Deep Learning: Sequencial Models

Luciano Barbosa

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

Centro de Informática UFPE

Universidade Federal de Pernambuco

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?

Convolution
Pooling
Softmax
Other

- Google’s Entry for the 2014 ImageNet Challenge
- 5 million parameters – 20MB model size

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

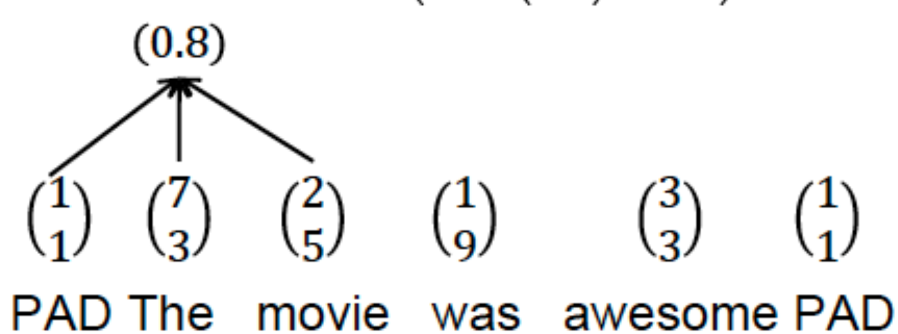
Word Vectors: $w_i \in \mathbb{R}^2$

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: $b \in \mathbb{R}$

$$\left.\begin{array}{l} W \in \mathbb{R}^{1\times 6} \\ b \in \mathbb{R} \end{array}\right\} \text{Local Filter}$$

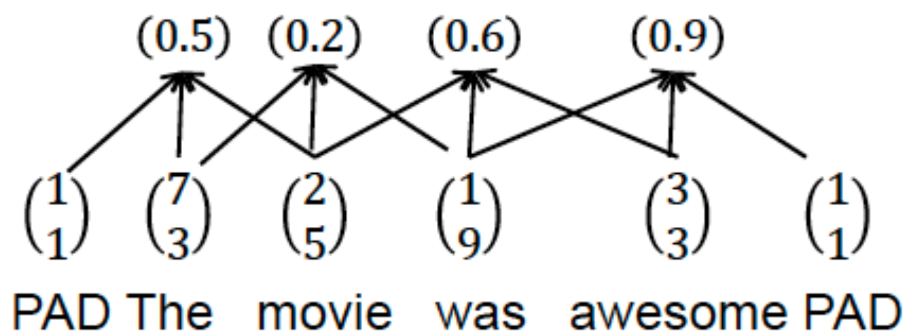
$$\text{than} \quad \left[\begin{bmatrix} \quad W \quad \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 7 \\ 3 \\ 2 \\ 5 \end{bmatrix} + b \right] = 0.8$$

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

$$(0.8)$$

$$\begin{array}{cccccc} \binom{1}{1} & \binom{7}{3} & \binom{2}{5} & \binom{1}{9} & \binom{3}{3} & \binom{1}{1} \\ \text{PAD} & \text{The} & \text{movie} & \text{was} & \text{awesome} & \text{PAD} \end{array}$$

Single Layer CNN – Shared Weights



- 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?

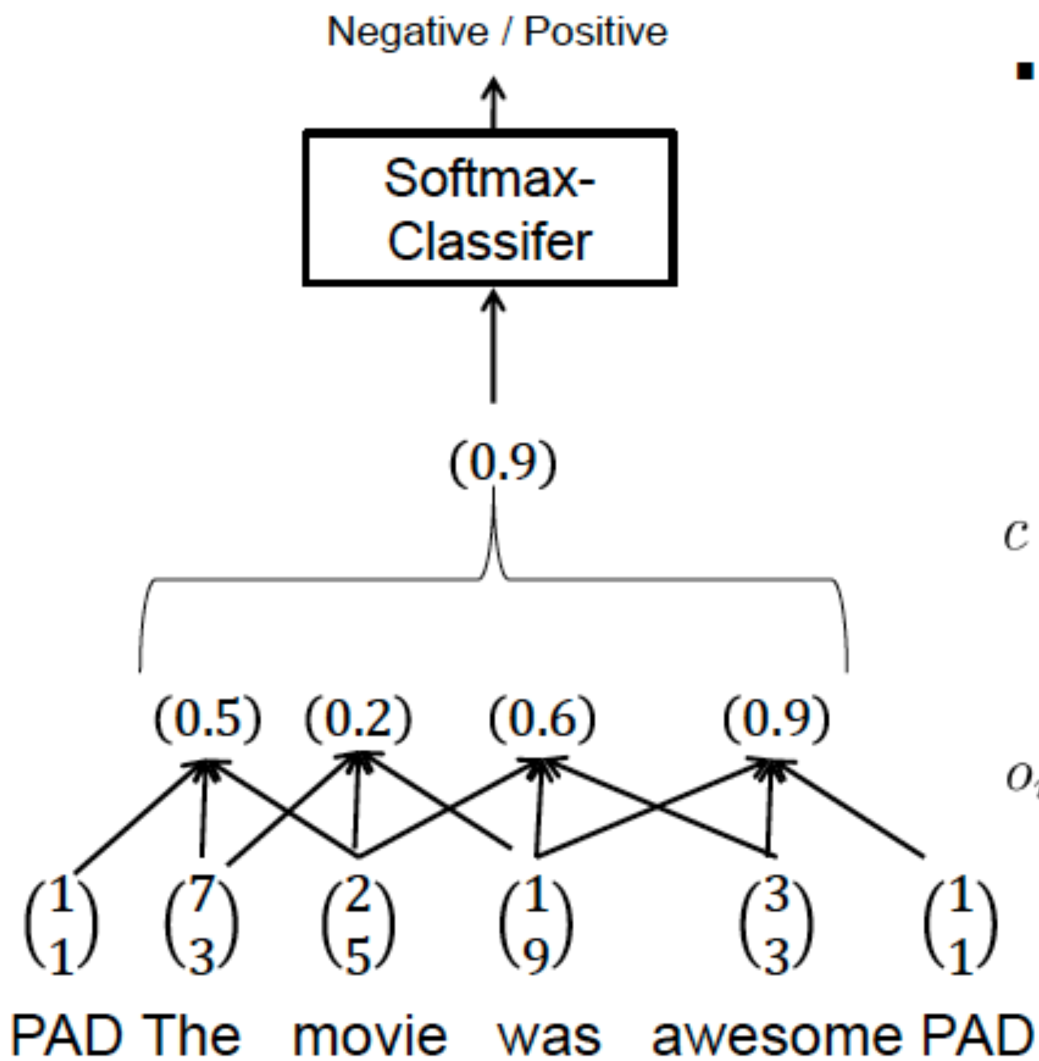
Pooling Layer

- Idea: to capture the most important activation
- Example: max-pooling layer
- Let $o_1, o_2, \ldots \in \mathbb{R}$ 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_i
- Max-pooling most common in NLP. In Computer Vision, min-pooling and mean-pooling also common.

