@DocXavi

Master in Computer Vision Barcelona

[http://pagines.uab.cat/mcv/]

Xavier Giró-i-Nieto

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Department of Signal Theory and Communications
Image Processing Group

Module 4 - Lecture 7

Video Analysis with RNNs

2 February 2017

Acknowledgments

Santi Pascual

Amaia Salvador

Alberto Montes

More details:
D2L2, [“Recurrent Neural Networks I”]
D2L3, [“Recurrent Neural Networks II”]

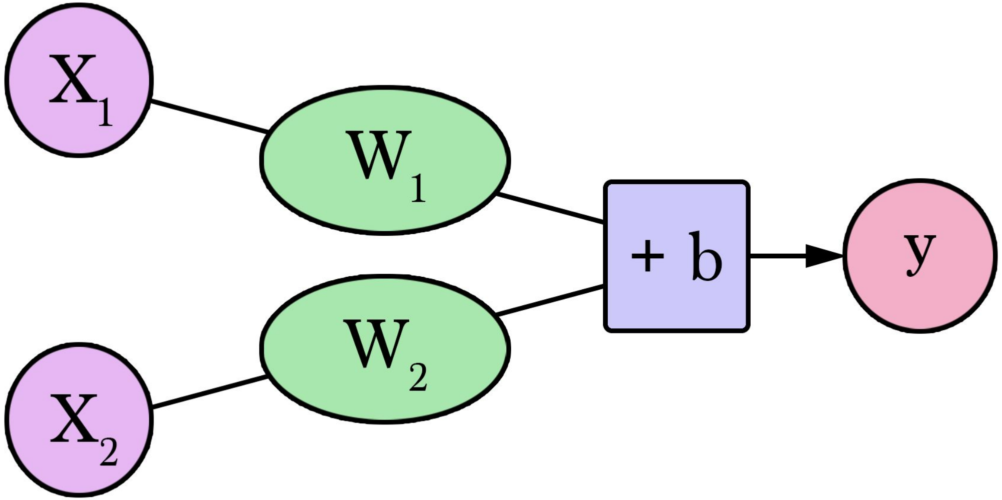
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Linked slides

Outline

1. Recurrent Neural Networks
2. Activity Recognition
3. Object Tracking
4. Speech and Video
5. Learn more

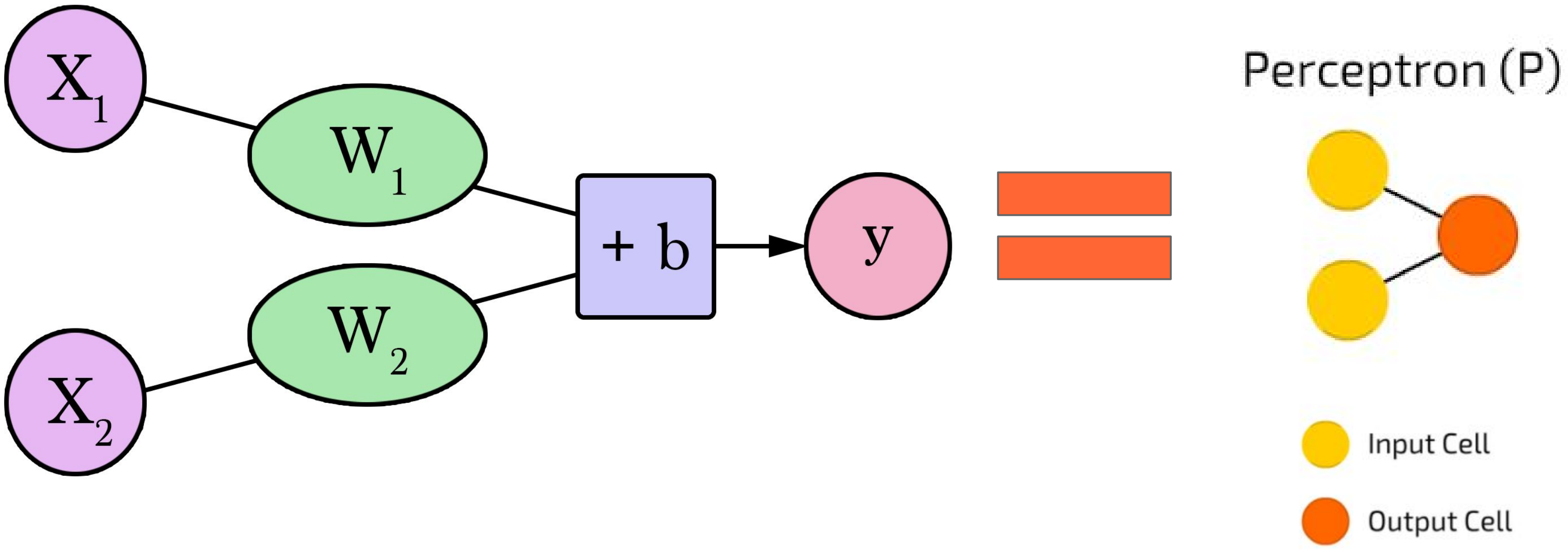
Previously... A Perceptron



$$y = W_1 x_1 + W_2 x_2 + b$$

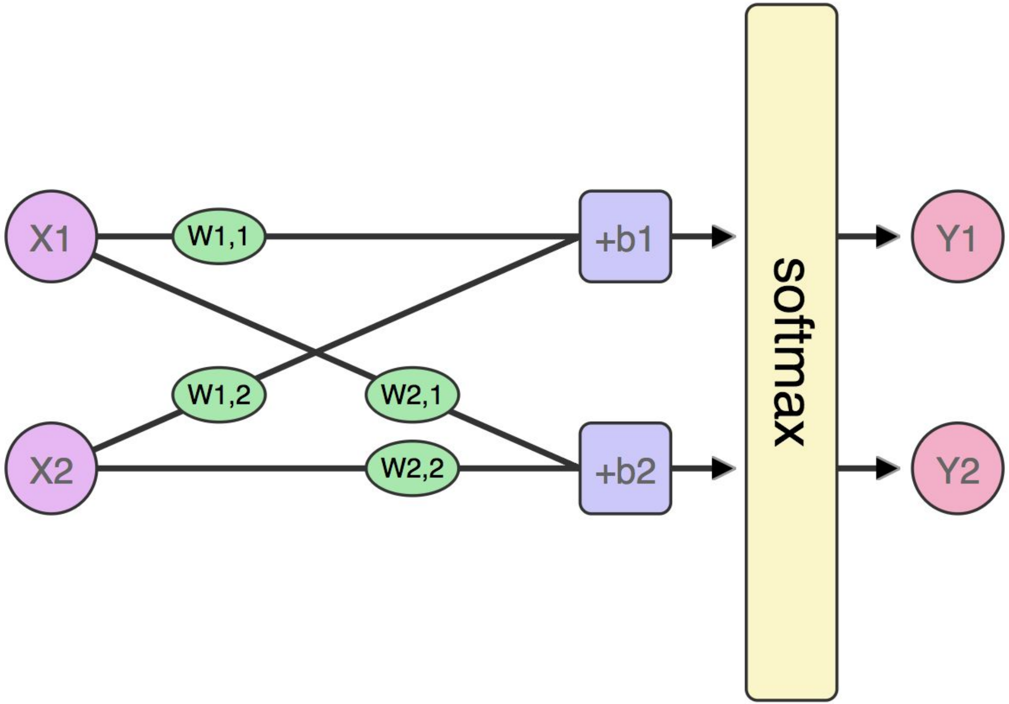
J. Alammar, [“A visual and interactive guide to the Basics of Neural Networks”]() (2016)

Previously... A Perceptron



J. Alammar, "A visual and interactive guide to the Basics of Neural Networks" (2016)
F. Van Veen, "The Neural Network Zoo" (2016)

Two Perceptrons + Softmax classifier

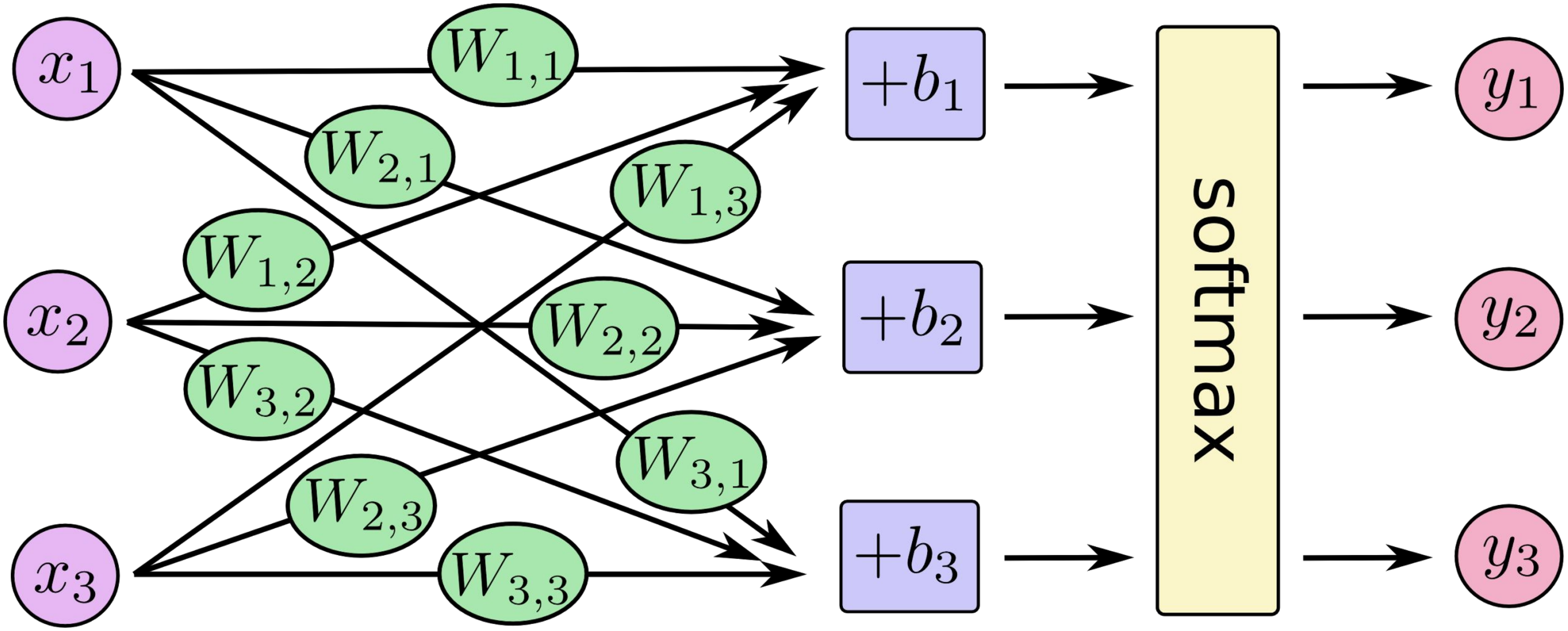


$$\text{evidence}_i = \sum_j W_{i,\,j} x_j + b_i$$

$$\text{softmax}(x)_i = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

J. Alammar, [“A visual and interactive guide to the Basics of Neural Networks”] (2016)

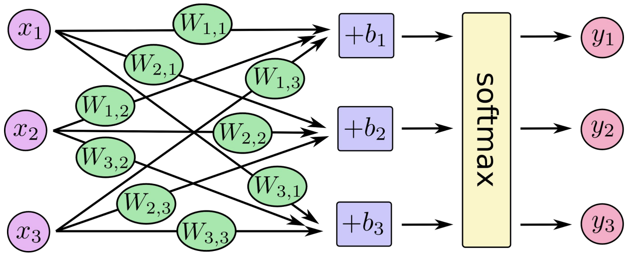
Three perceptrons + Softmax classifier



TensorFlow, [“MNIST for ML beginners”]

Three perceptrons + Softmax classifier

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \text{softmax} \begin{pmatrix} W_{1,1}x_1 + W_{1,2}x_2 + W_{1,3}x_3 + b_1 \\ W_{2,1}x_1 + W_{2,2}x_2 + W_{2,3}x_3 + b_2 \\ W_{3,1}x_1 + W_{3,2}x_2 + W_{3,3}x_3 + b_3 \end{pmatrix}$$

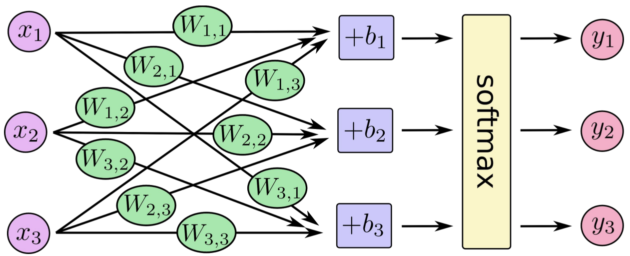


TensorFlow, “MNIST for ML beginners”

Three perceptrons + Softmax classifier

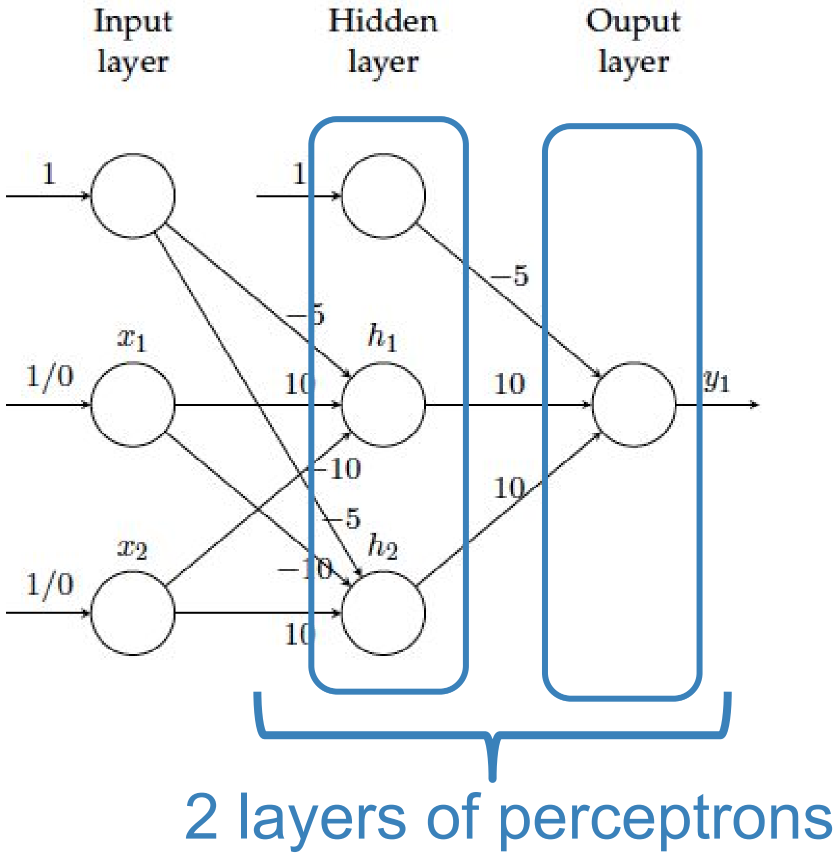
$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \text{softmax}\left(\begin{bmatrix} W_{1,1} & W_{1,2} & W_{1,3} \\ W_{2,1} & W_{2,2} & W_{2,3} \\ W_{3,1} & W_{3,2} & W_{3,3} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \right)$$

$$y = \text{softmax}(Wx + b)$$

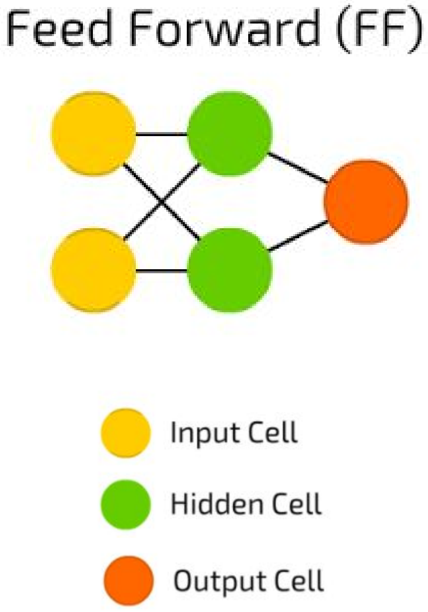


TensorFlow, [“MNIST for ML beginners”]

