gate

- Create next hidden state from the input and the new cell state

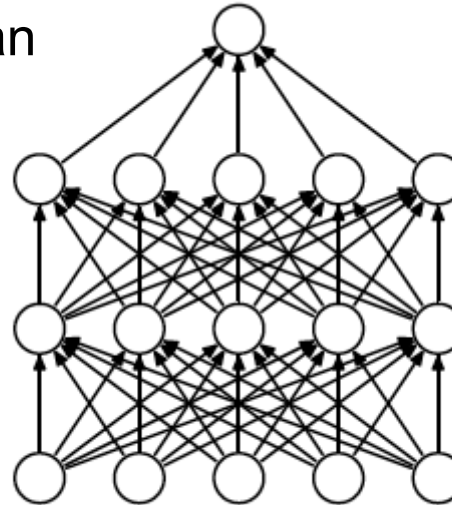


img-source:

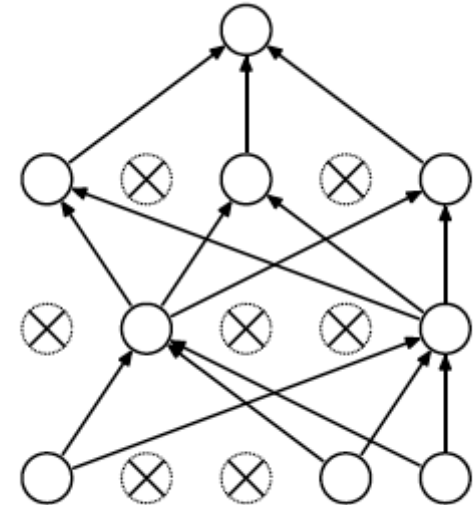
<https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>

# Appendix: Dropout

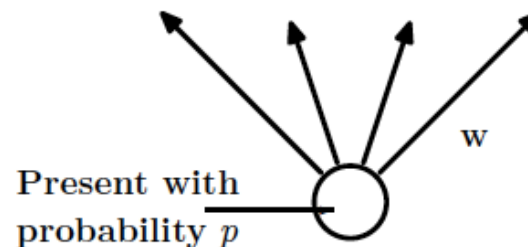
- Probability  $p$  of removing an edge
- Samples different architectures
- Share the same weight
- Regularization method



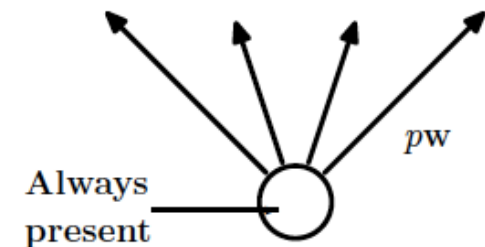
(a) Standard Neural Net



(b) After applying dropout.



(a) At training time



(b) At test time