Max Pooling

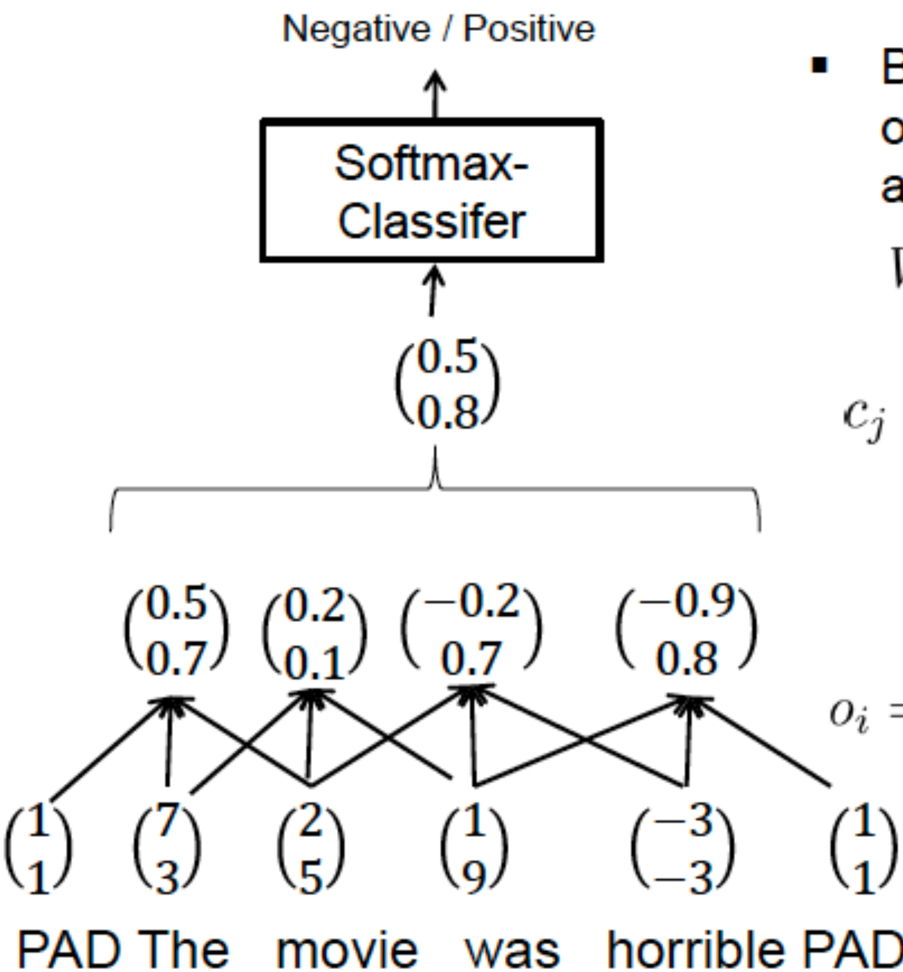


- You can train this like any other Neural Network

$$c = \max_i(o_i)$$

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

Increasing Number of Filters: K=2



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

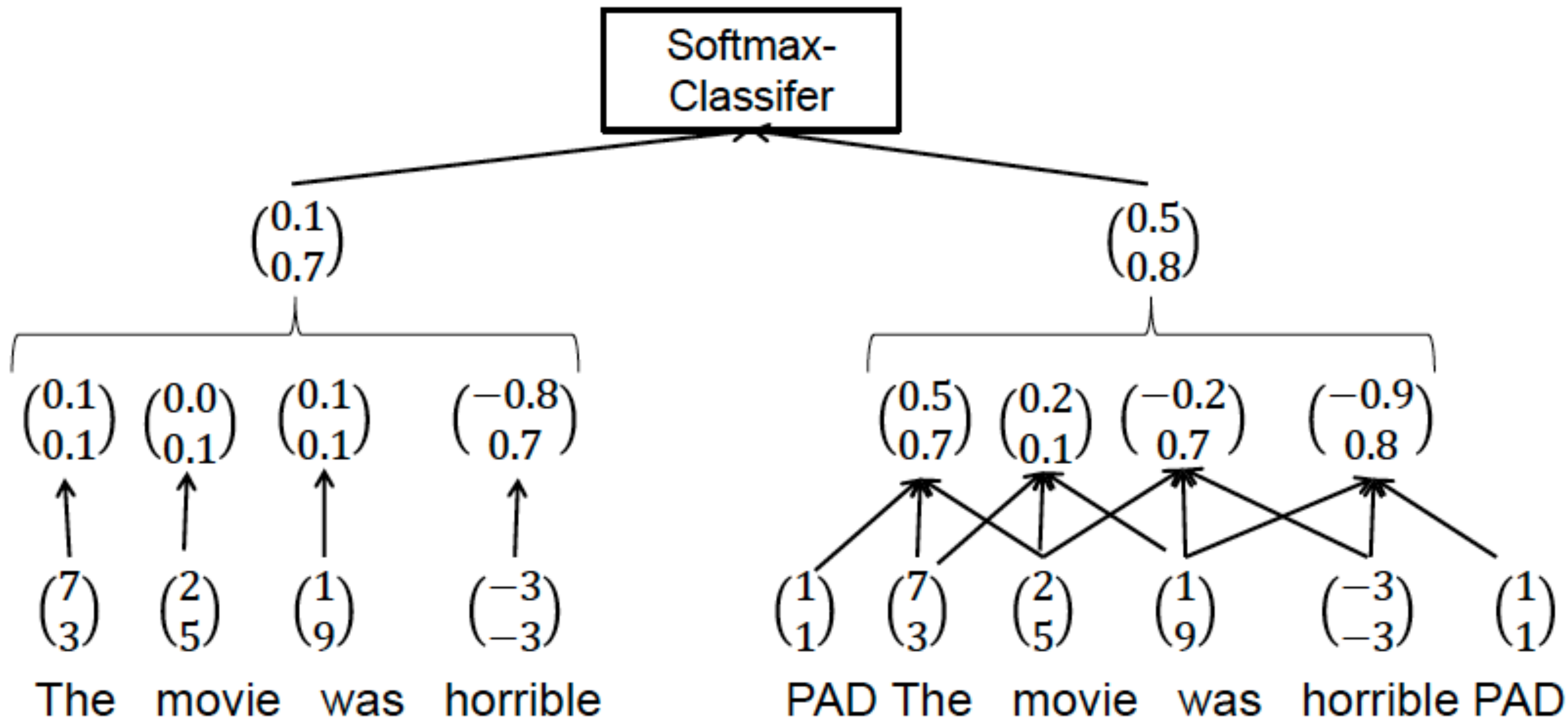
$$W \in \mathbb{R}^{k\times 6}$$

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

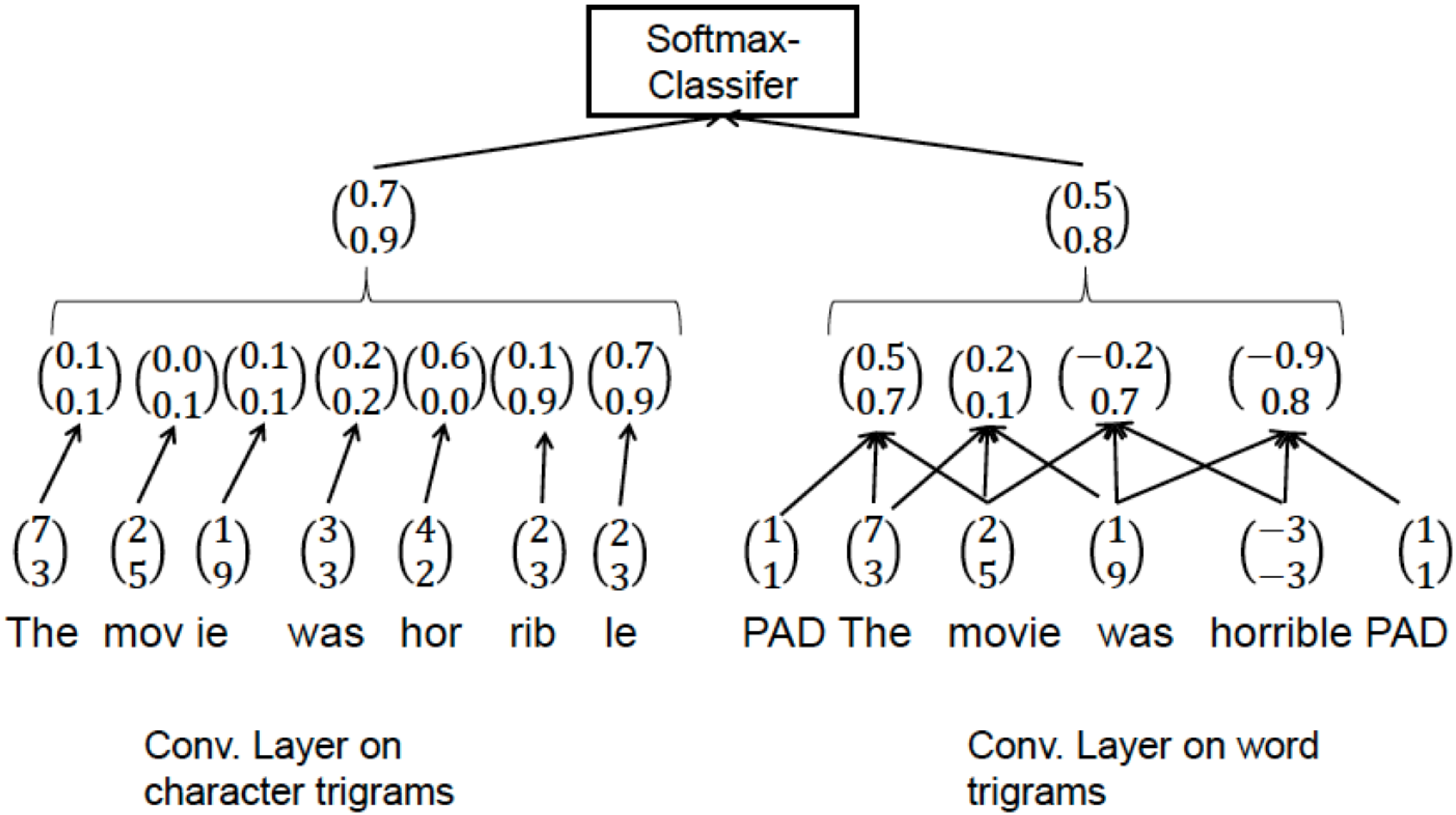
$$o_i = \tanh\left(W\begin{pmatrix} w_{i-1} \\ w_i \\ w_{i+1}\end{pmatrix} + b\right) \in \mathbb{R}^k$$

Increasing Windows Size

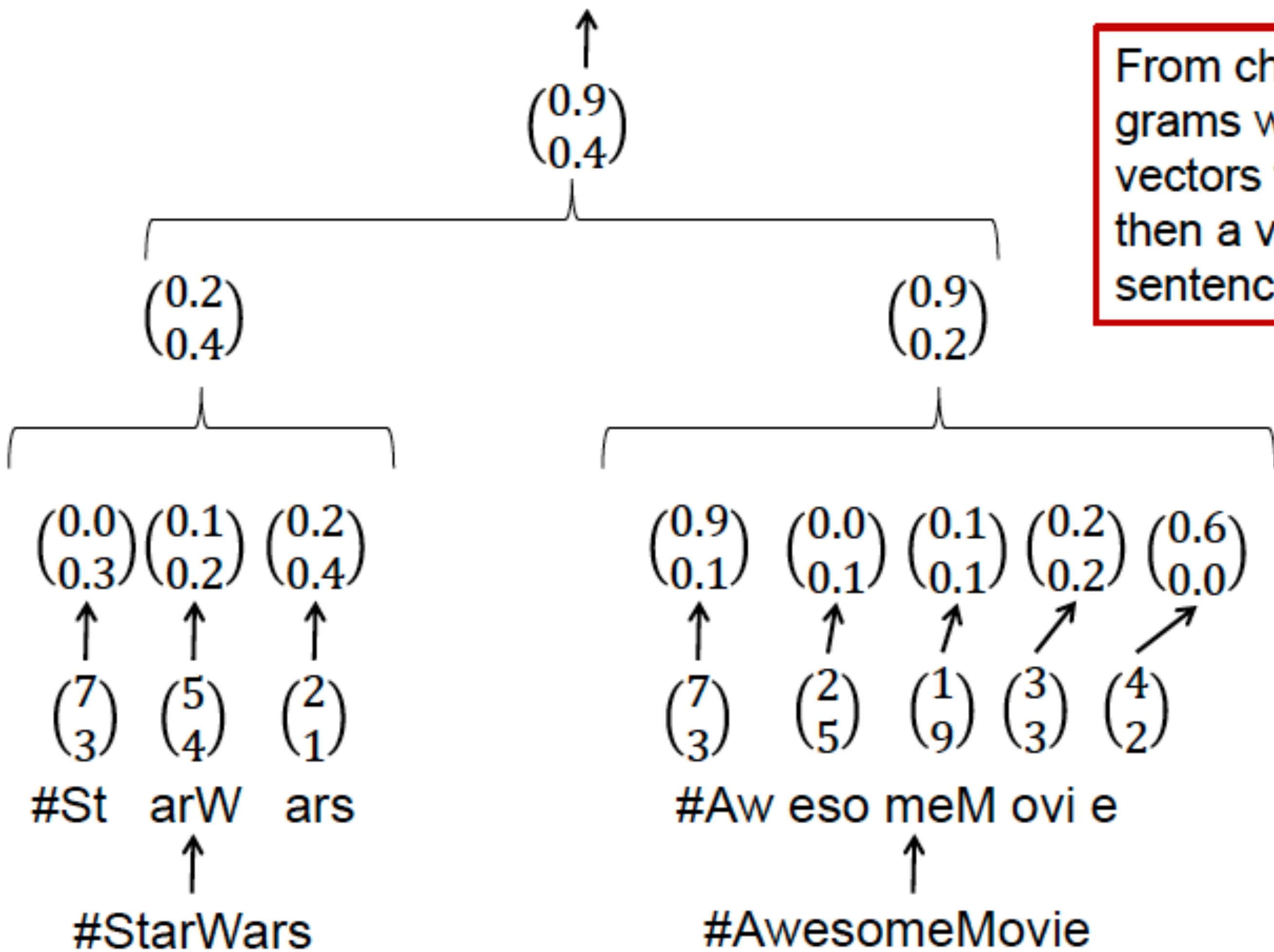
- Combining output of unigrams and trigrams