Neural Network = Multi Layer Perceptron

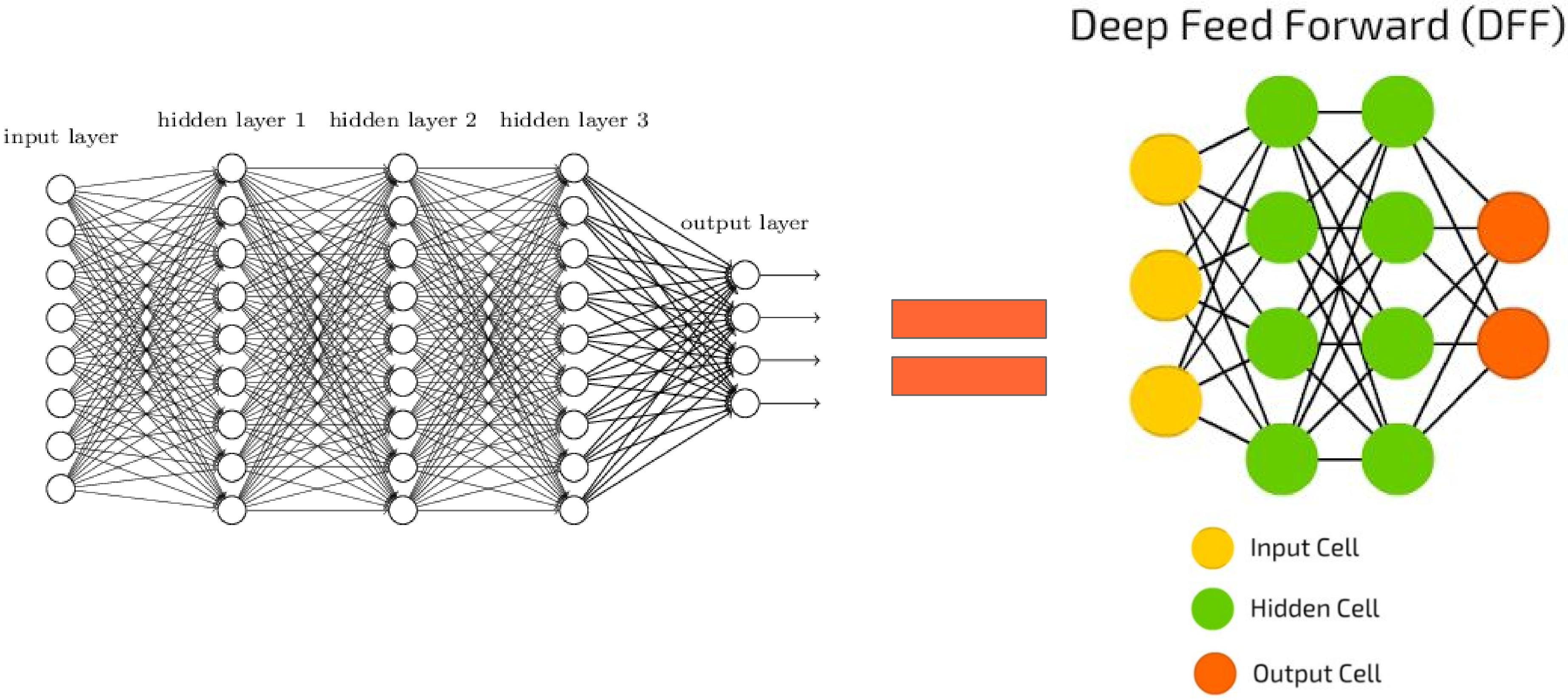


=



F. Van Veen, [“The Neural Network Zoo”](#) (2016)

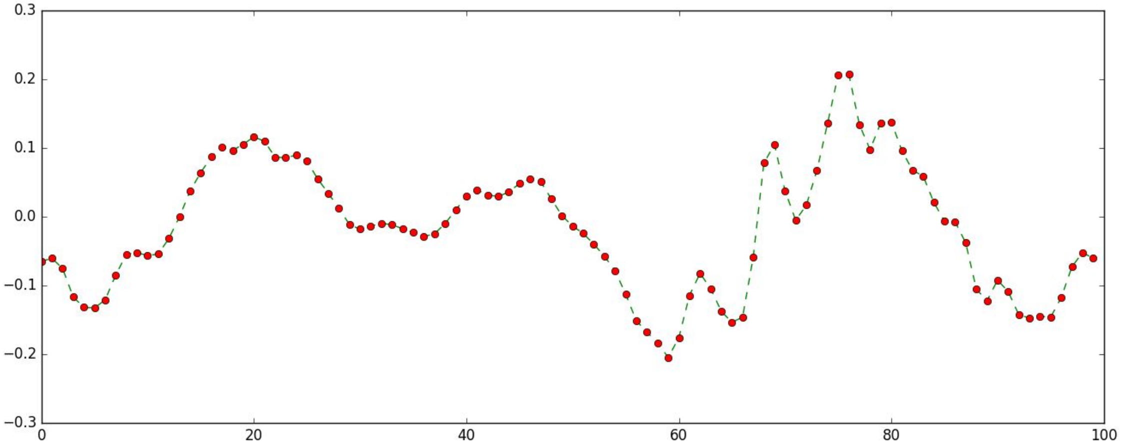
Deep Feed Forward (DFF)



F. Van Veen, [“The Neural Network Zoo”]() (2016)

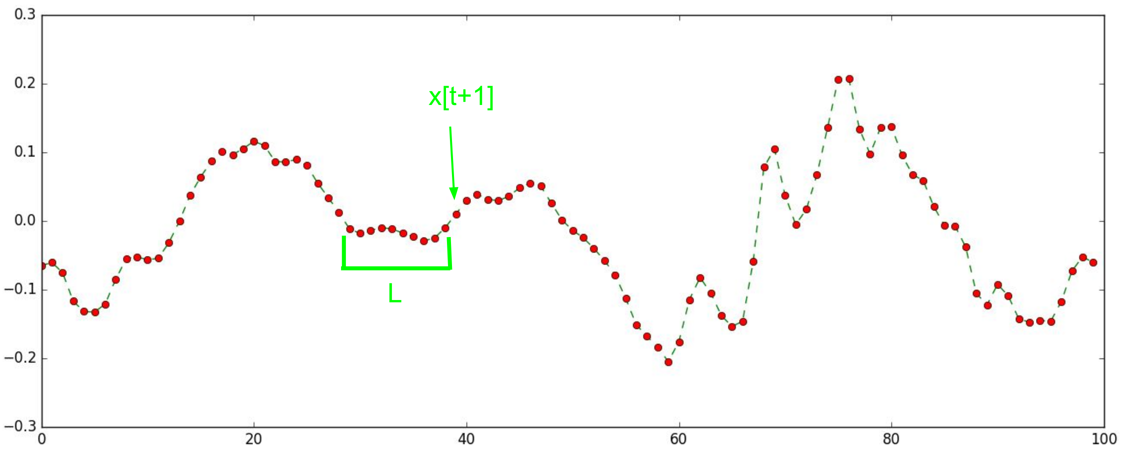
Deep Feed Forward (DFF)

If we have a sequence of samples...



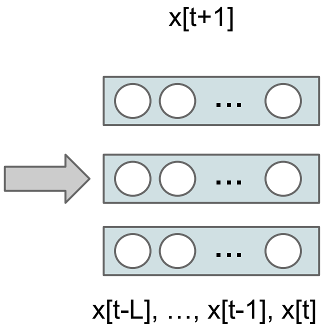
predict sample x[t+1] knowing previous values {x[t], x[t-1], x[t-2], …, x[t-τ]}

Deep Feed Forward (DFF)



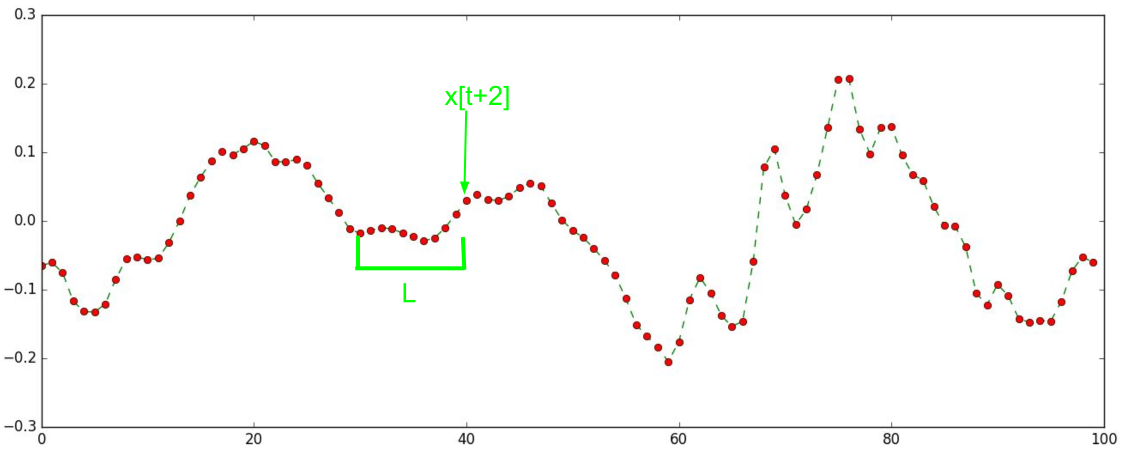
Feed Forward approach:

- static window of size L
- slide the window time-step wise



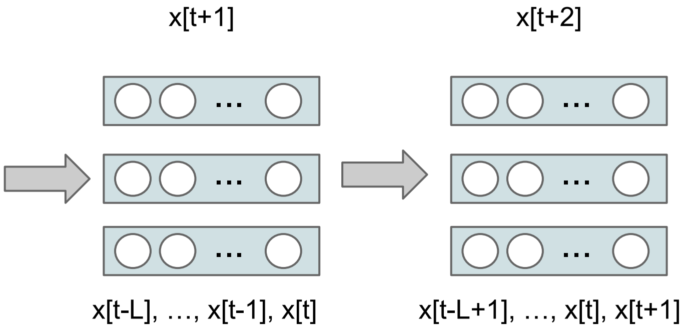
<u>Slide credit</u>: Santi Pascual

Deep Feed Forward (DFF)



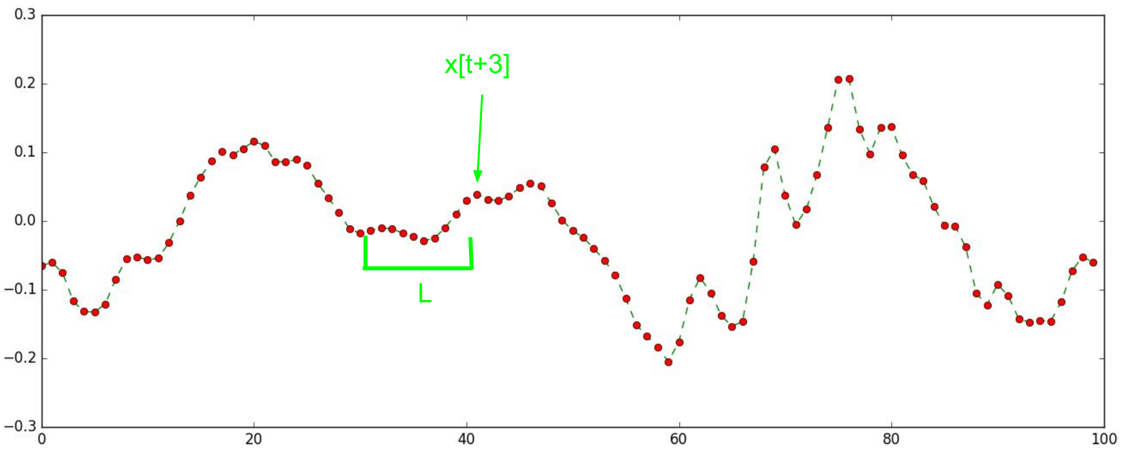
Feed Forward approach:

- static window of size L
- slide the window time-step wise



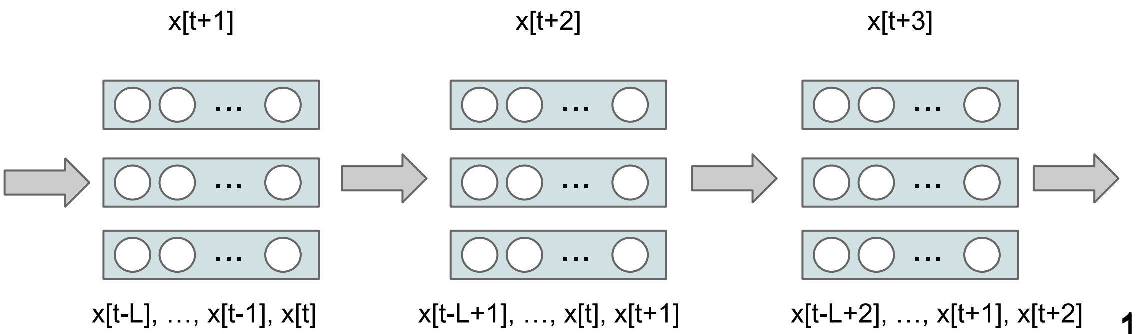
<u>Slide credit</u>: Santi Pascual

Deep Feed Forward (DFF)



Feed Forward approach:

- static window of size L
- slide the window time-step wise

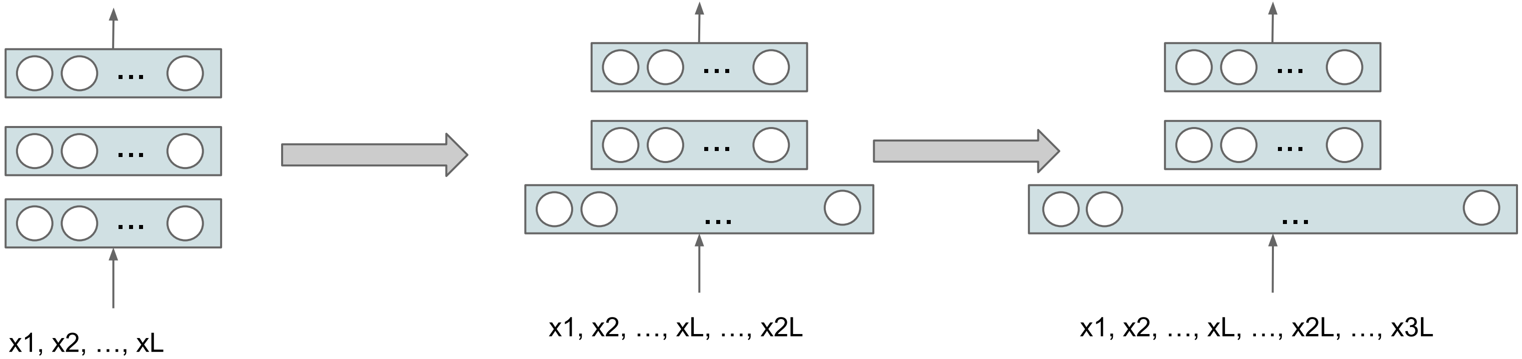


<u>Slide credit</u>: Santi Pascual

Deep Feed Forward (DFF)

Problems for the feed forward + static window approach:

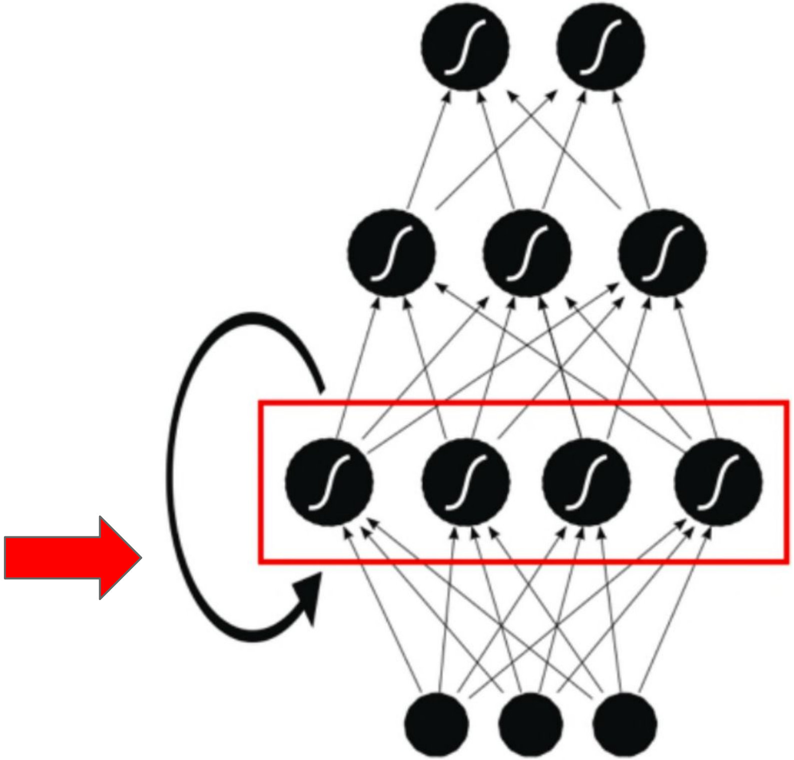
- What's the matter increasing L? → Fast growth of num of parameters!
- Decisions are independent between time-steps!
  - The network doesn't care about what happened at previous time-step, only present window matters → doesn't look good
- Cumbersome padding when there are not enough samples to fill L size
  - Can't work with variable sequence lengths



Slide credit: Santi Pascual

Recurrent Neural Network (RNN)

The hidden layers and the output depend from previous states of the hidden layers



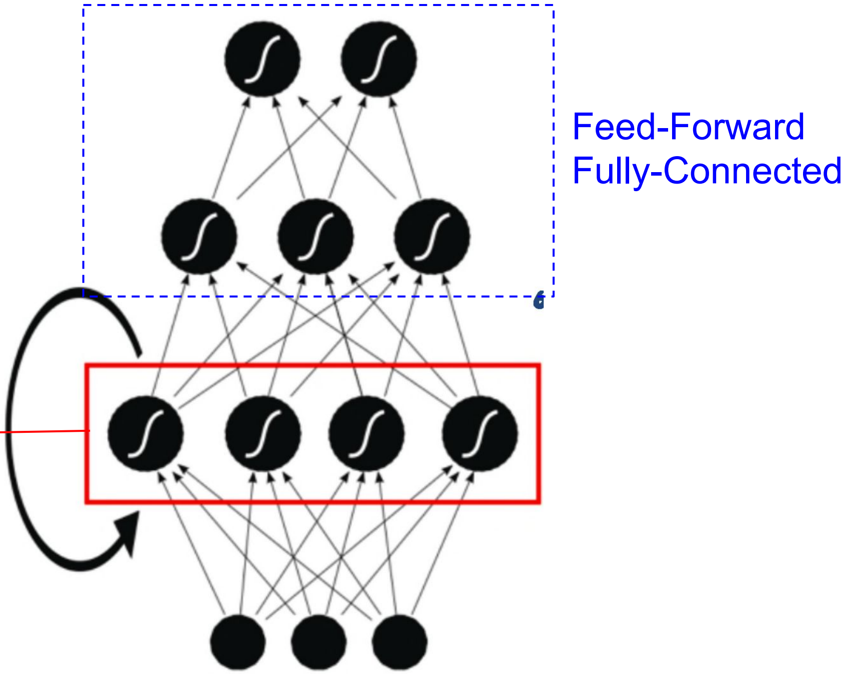
Alex Graves, [“Supervised Sequence Labelling with Recurrent Neural Networks”]

Recurrent Neural Network (RNN)

Solution: Build specific connections capturing the temporal evolution → **Shared weights in time**

- Give volatile memory to the network

$$\boxed{h_t = f(W \cdot x_t + U \cdot h_{t-1} + b)}$$

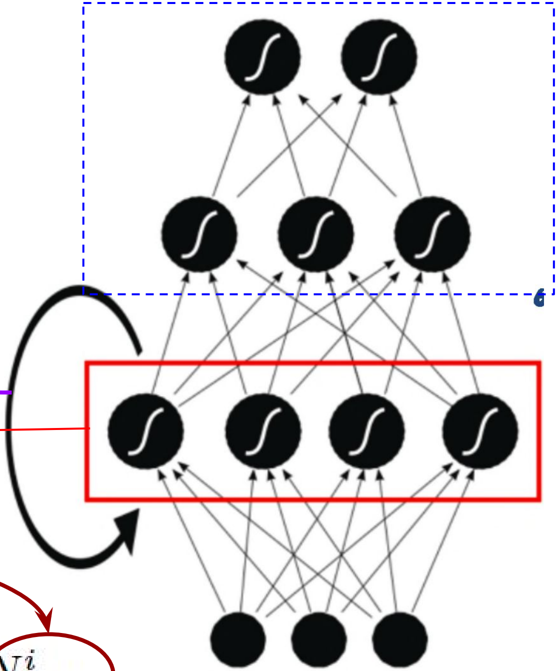


Recurrent Neural Network (RNN)

Solution: Build specific connections capturing the temporal evolution → **Shared weights in time**

- Give volatile memory to the network

Fully connected in time: recurrent matrix



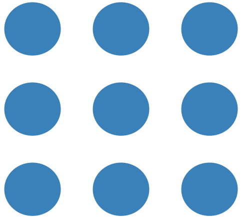
Feed-Forward Fully-Connected

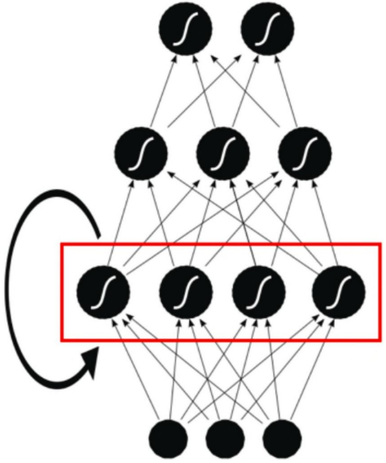
$$h_t = f(W \cdot x_t + U \cdot h_{t-1} + b)$$

$$N^i_{params} = N^i_{inputs} \times N^i_{units} + N^i_{units} \times N^i_{units} + N^i_{units}$$

Recurrent Neural Network (RNN)

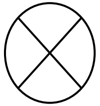
Front View





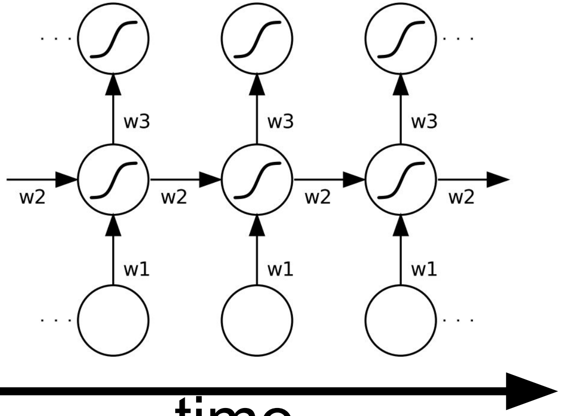
Rotation 90°

time



Rotation 90°

Side View

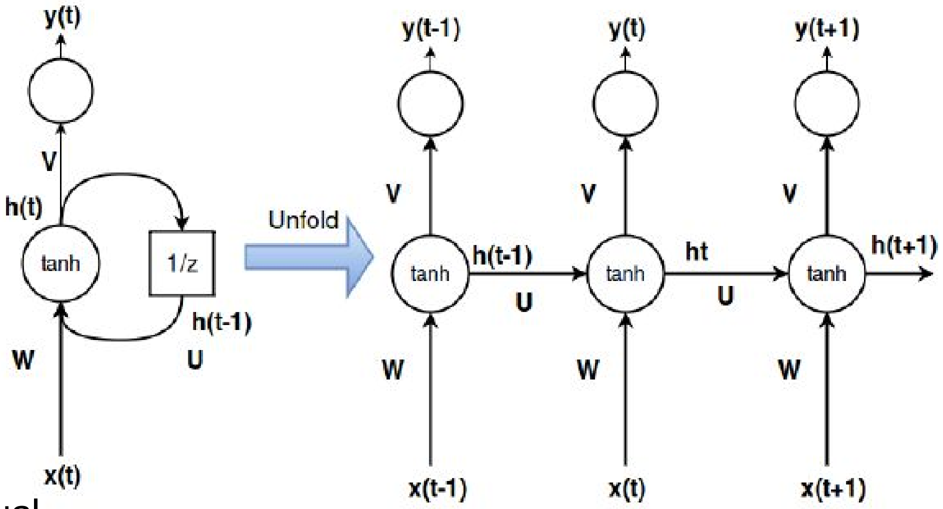


time

Recurrent Neural Network (RNN)

Hence we have two data flows: **Forward in space + time** propagation: **2 projections per layer activation.**

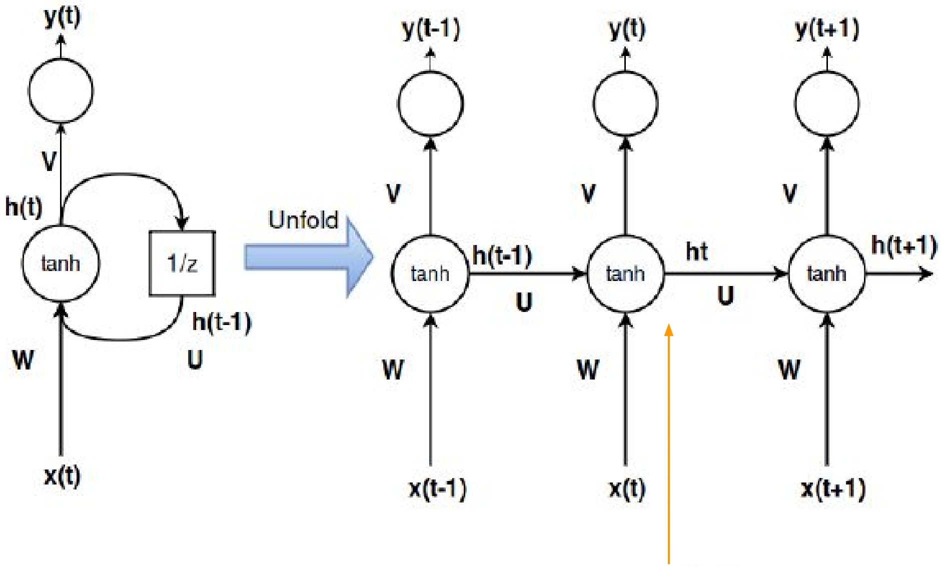
BEWARE: We have extra depth now! Every time-step is an extra level of depth (as a deeper stack of layers in a feed-forward fashion!)



Recurrent Neural Network (RNN)

Hence we have two data flows: **Forward in space + time** propagation: **2 projections per layer activation**

- Last time-step includes the context of our decisions recursively



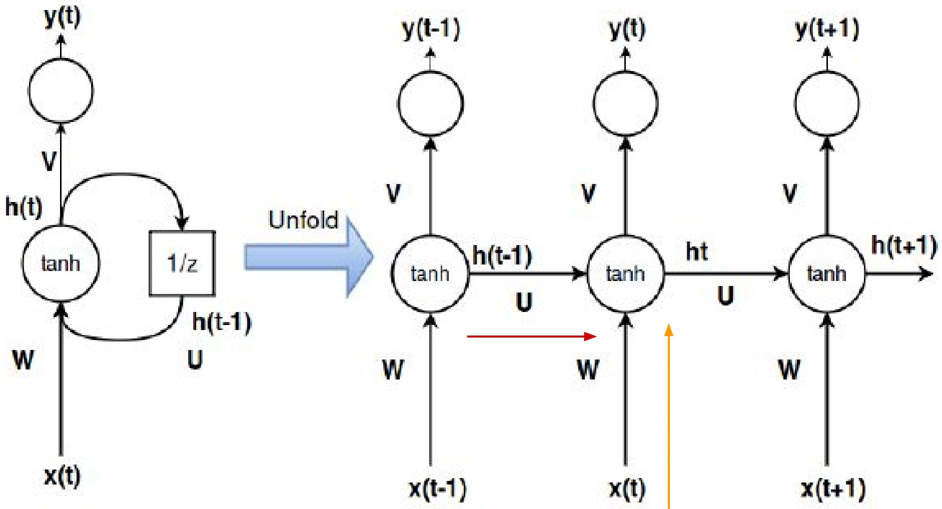
$$\mathbf{h}_t = g(\mathbf{W} \cdot \mathbf{x}_t + \mathbf{U} \cdot \mathbf{h}_{t-1} + \mathbf{b}_h)$$

$\mathbf{x}_t \in \mathbb{R}^n$

Recurrent Neural Network (RNN)

Hence we have two data flows: **Forward in space + time** propagation: **2 projections per layer activation**

- Last time-step includes the context of our decisions recursively



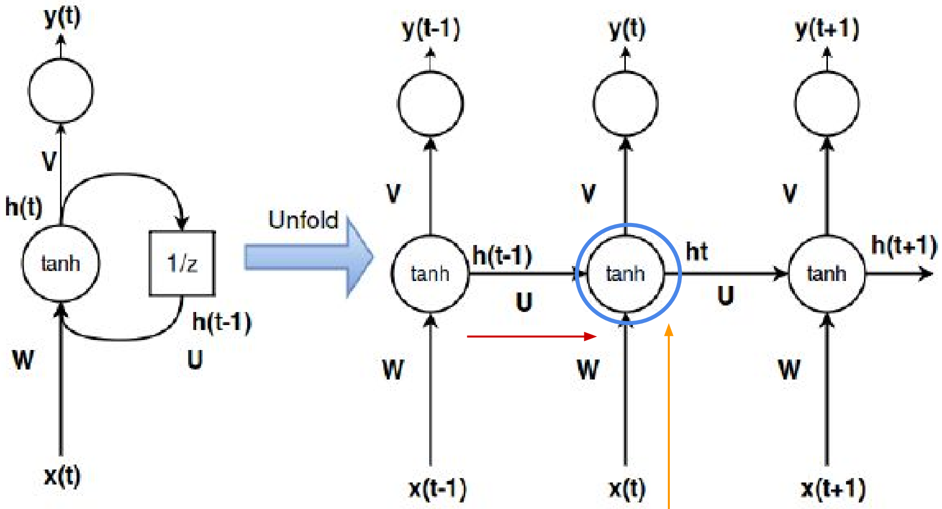
$$\mathbf{h}_t = g(\mathbf{W} \cdot \mathbf{x}_t + \mathbf{U} \cdot \mathbf{h}_{t-1} + \mathbf{b}_h)$$

$\mathbf{x}_t \in \mathbb{R}^n$

Recurrent Neural Network (RNN)

Hence we have two data flows: **Forward in space + time** propagation: **2 projections per layer activation**

- Last time-step includes the context of our decisions recursively



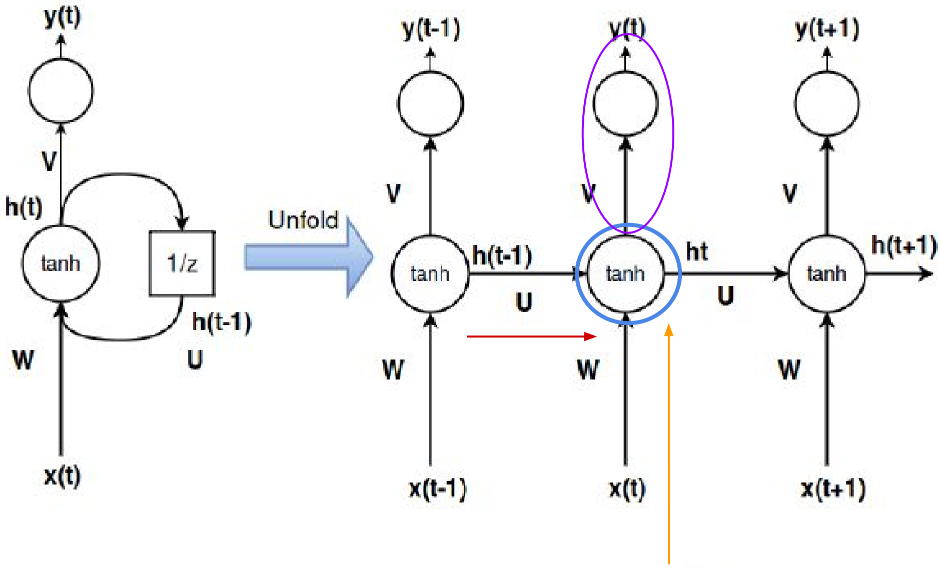
$$\mathbf{h}_t = g(\mathbf{W} \cdot \mathbf{x}_t + \mathbf{U} \cdot \mathbf{h}_{t-1} + \mathbf{b}_h)$$