Different granularity



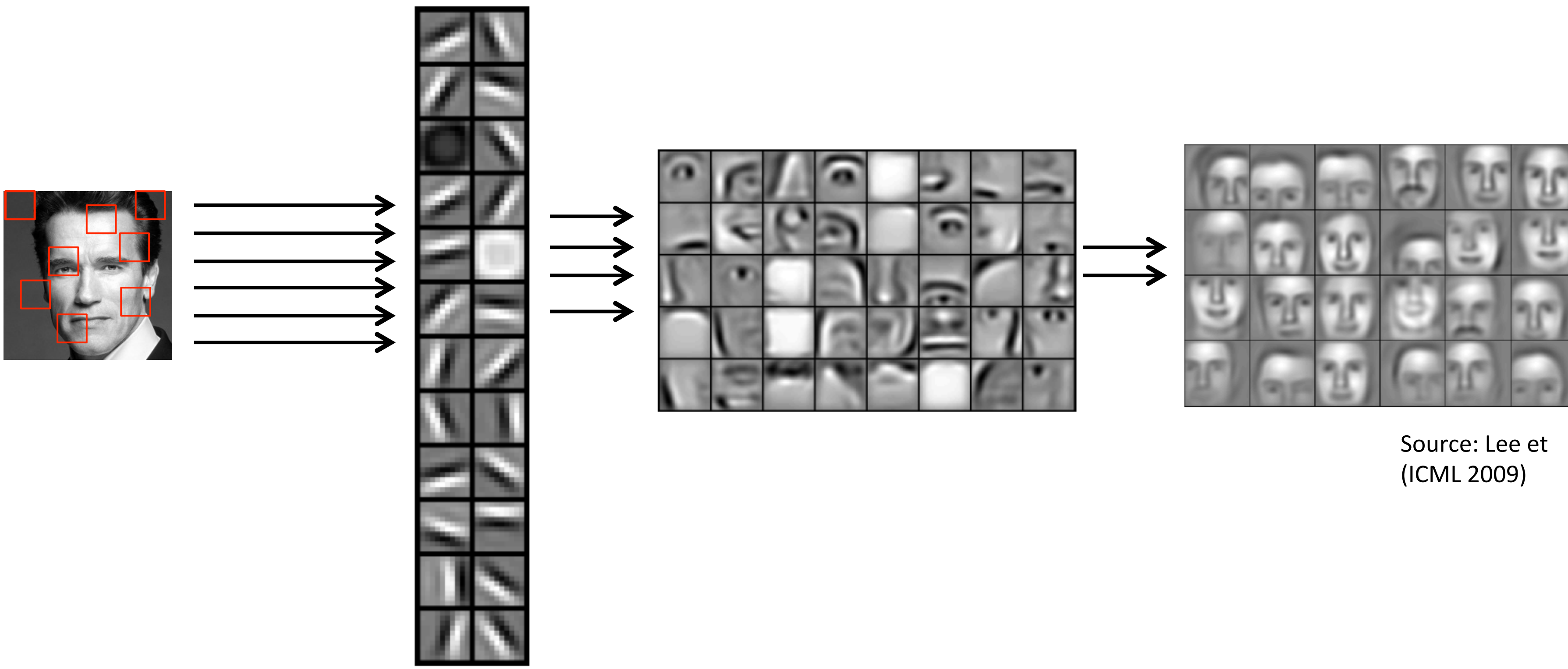
Stacking Convolutional Layers



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)

Input
e.g. 40 000 dim

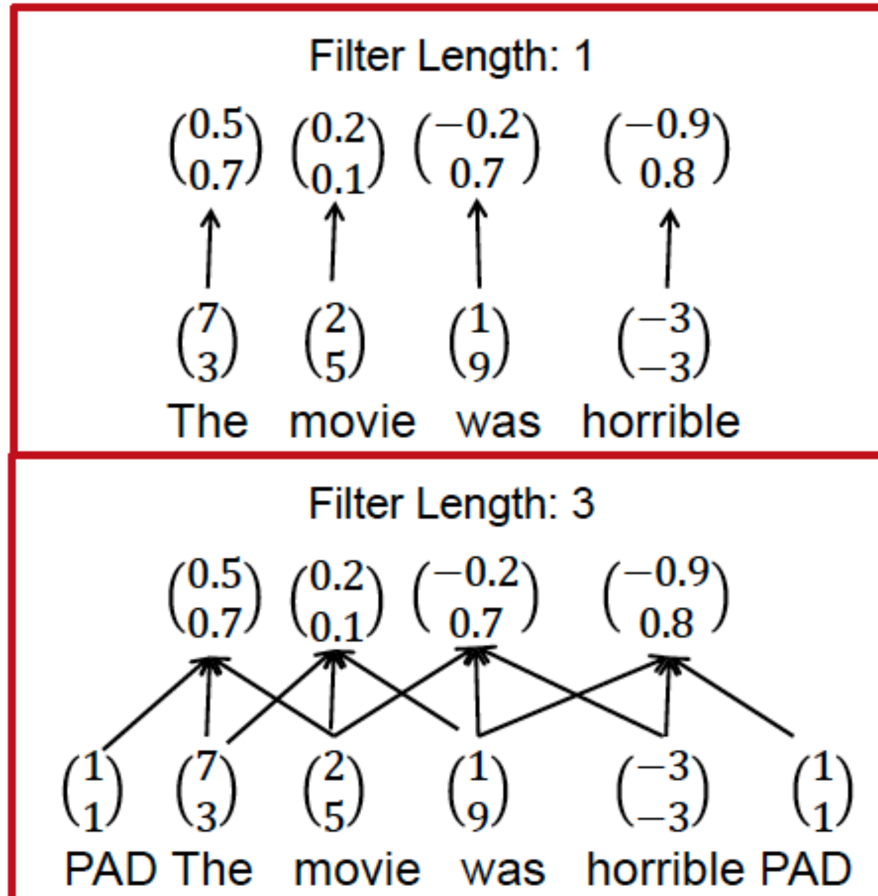
Layer 1
9 600 dim

Layer 2
3800 dim

Layer 3
900 dim

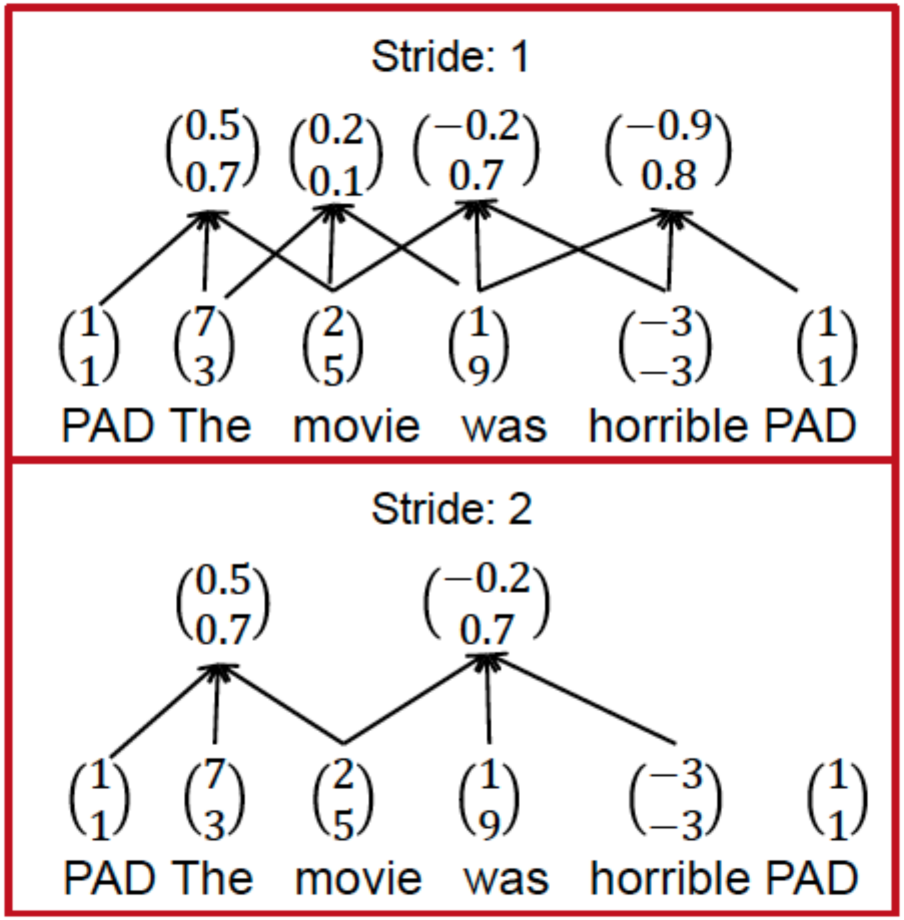
Terminology: Filter Length

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



Terminology: Stride

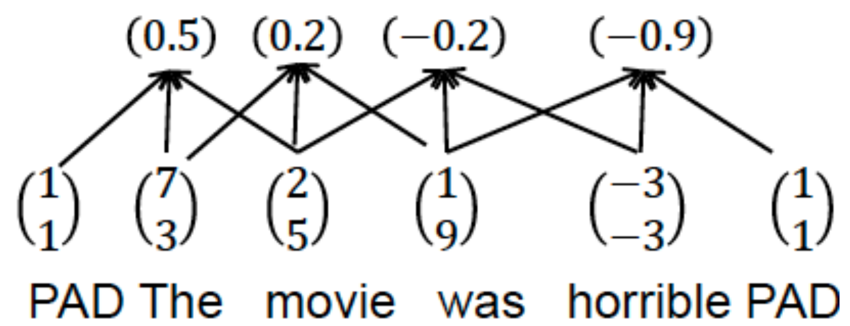
- The *stride* specifies the step size we move across a sentence
- In NLP: typically stride=1
- In computer vision: Other values can be used



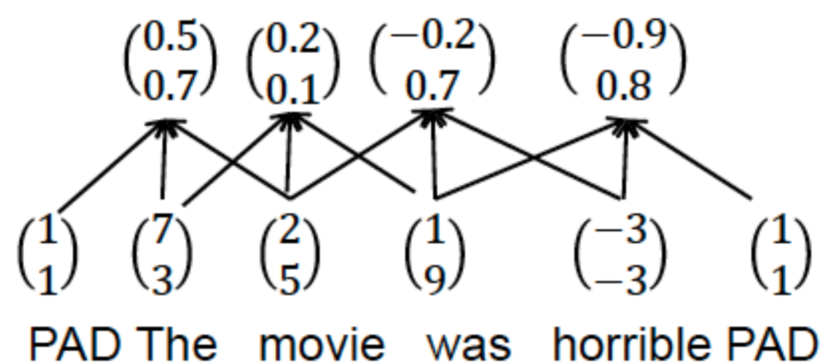
Terminology: Number of Filters

- Size of the CNN's output vector

Number of filters: 1

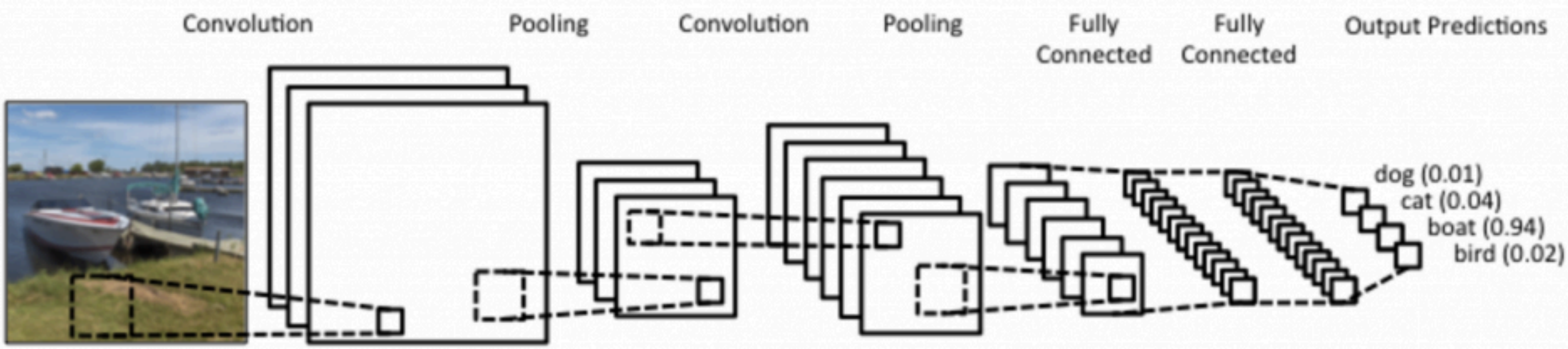


Number of filters: 2



Applications of CNNs

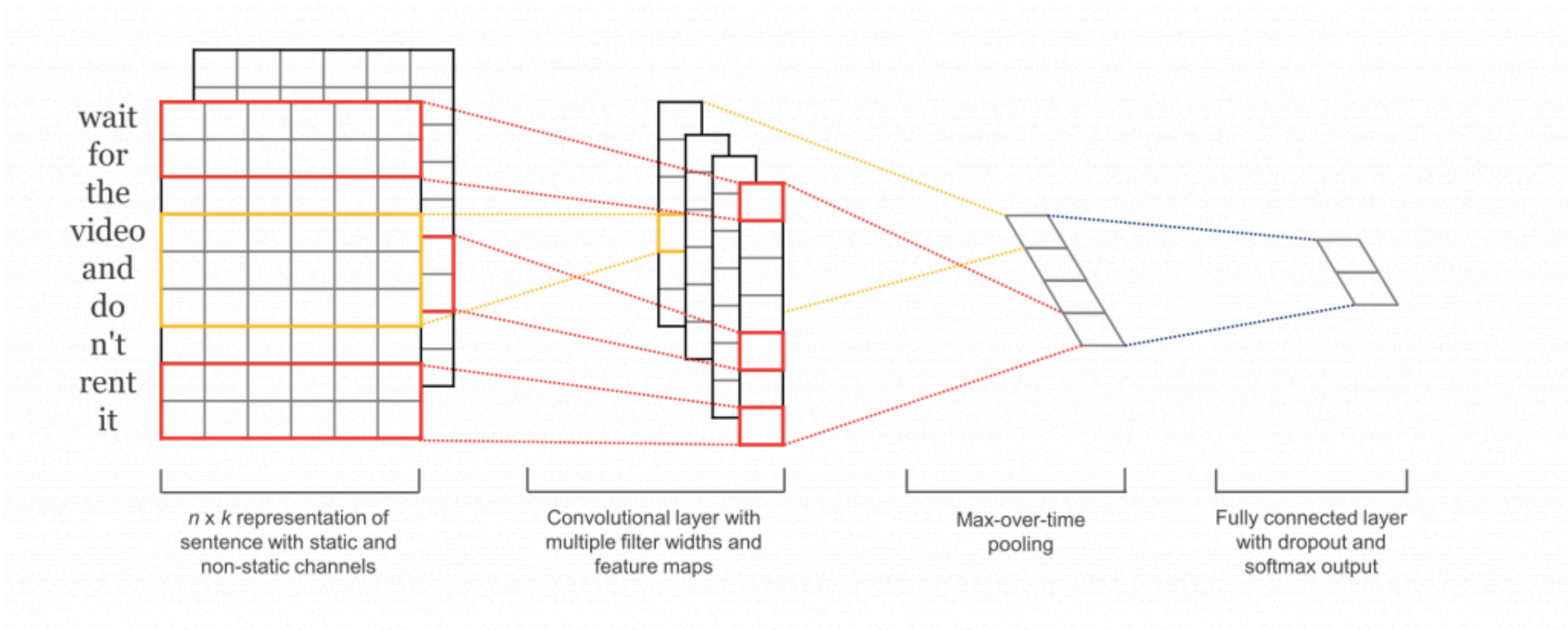
- Object detection



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

Applications of CNNs

- Sentiment analysis



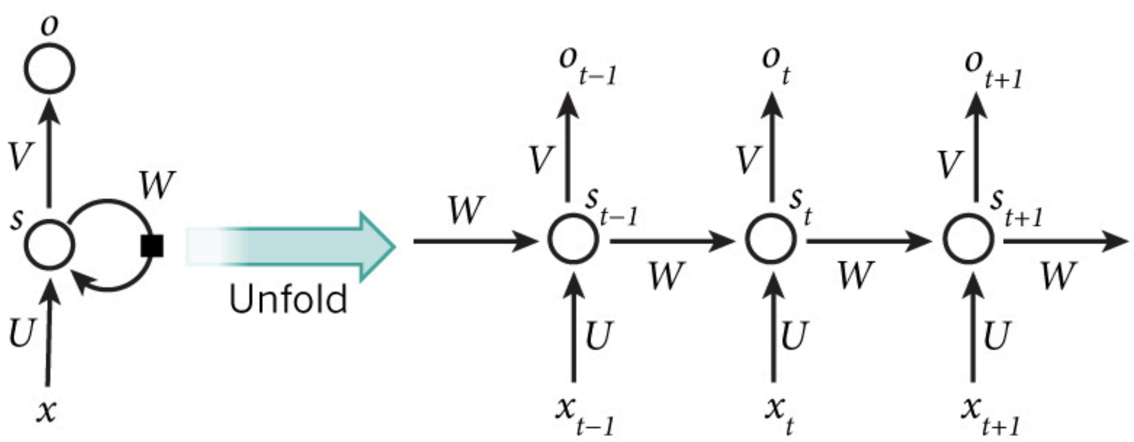
Img-Source: https://arxiv.org/abs/1408.5882