$\mathbf{x}_t \in \mathbb{R}^n$

Recurrent Neural Network (RNN)

Hence we have two data flows: **Forward in space + time** propagation: **2 projections per layer activation**

- Last time-step includes the context of our decisions recursively



$\mathbf{x}_t \in \mathbb{R}^n$

$$\mathbf{y}_t = f(\mathbf{V} \cdot \mathbf{h}_t + \mathbf{b}_v)$$

$$\mathbf{h}_t = g(\mathbf{W} \cdot \mathbf{x}_t + \mathbf{U} \cdot \mathbf{h}_{t-1} + \mathbf{b}_h)$$

Recurrent Neural Network (RNN)

Back Propagation Through Time (BPTT): The training method has to take into account the time operations → a cost function ***E*** is defined to train our RNN, and in this case the total error at the output of the network is the sum of the errors at each time-step:

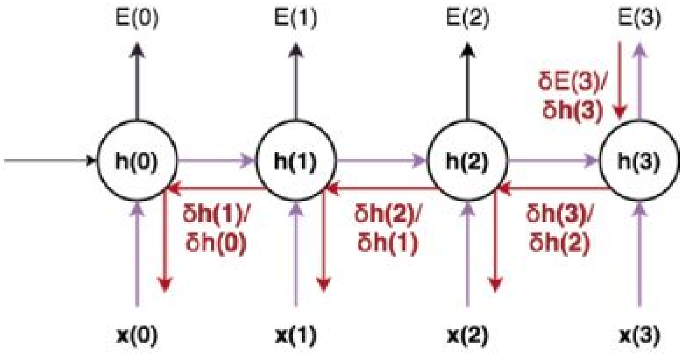
$$E(\mathbf{y}, \hat{\mathbf{y}}) = \sum_{t=1}^{T} E_t(\mathbf{y}_t, \hat{\mathbf{y}}_t)$$

$$\frac{\partial E}{\partial \mathbf{W}} = \sum_{t=0}^{T-1} \frac{\partial E_t}{\partial \mathbf{W}}$$

T: **max amount of time-steps to do back-prop**. In Keras this is specified when defining the "input shape" to the RNN layer, by means of:
(batch size, sequence length (T), input_dim)

Input shape

3D tensor with shape `(nb_samples, timesteps, input_dim)`.



Example back-prop in time with 3 time-steps

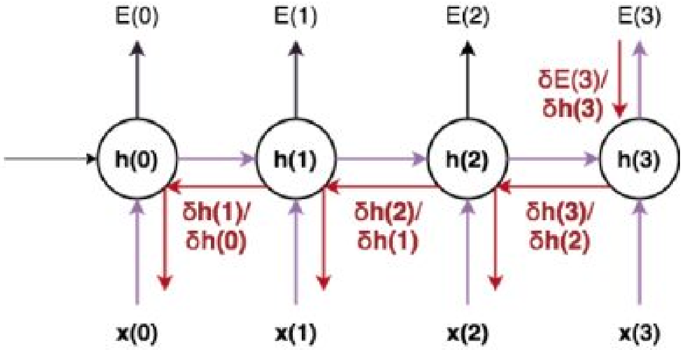
Recurrent Neural Network (RNN)

Main problems:

- **Long-term memory** (remembering quite far time-steps) **vanishes quickly** because of the recursive operation with **U**

$$\mathbf{h}_t = g(\mathbf{W} \cdot \mathbf{x}_t + \mathbf{U} \cdot g(\cdots g(\mathbf{W} \cdot \mathbf{x}_{t-T} + \mathbf{U} \cdot \mathbf{h}_{t-T} + \mathbf{b}_h) \cdots) + \mathbf{b}_h)$$

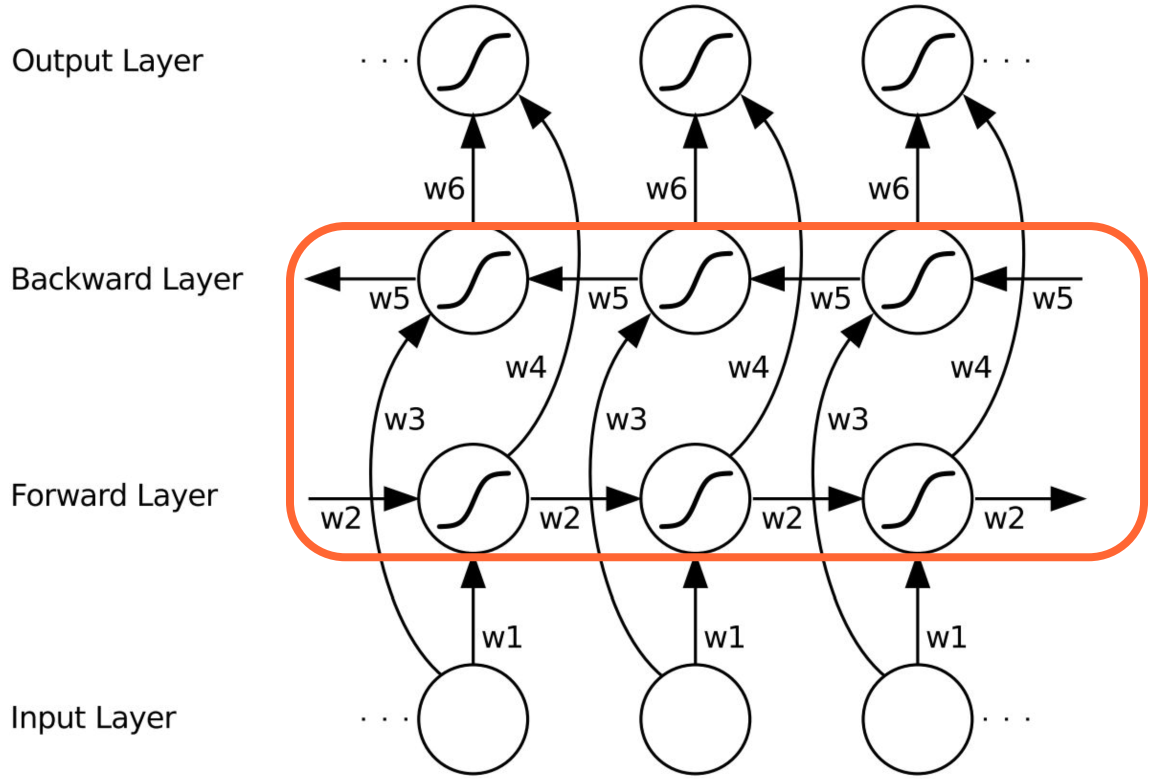
- **During training gradients explode/vanish easily because of depth-in-time** → Exploding/Vanishing gradients!



Example back-prop in time with 3 time-steps

Slide credit: Santi Pascual

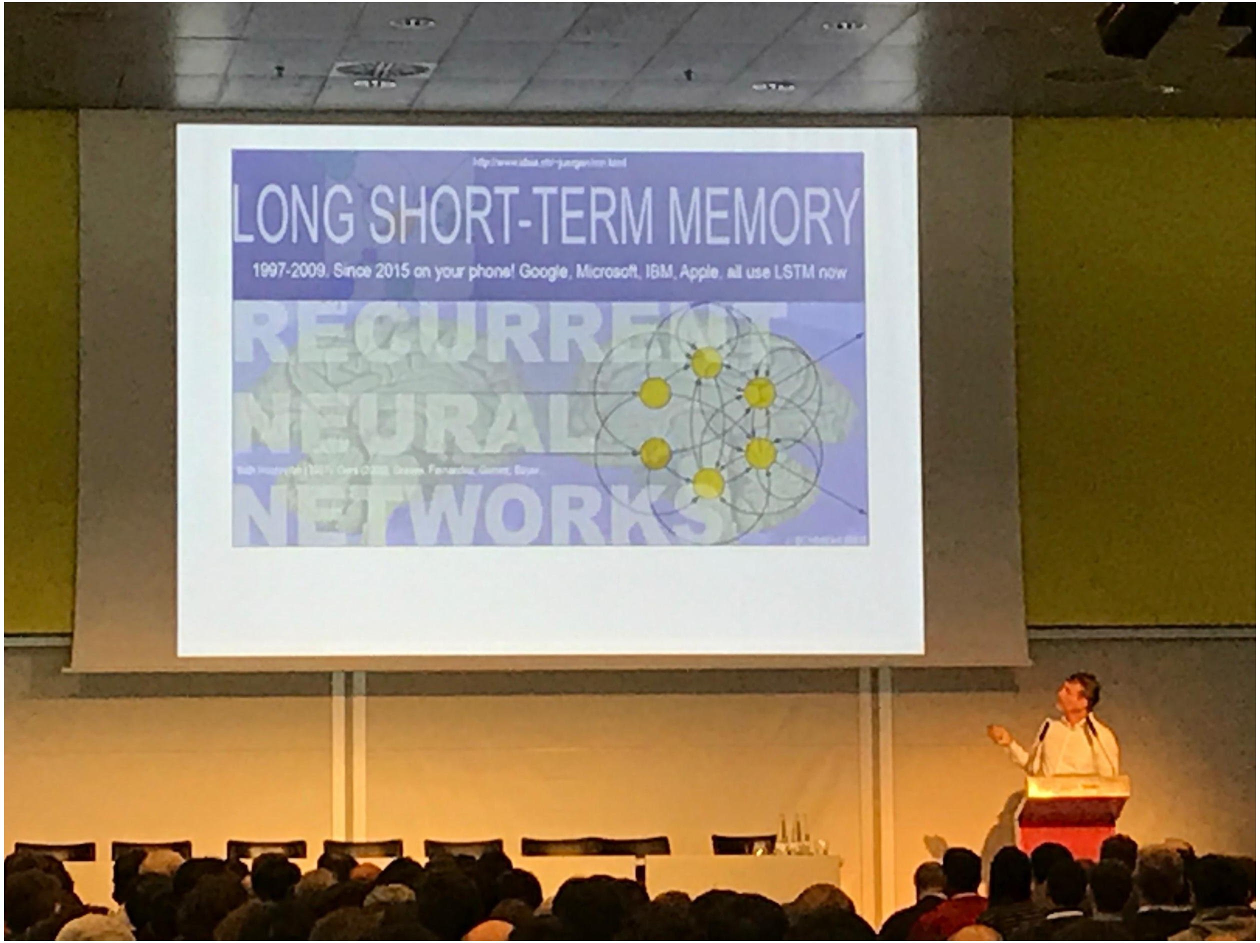
Bidirectional RNN (BRNN)



Must learn weights w2, w3, w4 & w5; in addition to w1 & w6.

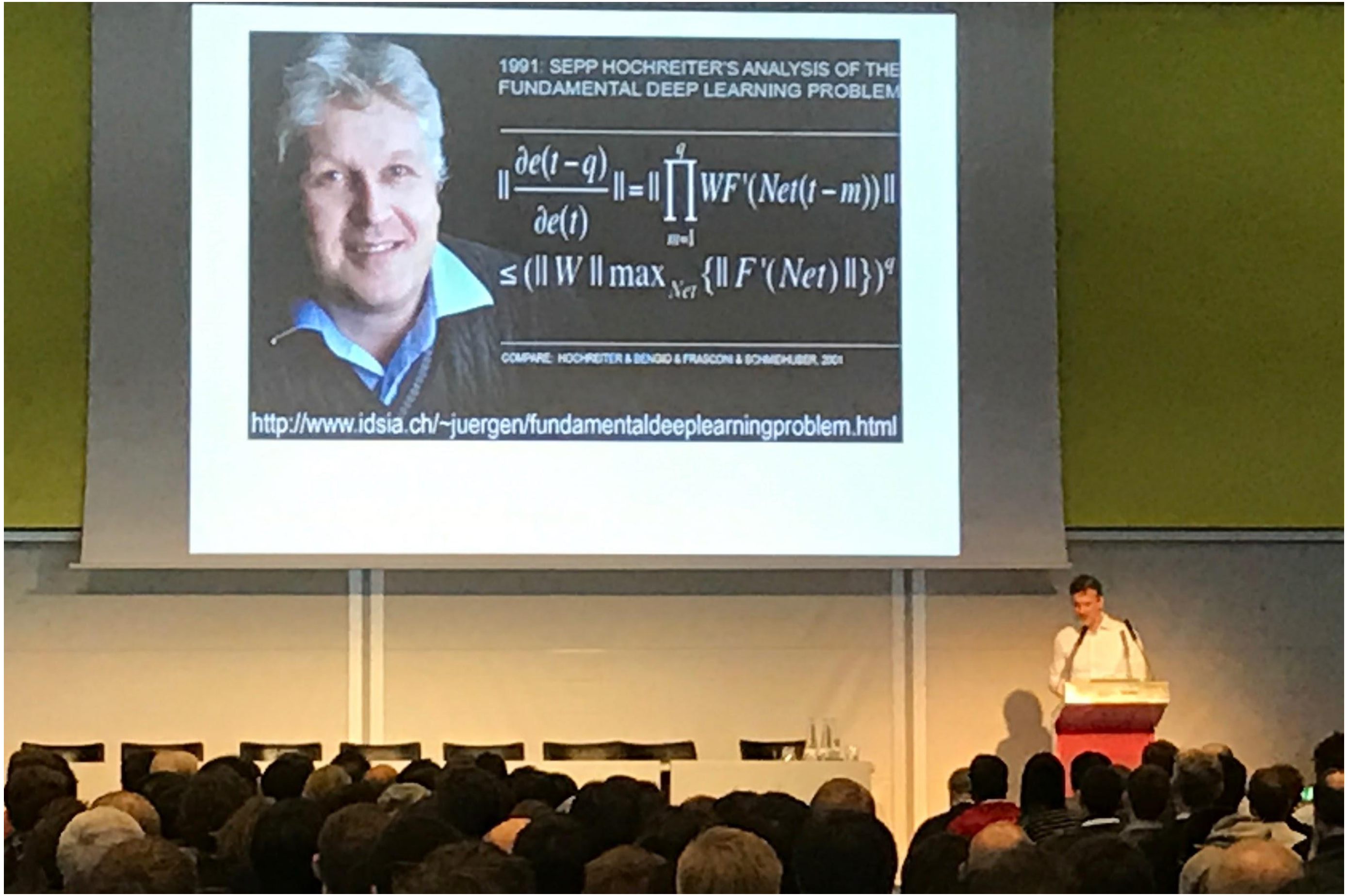
Alex Graves, [“Supervised Sequence Labelling with Recurrent Neural Networks”]

Long Short-Term Memory (LSTM)



Jürgen Schmidhuber @
NIPS 2016 Barcelona

Long Short-Term Memory (LSTM)