Applications of CNNs

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

Recurrent Neural Networks

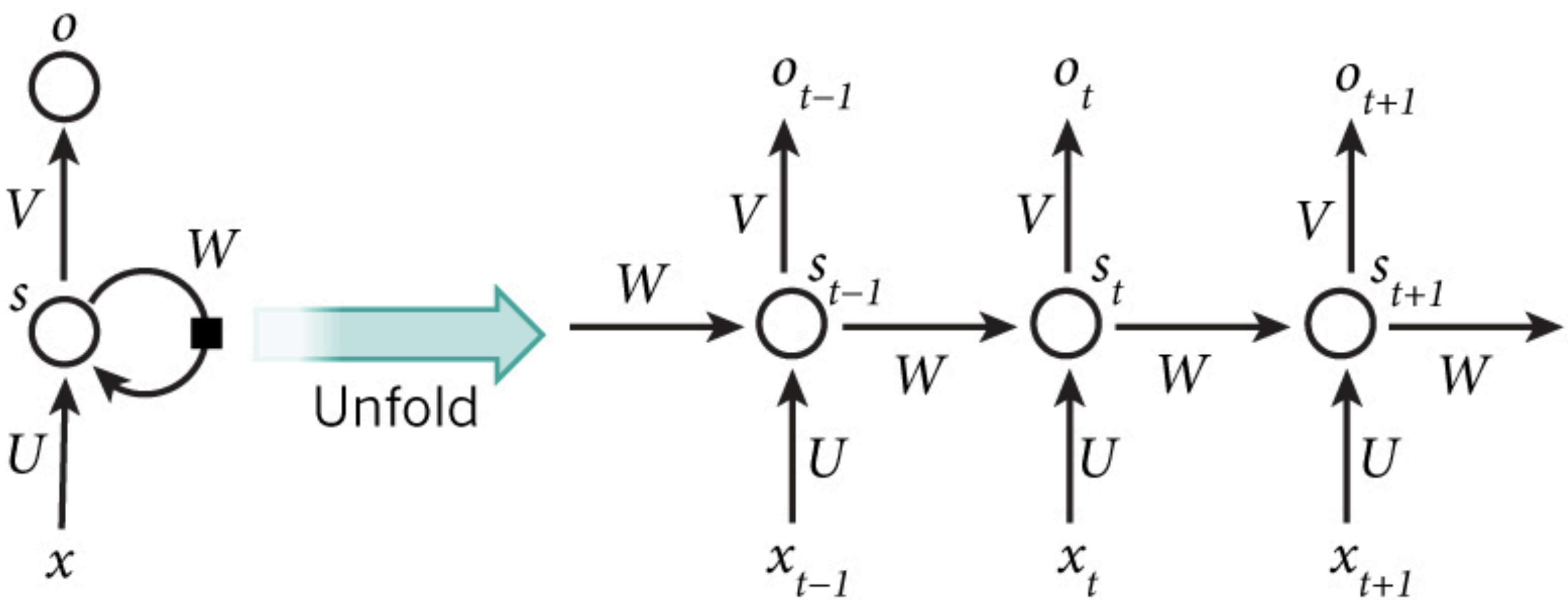
- Traditional NNs: don't take into account proximity
- Used for sequential information



- 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

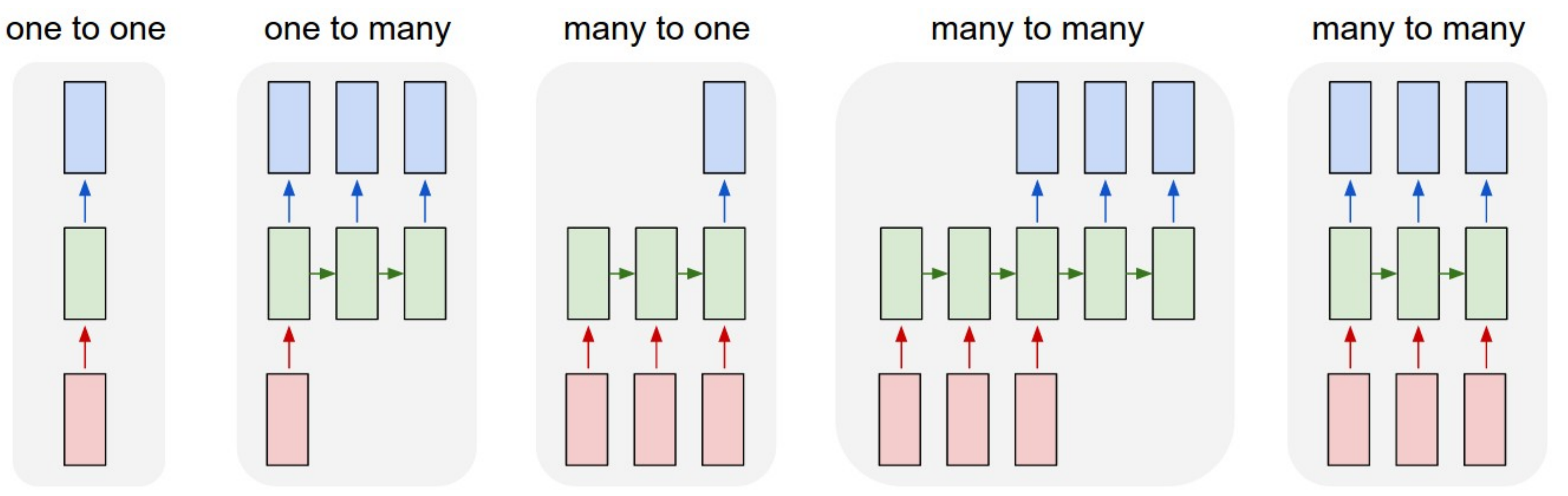
Img Source: http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

Unfolded RNN



Img Source: http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

Topologies of RNNs



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

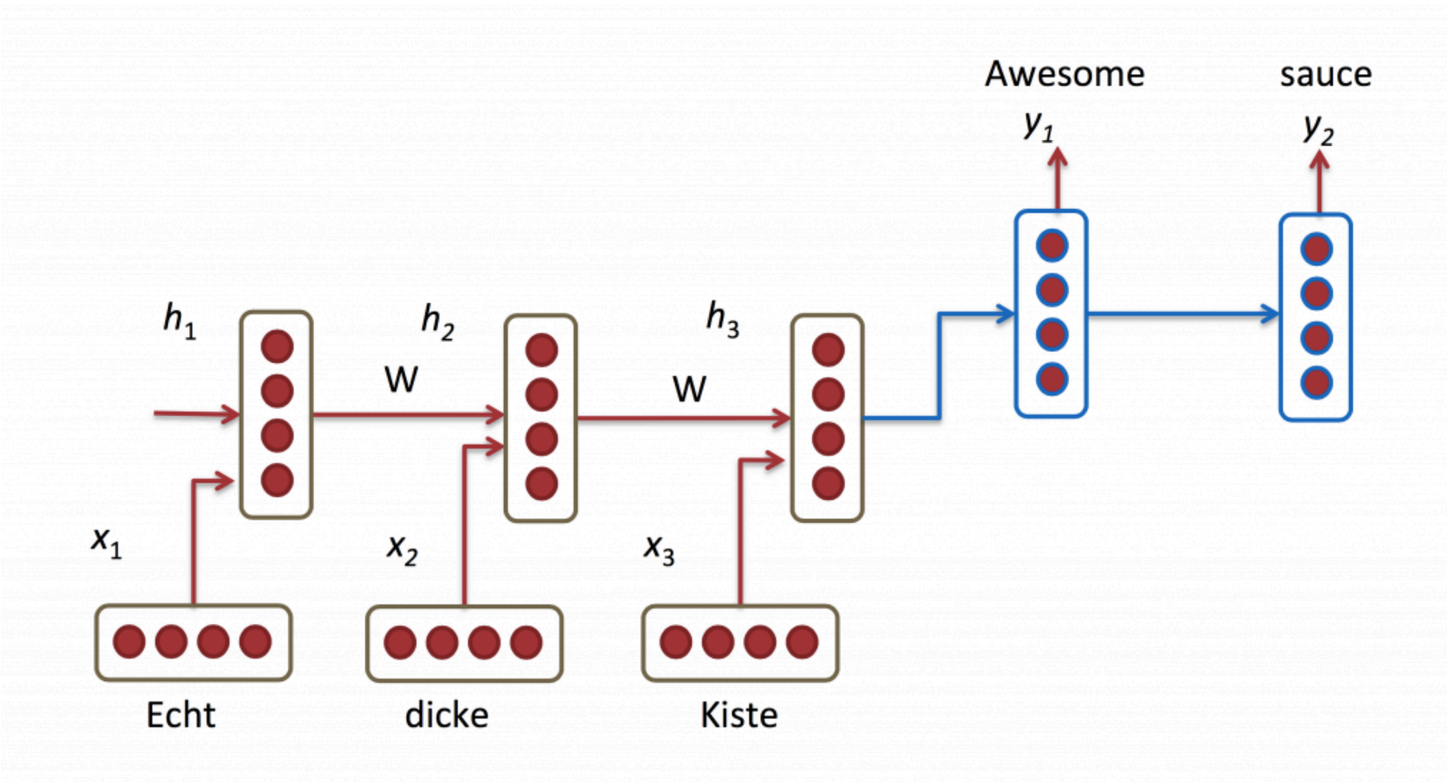
- Language modeling and text generation
  - Example: trained from essays in economic topics

> *"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."*

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

Applications of RNN

- Machine translation



Img Source: http://cs224d.stanford.edu/lectures/CS224d-Lecture8.pdf

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."

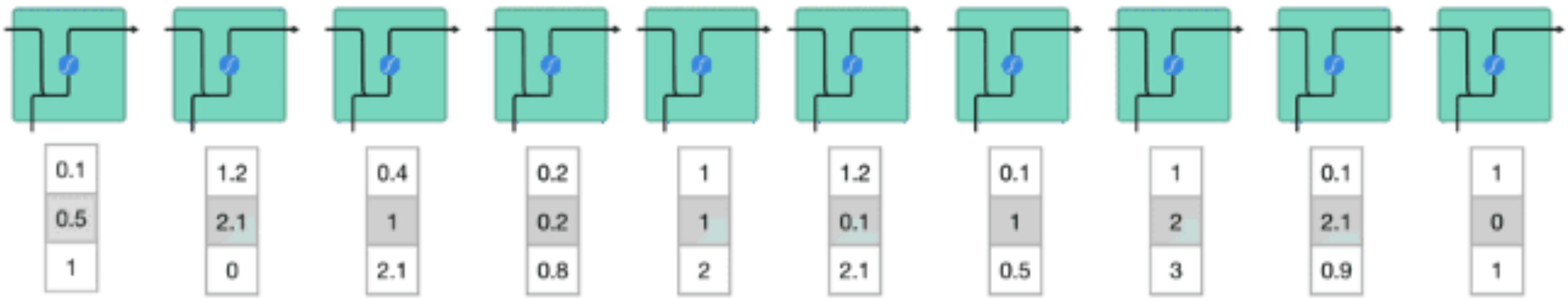

"a cat is sitting on a couch with a remote control."


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

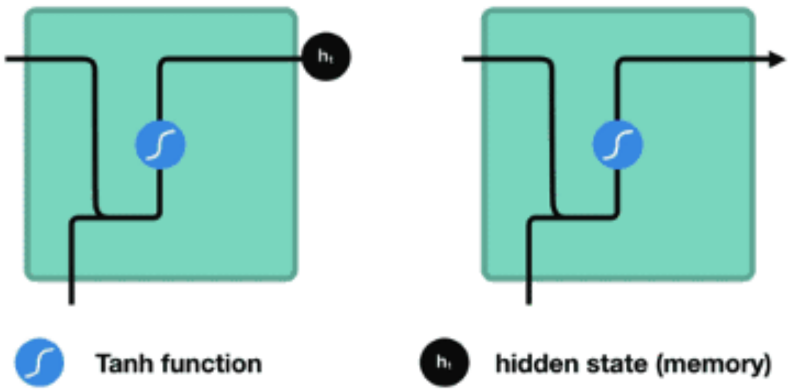

"a horse is standing in the middle of a road."

Img Source: http://cs224d.stanford.edu/lectures/CS224d-Lecture8.pdf

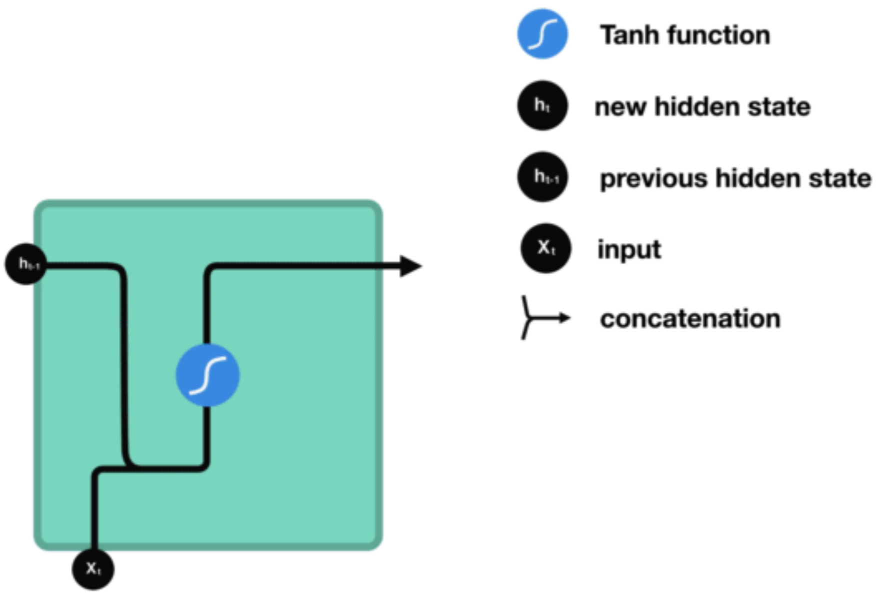
RNN



RNN

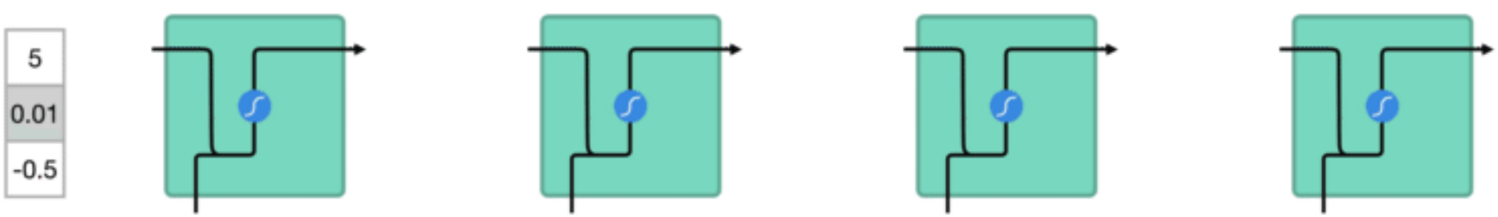


RNN Cell


h_{t-1}
x_t
Tanh function
h_t
new hidden state
h_{t-1}
previous hidden state
x_t
input
concatenation

RNN: Why Use Tanh

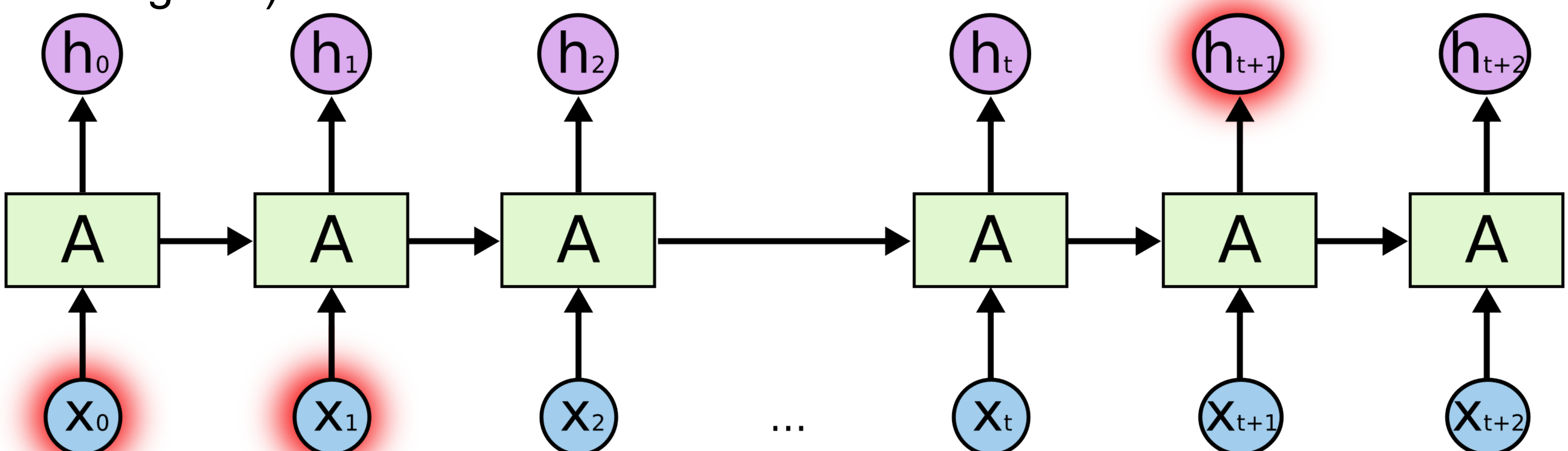
Without tanh



With tanh

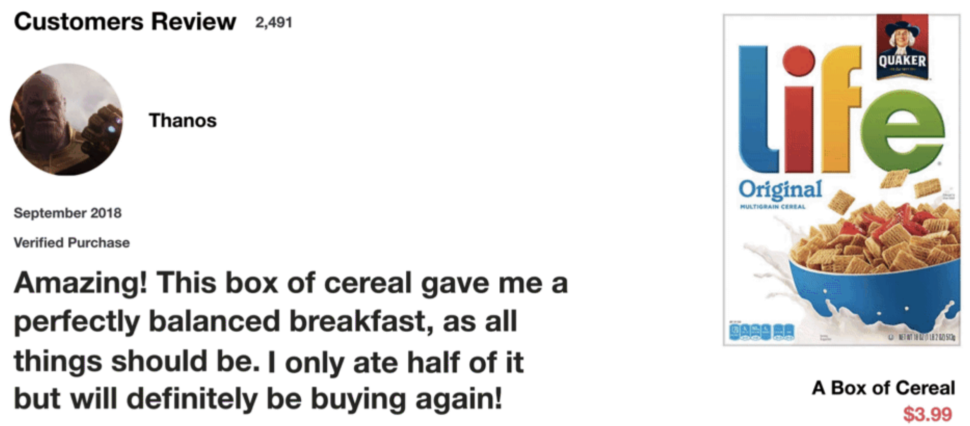
Long-Short-Term Memory (LSTM)

- 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)