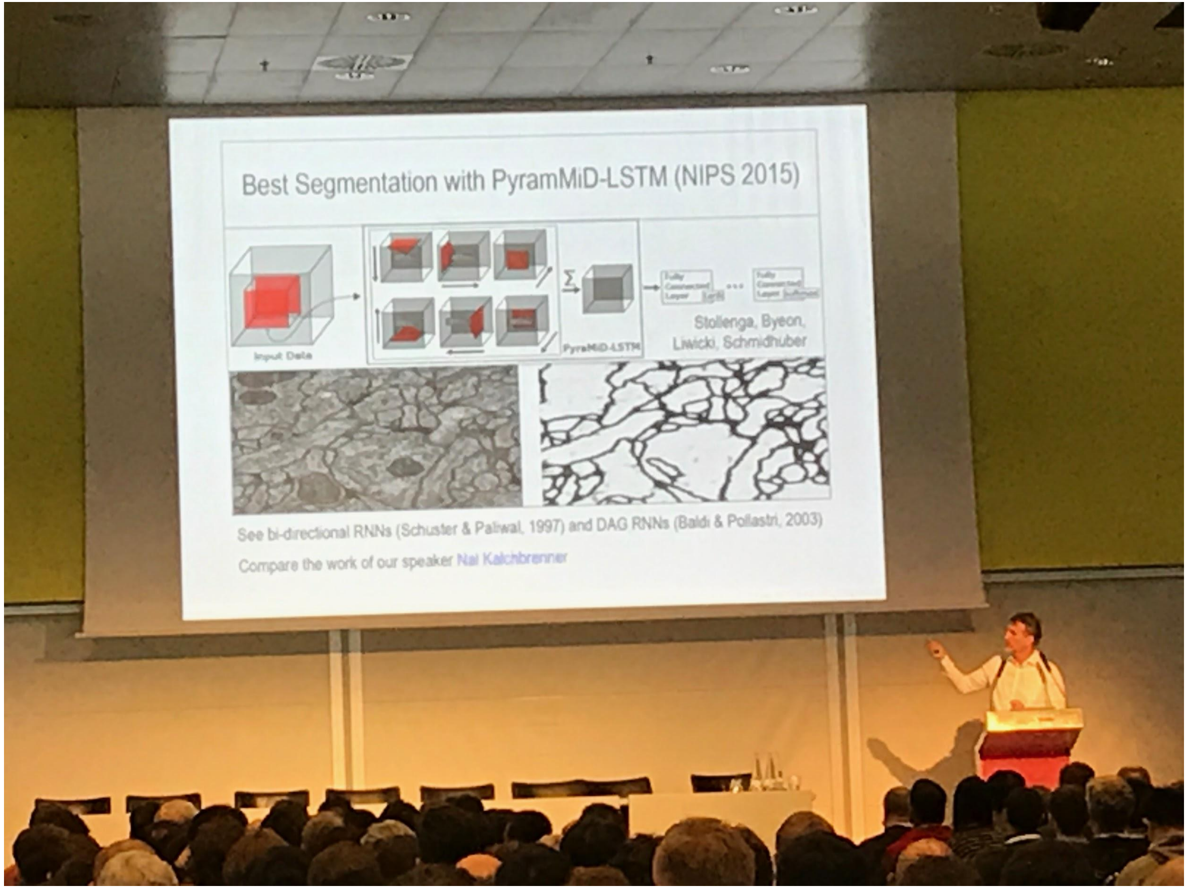


[Jürgen Schmidhuber]() @
NIPS 2016 Barcelona

Long Short-Term Memory (LSTM)

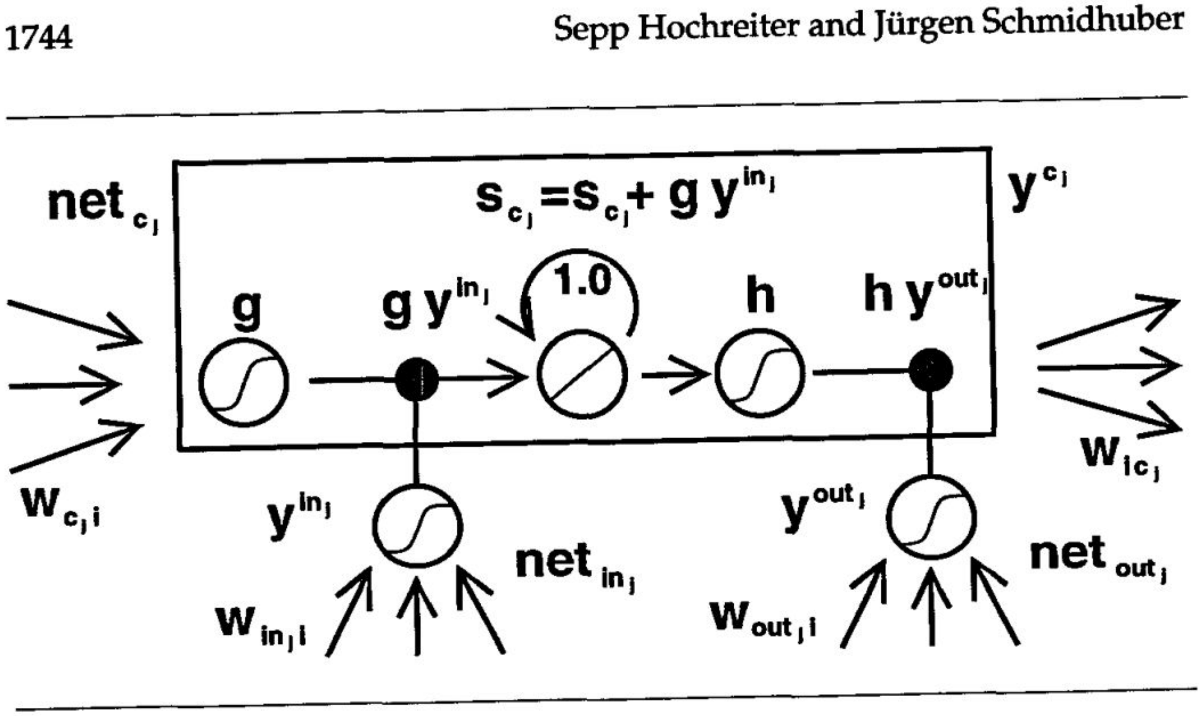
Jürgen Schmidhuber @ NIPS 2016 Barcelona

Long Short-Term Memory (LSTM)



[Jürgen Schmidhuber]() @
NIPS 2016 Barcelona

Long Short-Term Memory (LSTM)



Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9, no. 8 (1997): 1735-1780.

Long Short-Term Memory (LSTM)

Gating method

Solutions:

1. Change the way in which past information is kept → create the notion of **cell state, a memory unit that keeps long-term information in a safer way by protecting it from recursive operations**
2. **Make every RNN unit able to decide whether the current time-step information matters or not**, to accept or discard (optimized reading mechanism)
3. **Make every RNN unit able to forget whatever may not be useful anymore** by clearing that info from the cell state (optimized clearing mechanism)
4. **Make every RNN unit able to output the decisions whenever it is ready to do so** (optimized output mechanism)

Long Short-Term Memory (LSTM)

Gating method

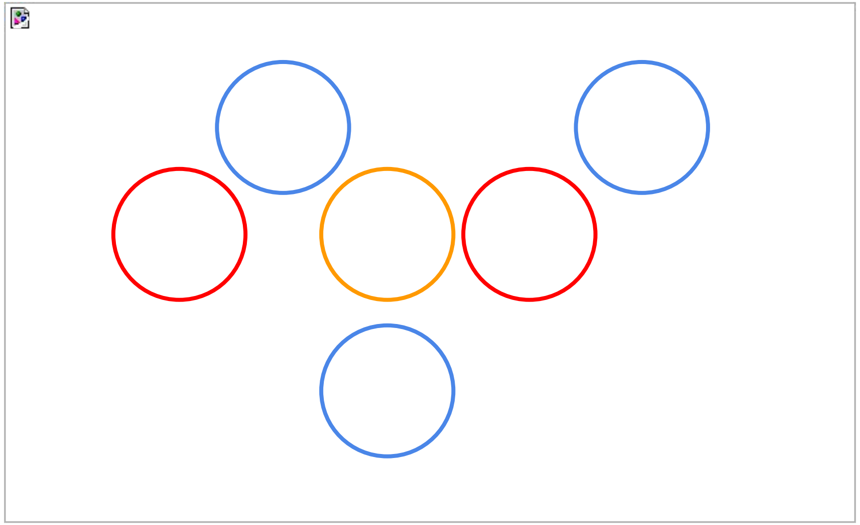
Solutions:

1. Change the way in which past information is kept → create the notion of **cell state, a memory unit that keeps long-term information in a safer way by protecting it from recursive operations**
2. **Make every RNN unit able to decide whether the current time-step information matters or not**, to accept or discard (optimized reading mechanism)
3. **Make every RNN unit able to forget whatever may not be useful anymore** by clearing that info from the cell state (optimized clearing mechanism)
4. **Make every RNN unit able to output the decisions whenever it is ready to do so** (optimized output mechanism)

Long Short-Term Memory (LSTM)

An LSTM cell is defined by two groups of neurons plus the cell state (memory unit):

1. **Gates**
2. **Activation units**
3. **Cell state**



$$\mathbf{i}_t = \sigma(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i)$$

$$\hat{\mathbf{C}}_t = \tanh(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \mathbf{h}_{t-1} + \mathbf{b}_c)$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f)$$

$$\mathbf{C}_t = \mathbf{i}_t \odot \hat{\mathbf{C}}_t + \mathbf{f}_t \odot \mathbf{C}_{t-1}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o)$$

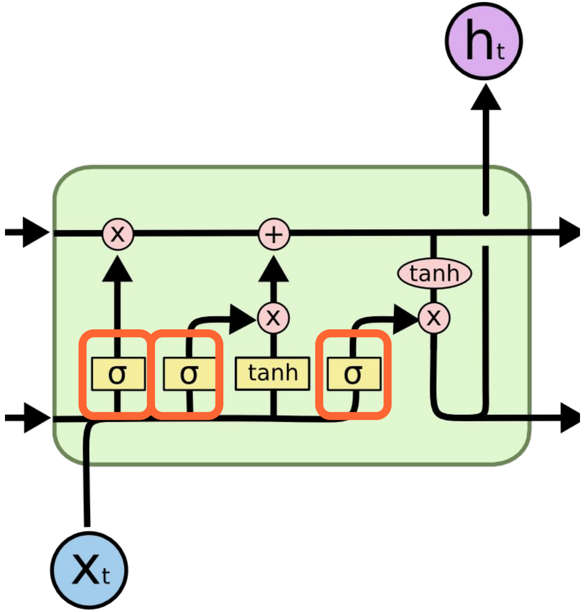
$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{C}_t)$$

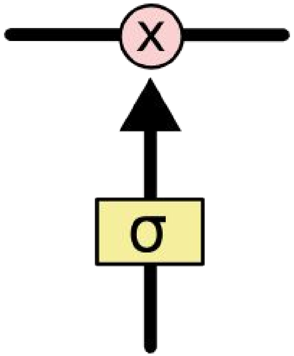
Computation Flow

Slide credit: Santi Pascual

Long Short-Term Memory (LSTM)

Three **gates** are governed by *sigmoid* units (btw [0,1]) define the control of in & out information..





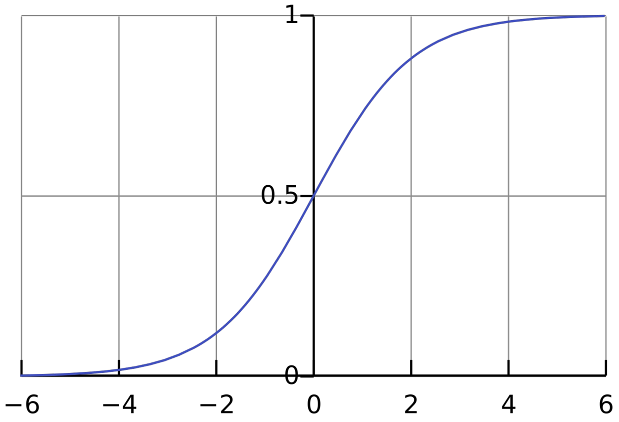
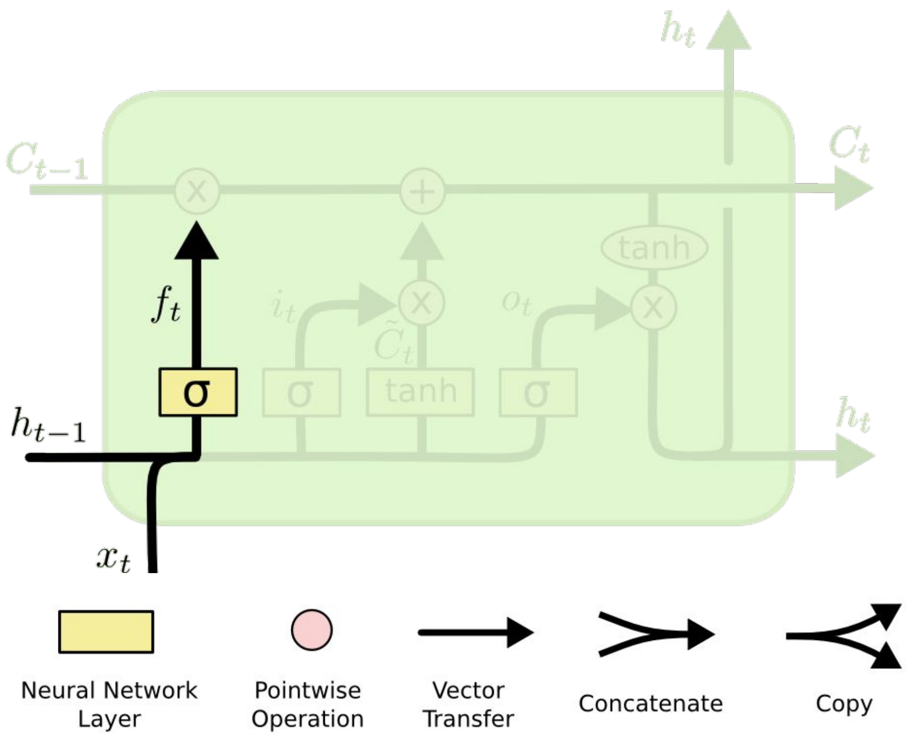


Figure: Cristopher Olah, “Understanding LSTM Networks” (2015)

slide credit: Xavi Giro

Long Short-Term Memory (LSTM)

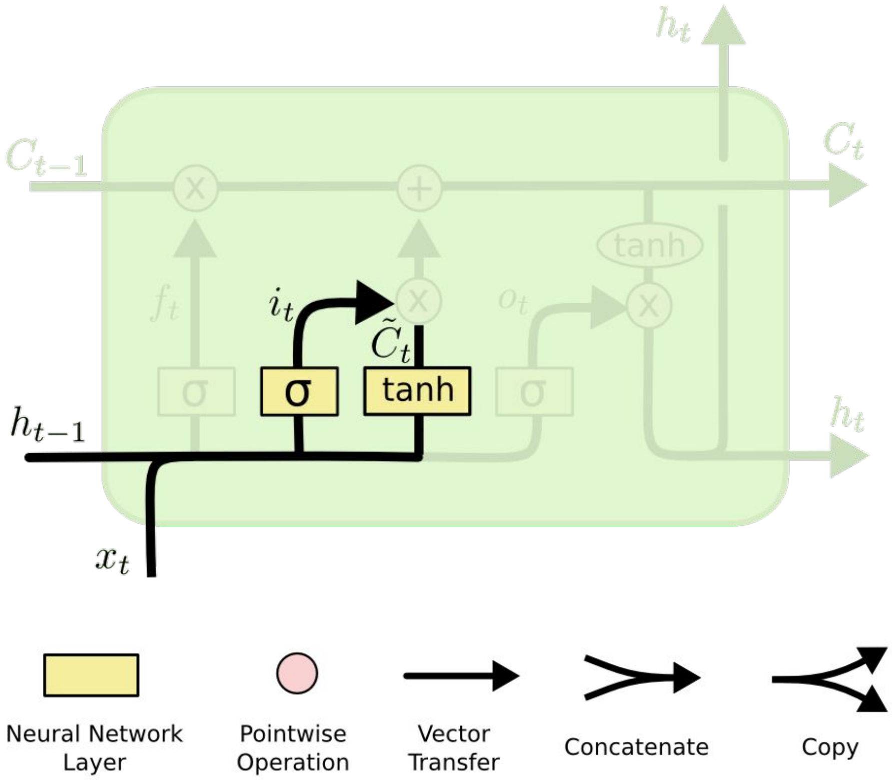


Forget Gate:

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

Figure: Cristopher Olah, "Understanding LSTM Networks" (2015) / Slide: Alberto Montes

Long Short-Term Memory (LSTM)



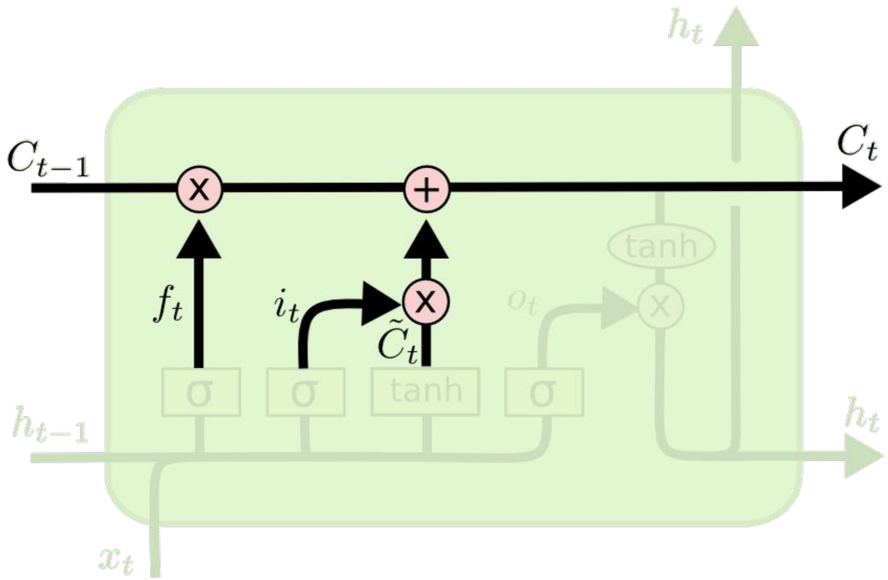
Input Gate Layer

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

New contribution to cell state

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

Long Short-Term Memory (LSTM)

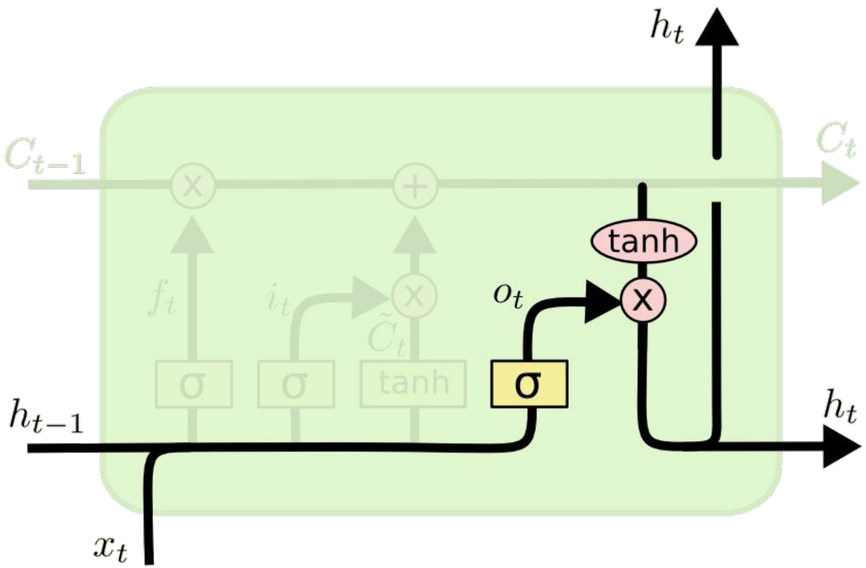


Update Cell State (memory):

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

Figure: Cristopher Olah, "Understanding LSTM Networks" (2015) / Slide: Alberto Montes

Long Short-Term Memory (LSTM)



Output Gate Layer

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

Output to next layer

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

Figure: Cristopher Olah, "Understanding LSTM Networks" (2015) / Slide: Alberto Montes

Long Short-Term Memory (LSTM)

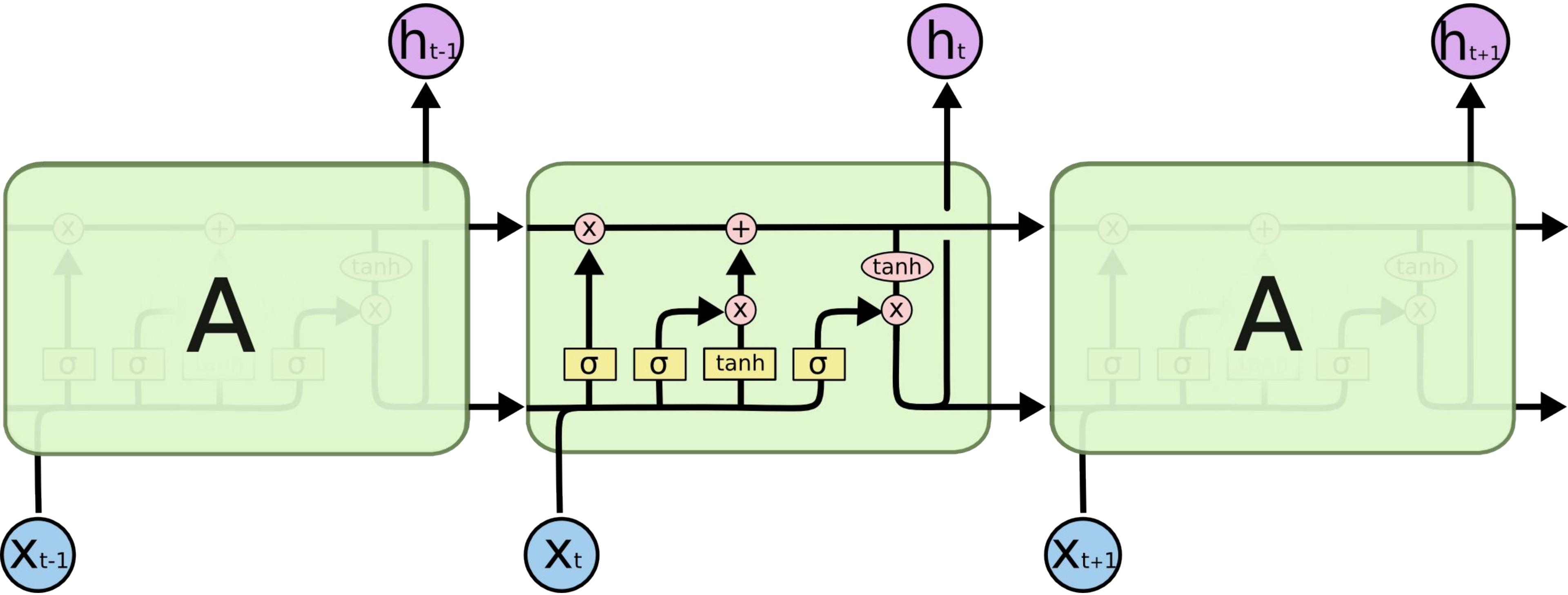
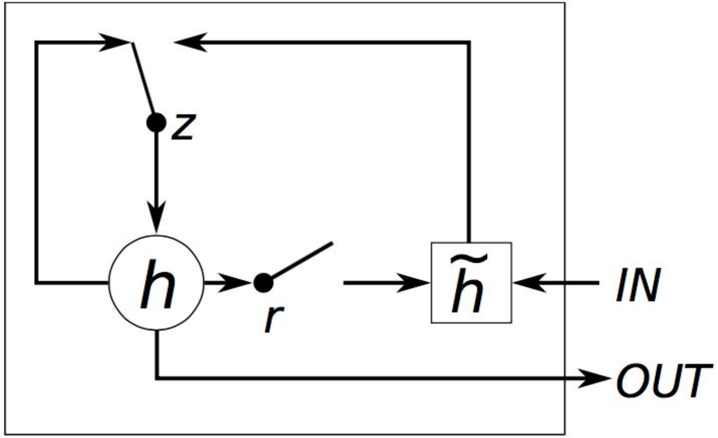


Figure: Cristopher Olah, [“Understanding LSTM Networks”] (2015) / Slide: Alberto Montes

Gated Recurrent Unit (GRU)

Similar performance as LSTM with less computation.



$$u_i = \sigma\left(W^{(u)}x_i + U^{(u)}h_{i-1} + b^{(u)}\right) \quad (1)$$

$$r_i = \sigma\left(W^{(r)}x_i + U^{(r)}h_{i-1} + b^{(r)}\right) \quad (2)$$

$$\tilde{h}_i = \tanh\left(Wx_i + r_i \circ Uh_{i-1} + b^{(h)}\right) \quad (3)$$

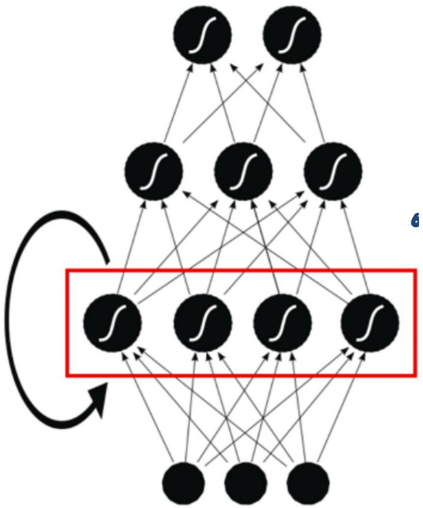
$$h_i = u_i \circ \tilde{h}_i + (1 - u_i) \circ h_{i-1} \quad (4)$$

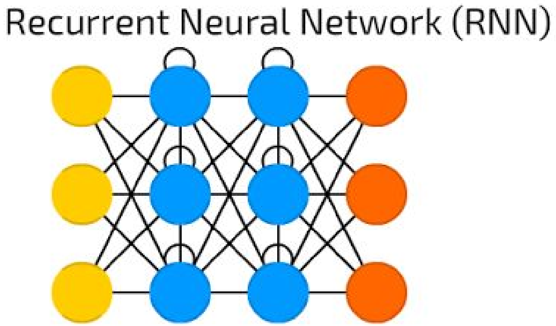
$$N^i_{params} = 3 \times (N^i_{inputs} \times N^i_{units} + N^i_{units} \times N^i_{units} + N^i_{units})$$

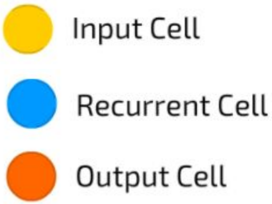
Cho, Kyunghyun, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. ["Learning phrase representations using RNN encoder-decoder for statistical machine translation."]() AMNLP 2014.

slide credit: Xavi Giro

Recurrent Neural Network (RNN)







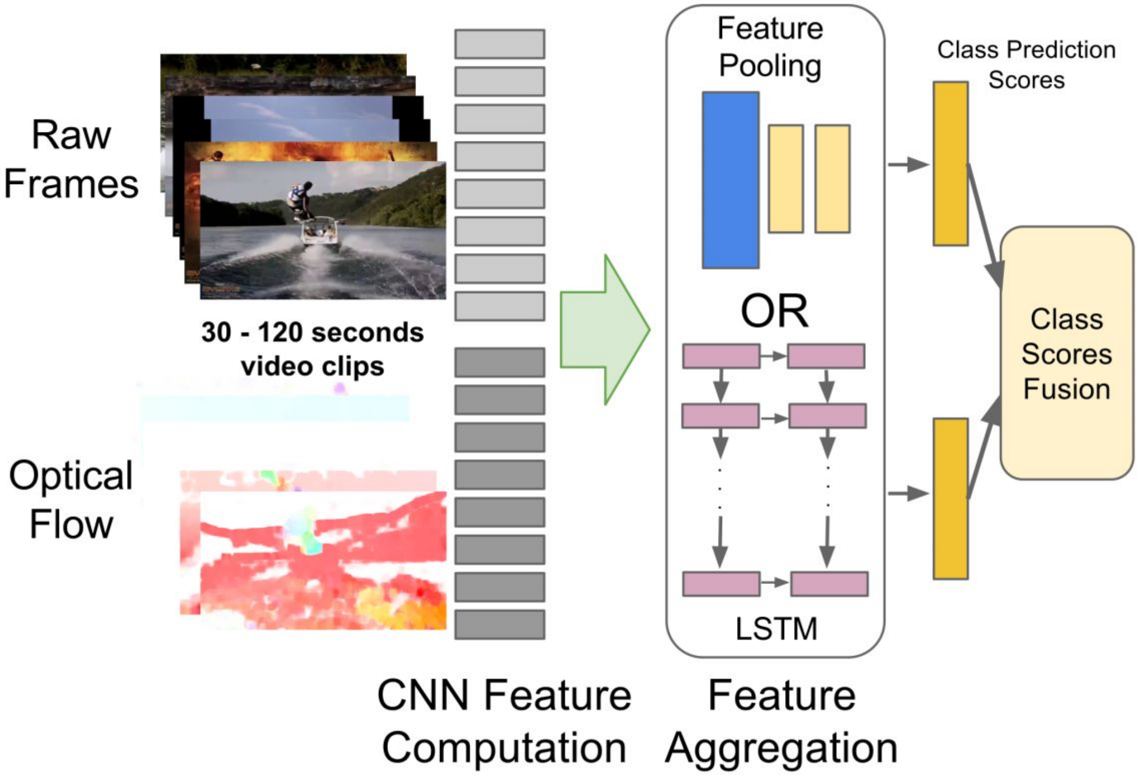
More details:
D2L2, [“Recurrent Neural Networks I”]
D2L3, [“Recurrent Neural Networks II”]

F. Van Veen, “The Neural Network Zoo” (2016)

Outline

1. Recurrent Neural Networks
2. Activity Recognition
3. Object Tracking
4. Speech and Video
5. Learn more

Recognition: Image & Optical Flow CNN + LSTM



Yue-Hei Ng, Joe, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. "Beyond short snippets: Deep networks for video classification." CVPR 2015