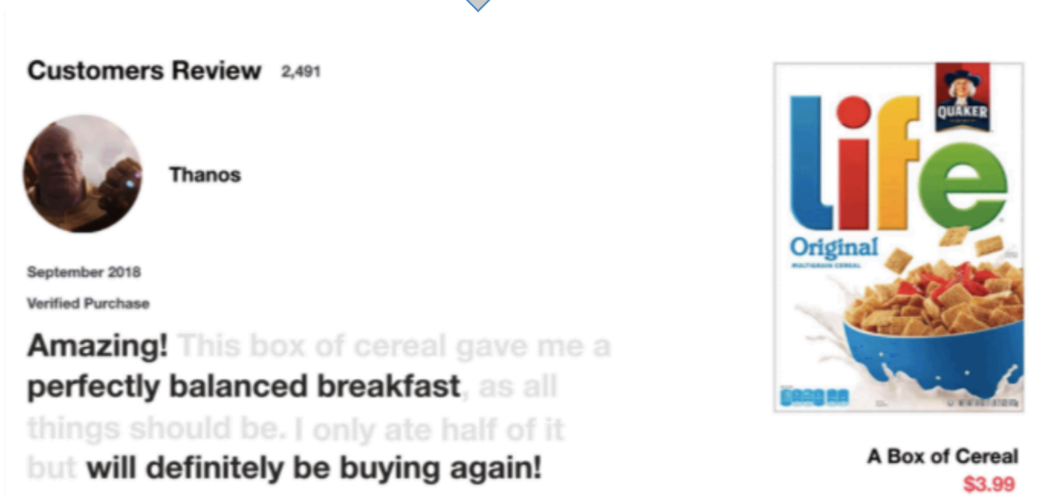


Img Source: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM Intuition

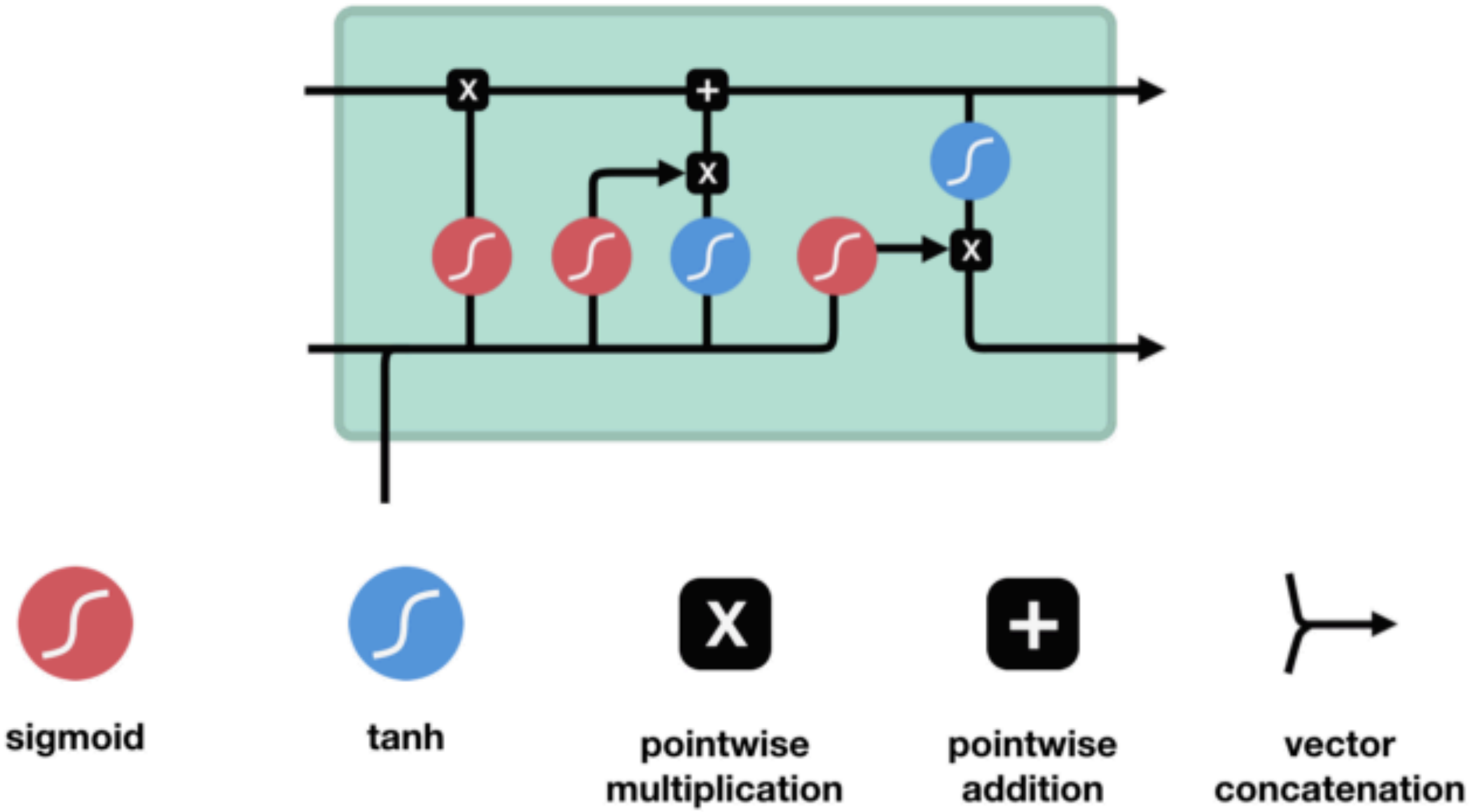


Keep relevant information



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

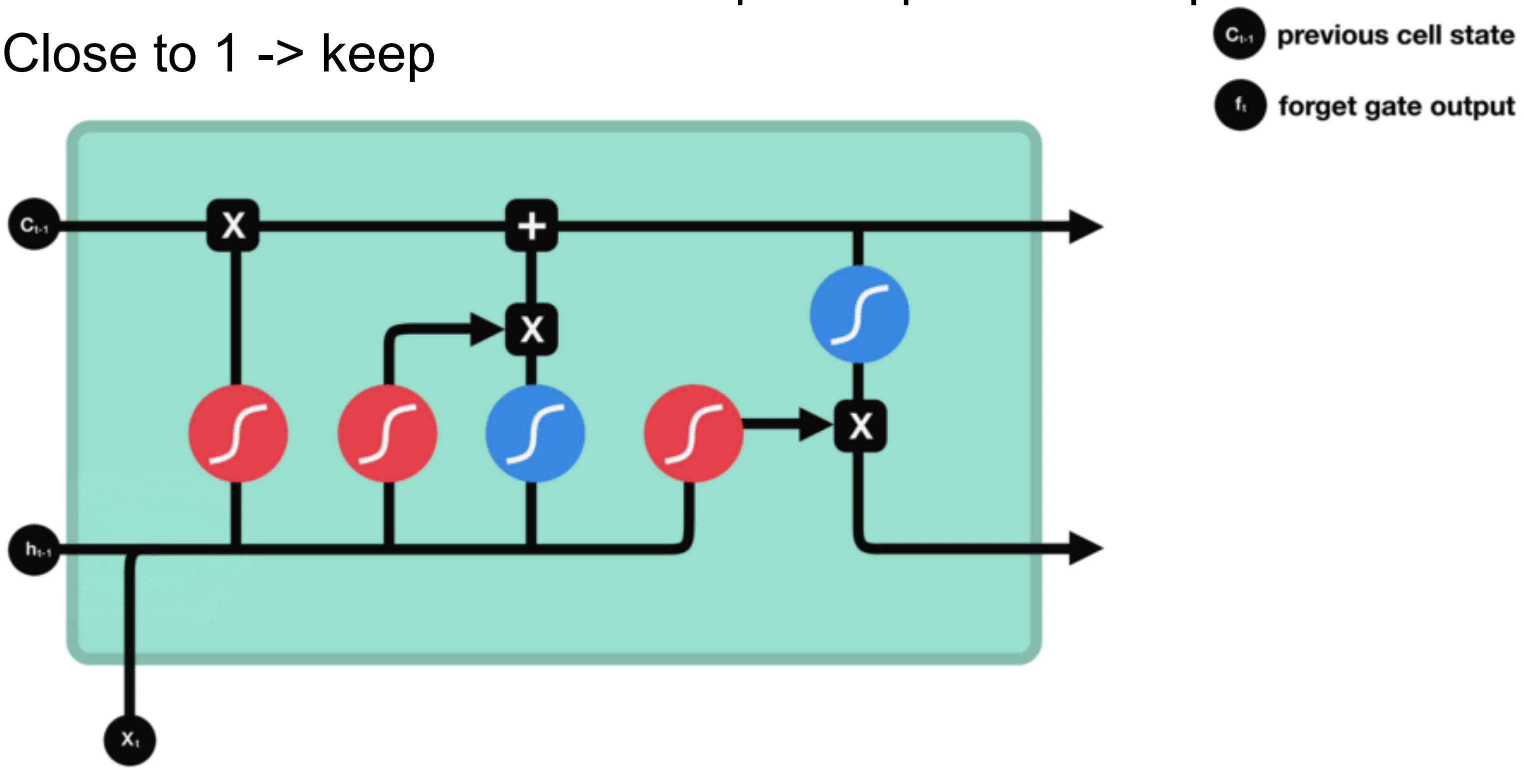
LSTM's Cell



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

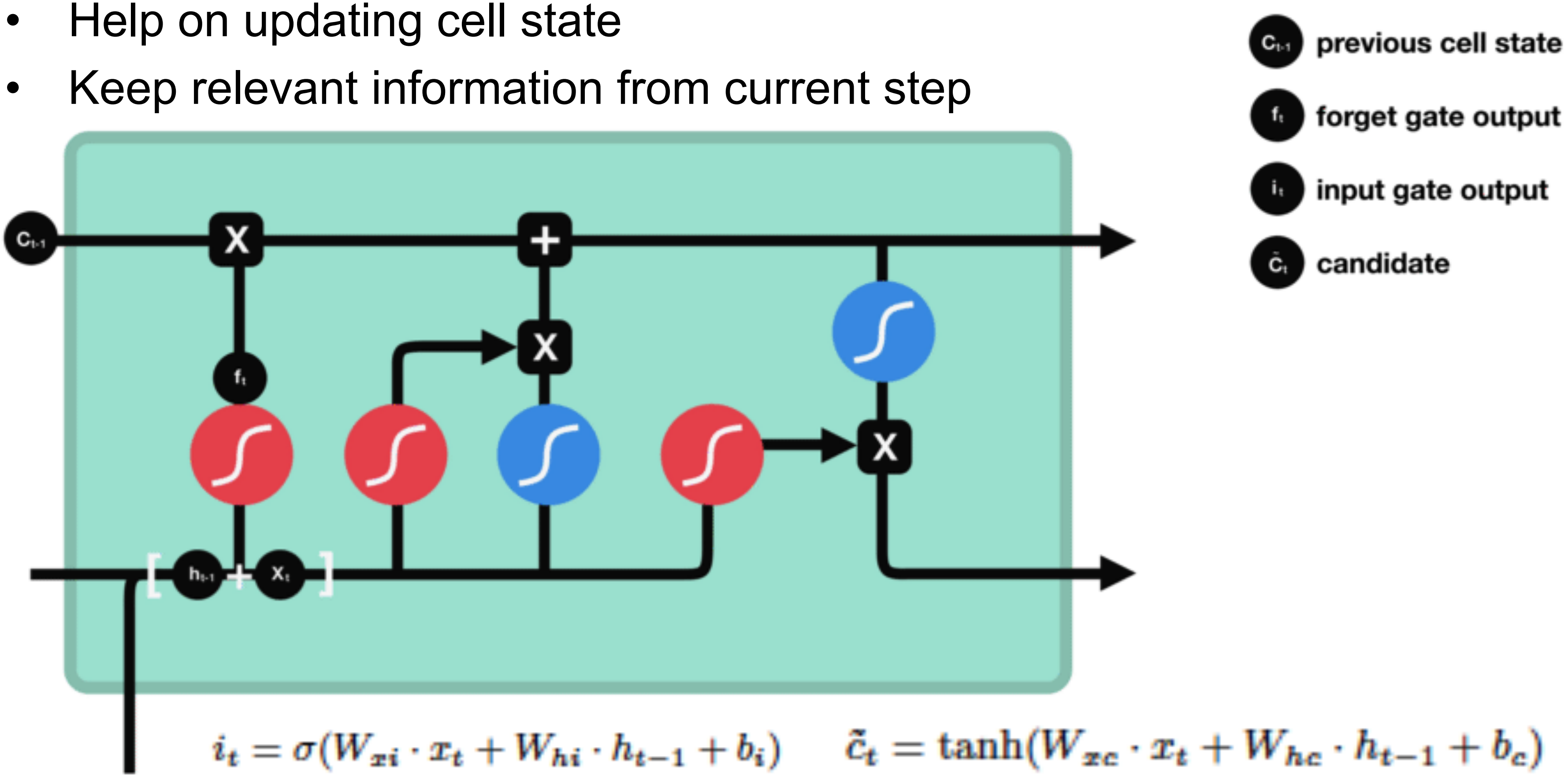


$$f_t = \sigma(W_{xf} \cdot x_t + W_{hf} \cdot h_{t-1} + b_f)$$

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

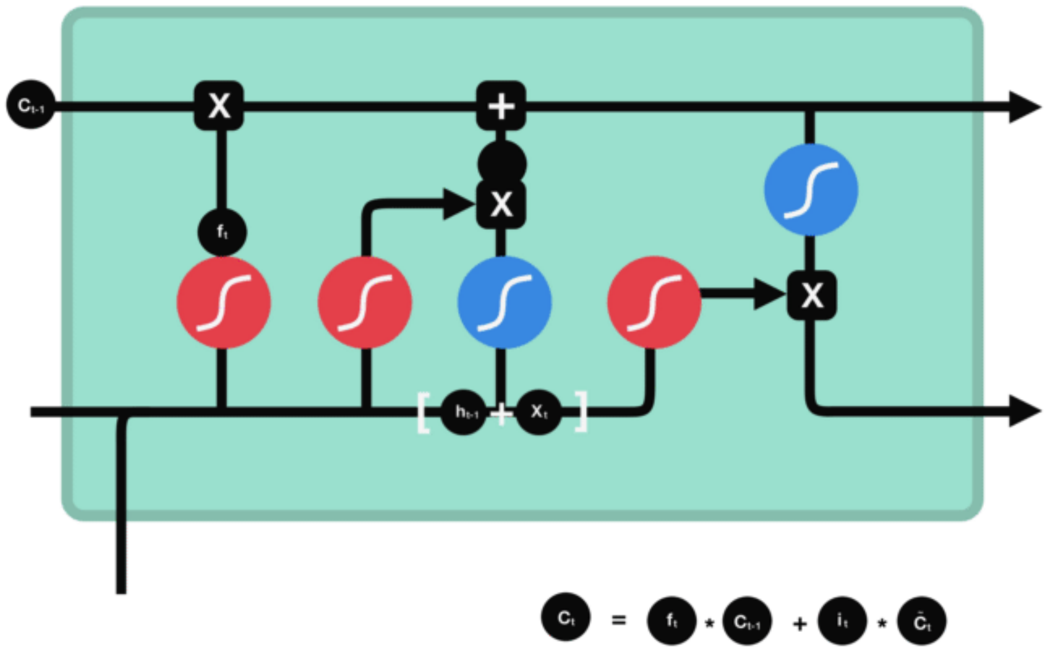


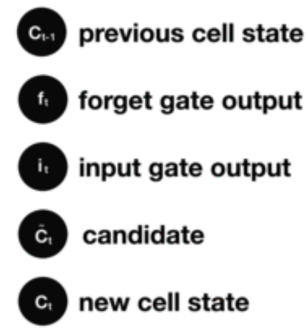