Temporal Activity Detection

http://activity-net.org/

Temporal Activity Detection

YouTube Videos

Longboarding

Activity Classification

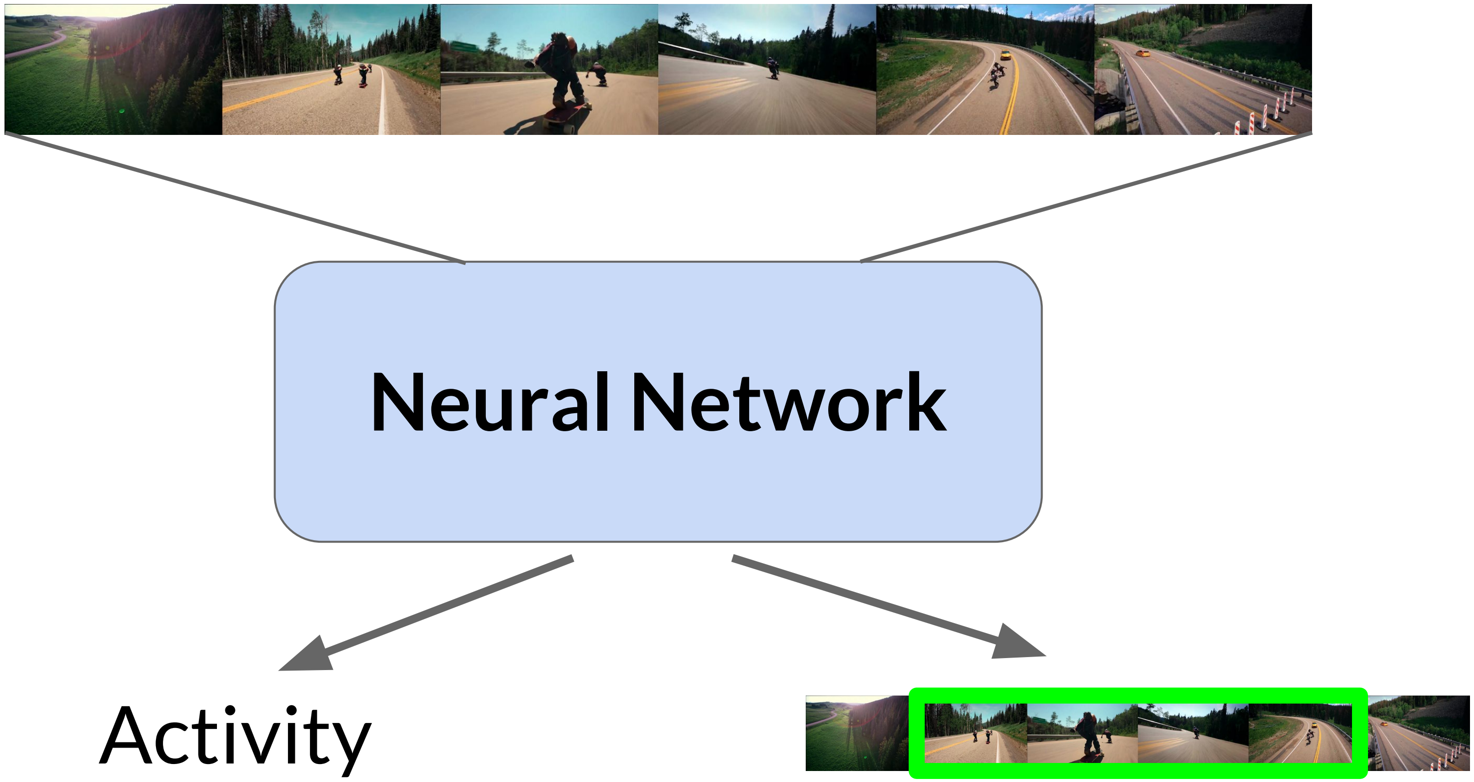
Temporal Activity Detection

Videos

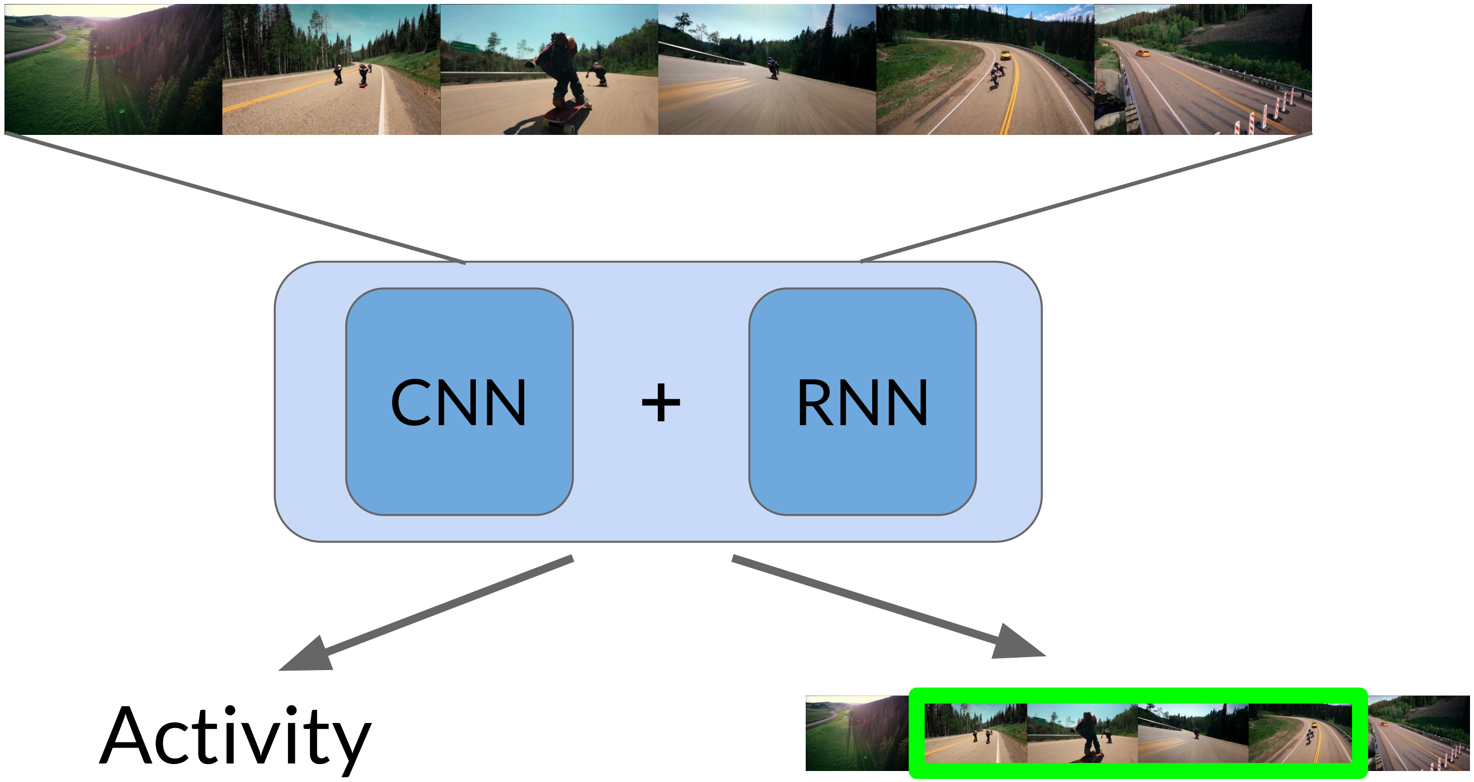
Longboarding

Activity Temporal Localization

Temporal Activity Detection

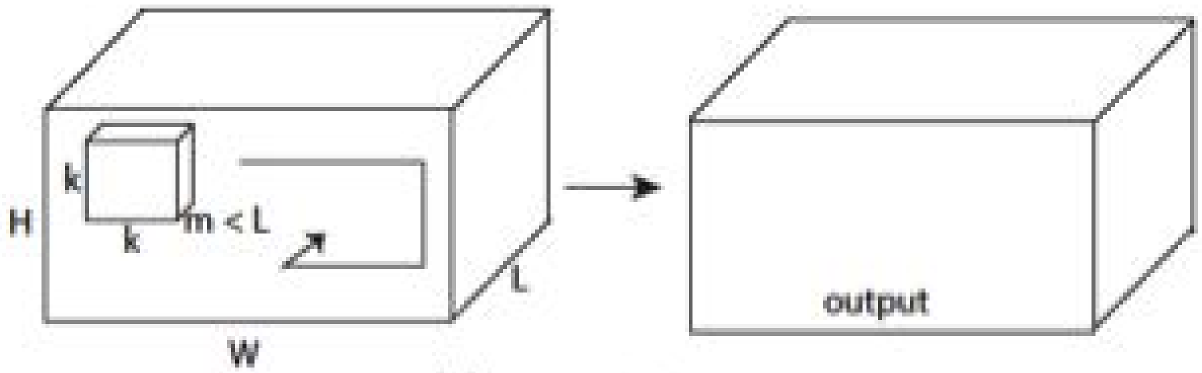


Temporal Activity Detection



Temporal Activity Detection

3D Convolutions over sets of 16 frames...

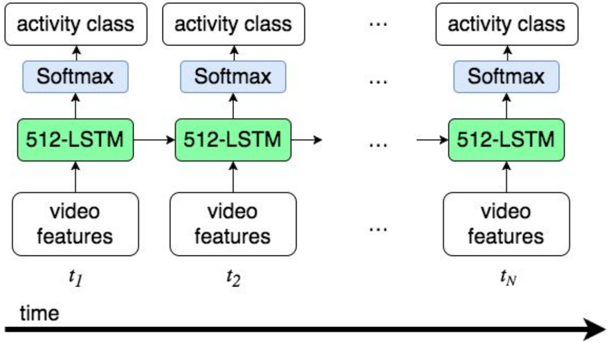


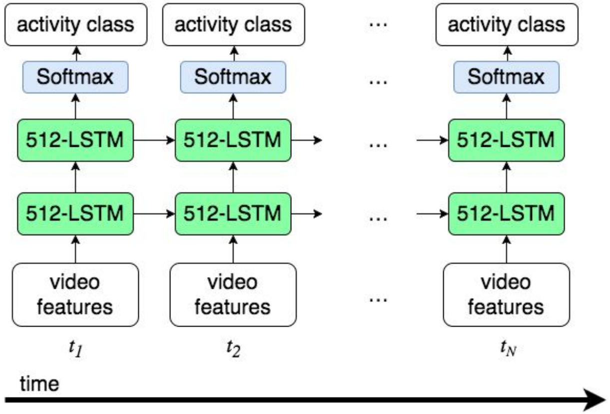
(c) 3D convolution

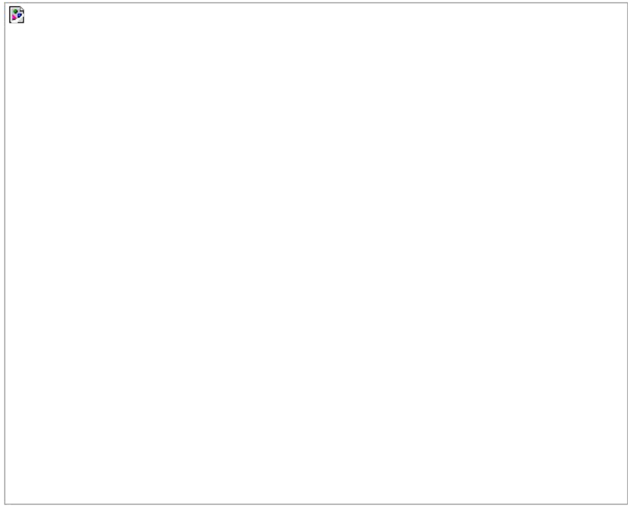
Figure: Tran, Du, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. "Learning spatiotemporal features with 3D convolutional networks." CVPR 2015