$$i_t = \sigma(W_{xi} \cdot x_t + W_{hi} \cdot h_{t-1} + b_i) \qquad \tilde{c}_t = \tanh(W_{xc} \cdot x_t + W_{hc} \cdot h_{t-1} + b_c)$$

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

Cell State

- Update cell state



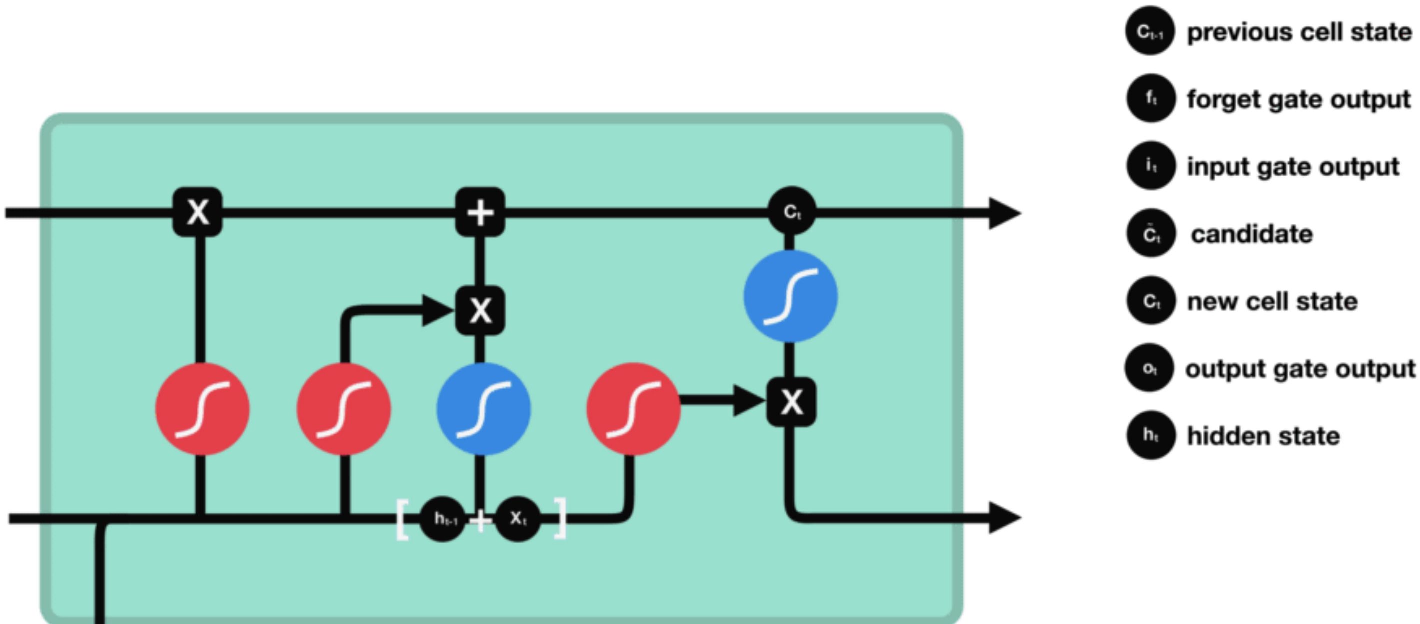


$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t$$

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



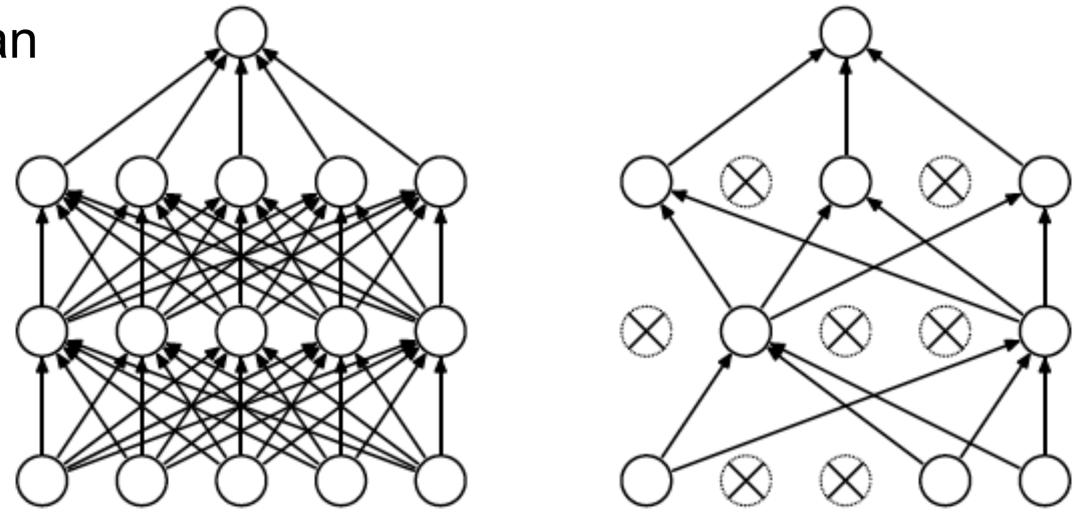
$$o_t = \sigma(W_{xo} \cdot x_t + W_{ho} \cdot h_{t-1} + b_o)$$

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

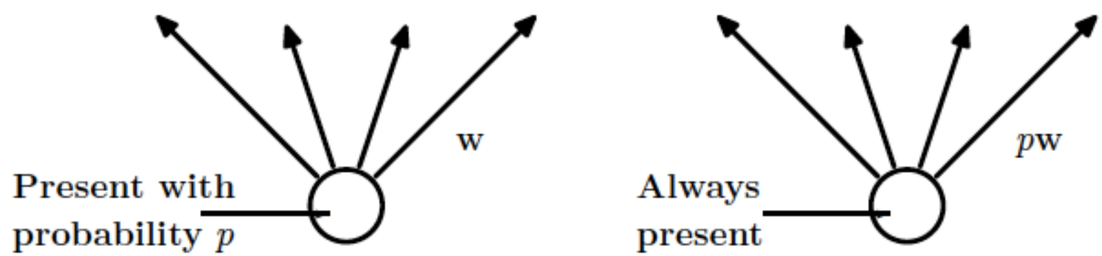
img-source:
https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

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