Temporal Activity Detection

Temporal Activity Detection







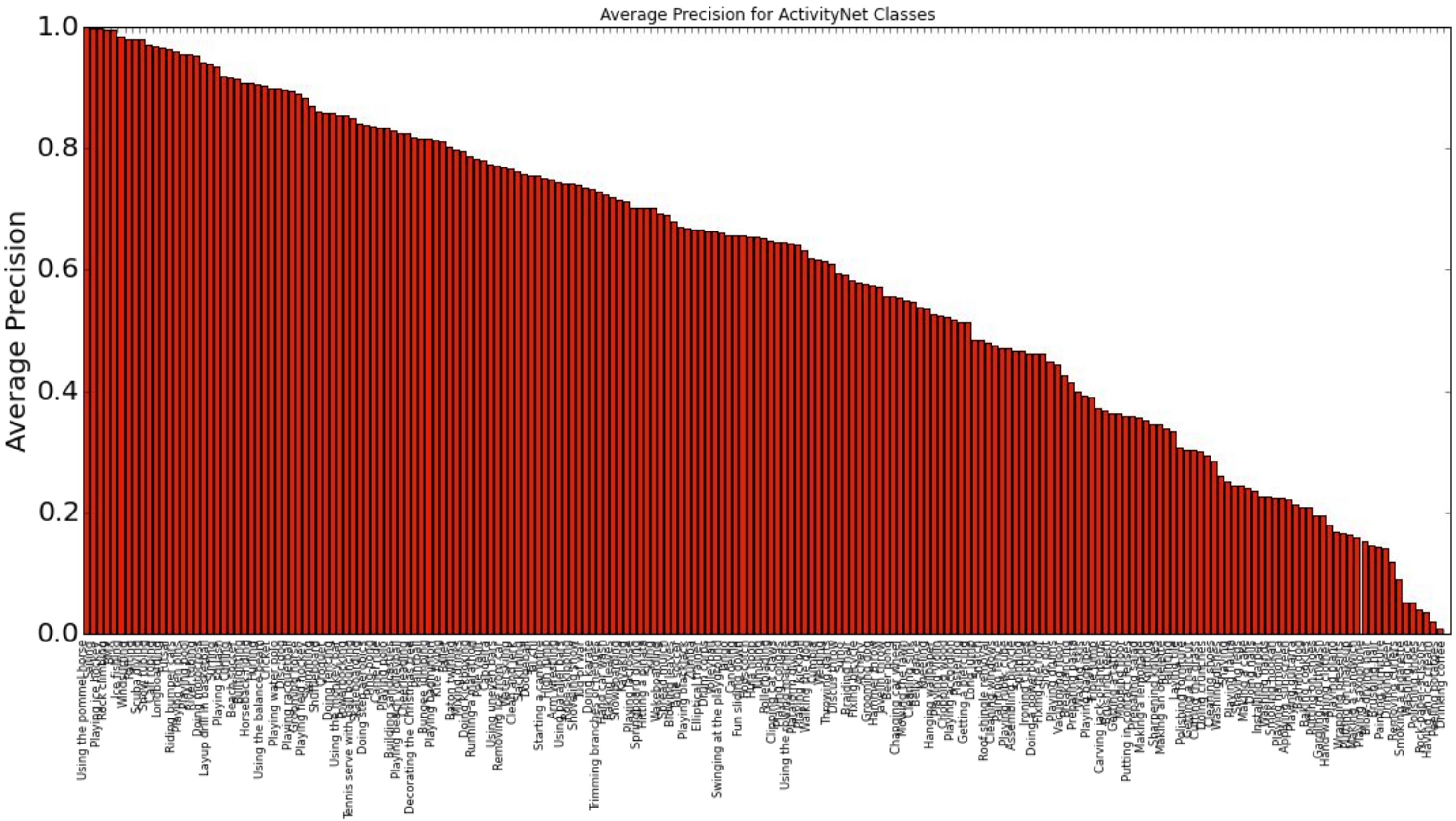
mAP = **0.5938**

mAP = 0.5492

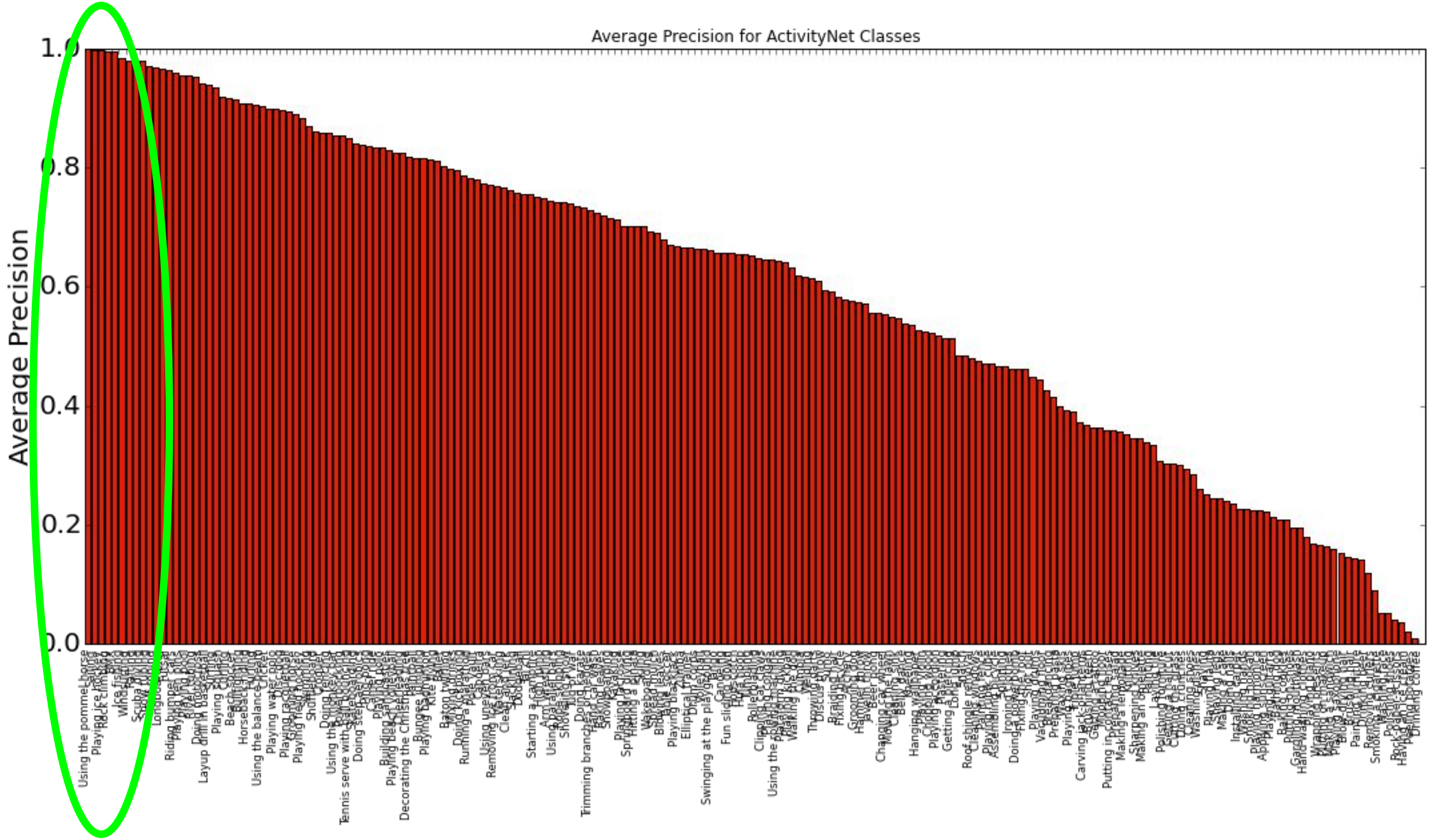
mAP = 0.5635

Deeper networks present overfitting

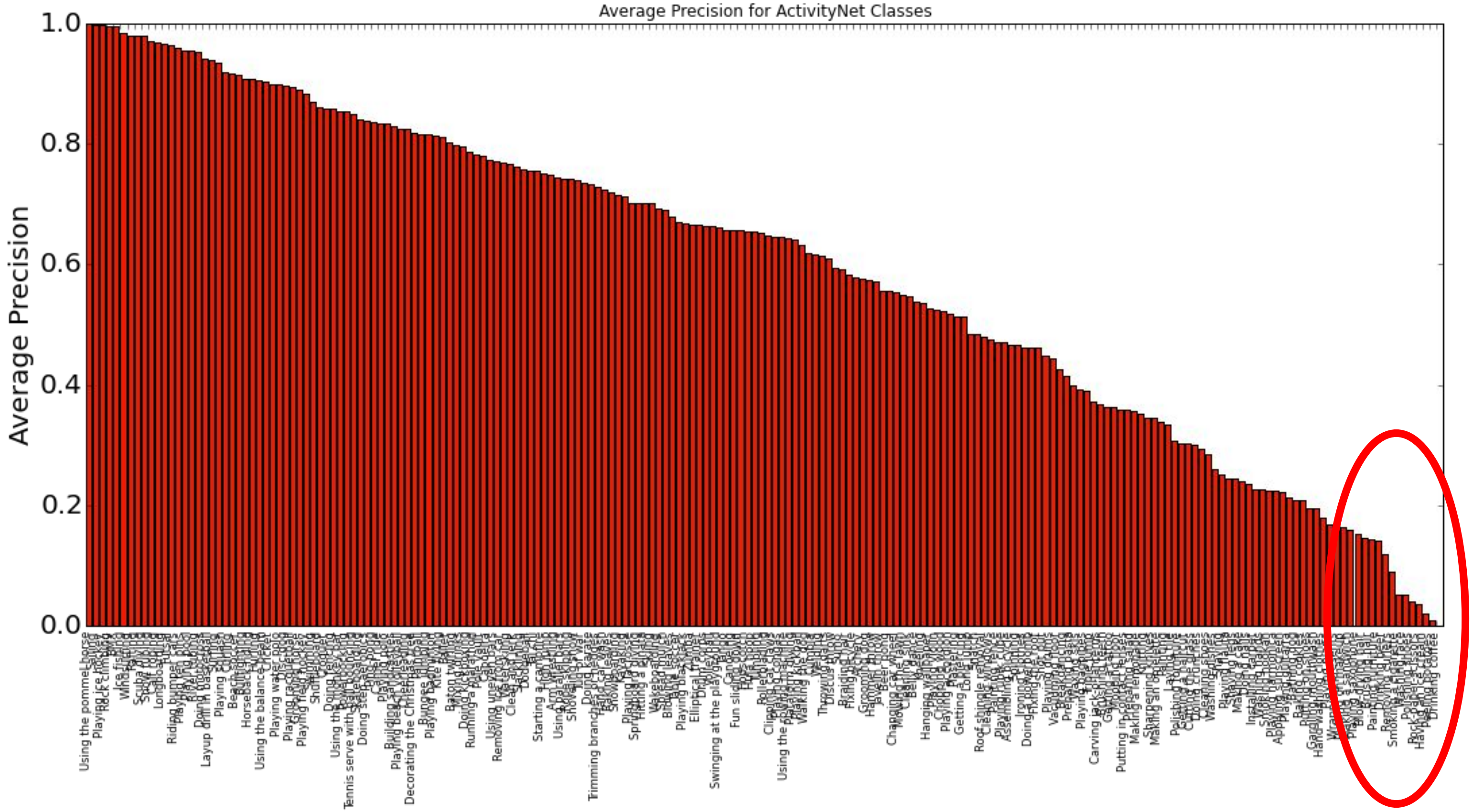
Temporal Activity Detection



Temporal Activity Detection



Temporal Activity Detection



Temporal Activity Detection

Ground Truth:
Playing water polo

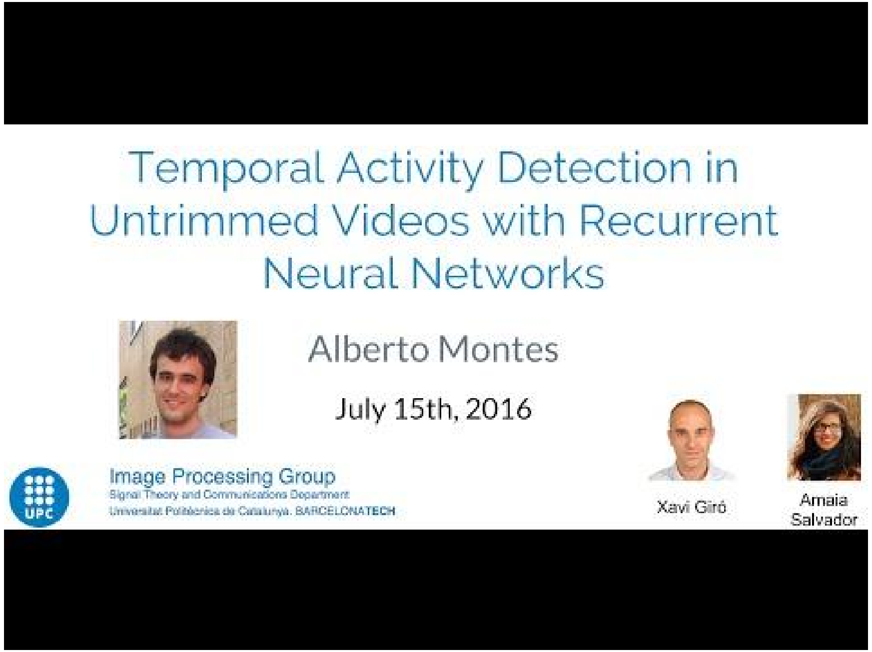
Prediction:
0.765 Playing water polo
0.202 Swimming
0.007 Springboard diving

Temporal Activity Detection

Ground Truth:
Hopscotch

Prediction:
0.848 Running a marathon
0.023 Triple jump
0.022 Javelin throw

Temporal Activity Detection



A. Montes, Salvador, A., Pascual-deLaPuente, S., and Giró-i-Nieto, X., "**Temporal Activity Detection in Untrimmed Videos with Recurrent Neural Networks**", in *1st NIPS Workshop on Large Scale Computer Vision Systems 2016* **(best poster award)**

Temporal Activity Detection

NIPS 2016
BARCELONA · SPAIN · DECEMBER 5-10 | http://nips.cc

A. Montes, Salvador, A., Pascual-deLaPuente, S., and Giró-i-Nieto, X., "**Temporal Activity Detection in Untrimmed Videos with Recurrent Neural Networks**", in *1st NIPS Workshop on Large Scale Computer Vision Systems 2016* **(best poster award)**

Outline

1. Recurrent Neural Networks
2. Activity Recognition
3. Object Tracking
4. Speech and Video
5. Learn more

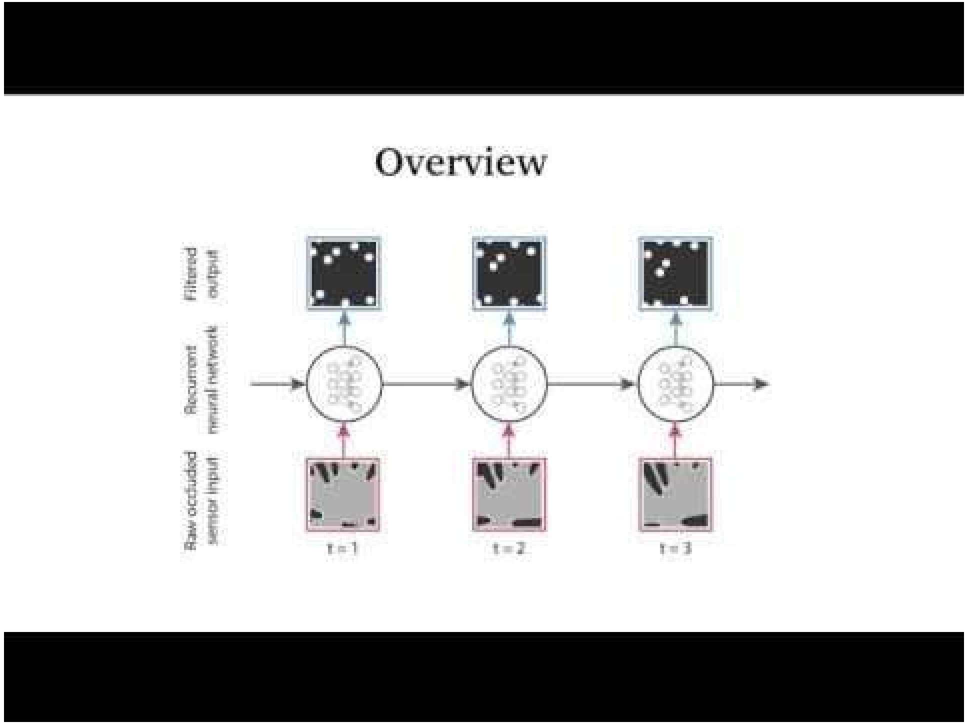
Object tracking: DeepTracking

Deep Tracking: Seeing Beyond Seeing Using Recurrent Neural Networks

Peter Ondrúška and **Ingmar Posner**
Mobile Robotics Group, University of Oxford, United Kingdom
{ondruska, ingmar}@robots.ox.ac.uk

P. Ondruska and I. Posner, [“Deep Tracking: Seeing Beyond Seeing Using Recurrent Neural Networks,”]() in *The Thirtieth AAAI Conference on Artificial Intelligence (AAAI)*, Phoenix, Arizona USA, 2016. [[code]]()

Object tracking: DeepTracking



P. Ondruska and I. Posner, [“Deep Tracking: Seeing Beyond Seeing Using Recurrent Neural Networks,”]() in *The Thirtieth AAAI Conference on Artificial Intelligence (AAAI)*, Phoenix, Arizona USA, 2016. [[code]]()

Object tracking: DeepTracking

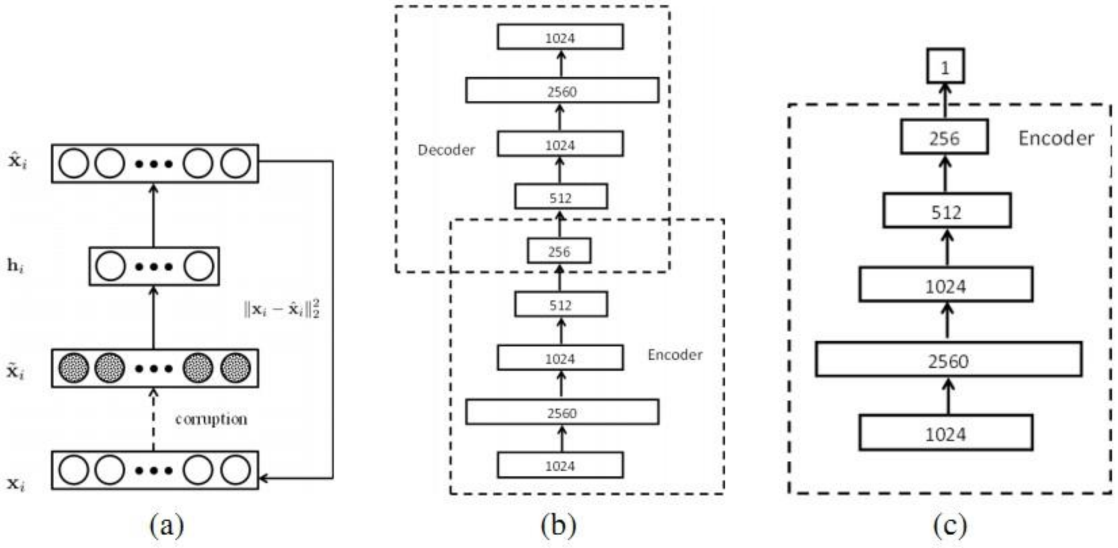
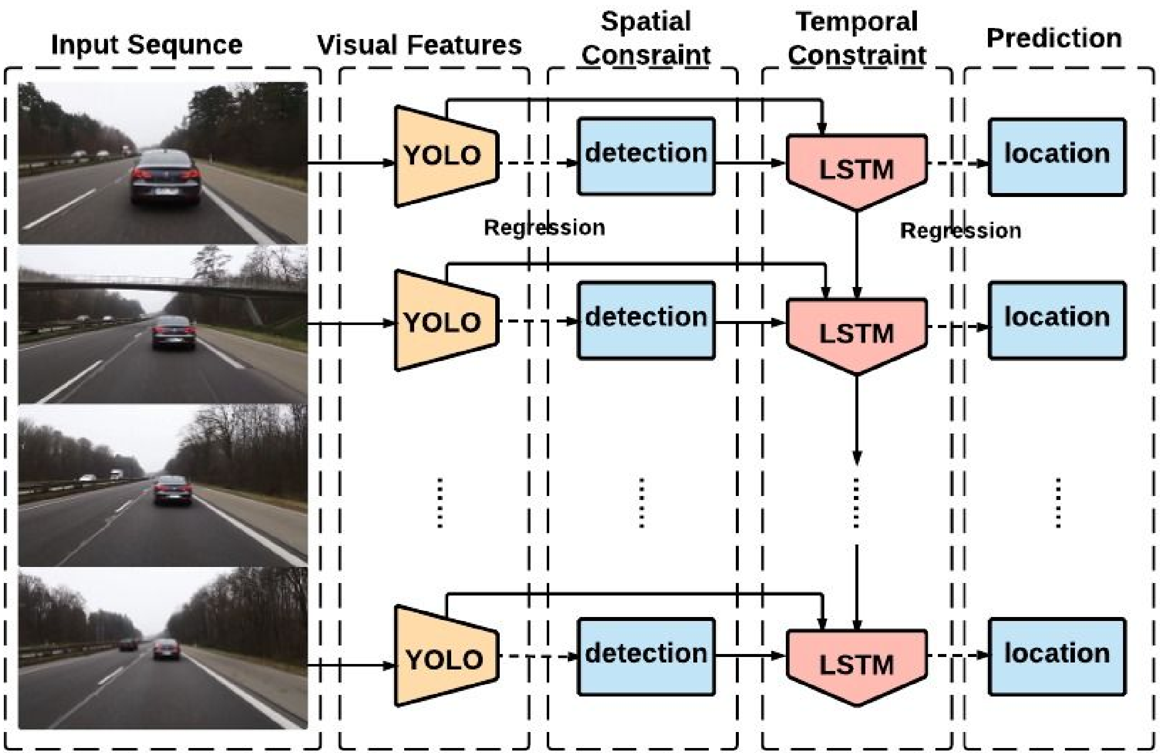


Figure 1: Some key components of the network architecture: (a) denoising autoencoder; (b) stacked denoising autoencoder; (c) network for online tracking.

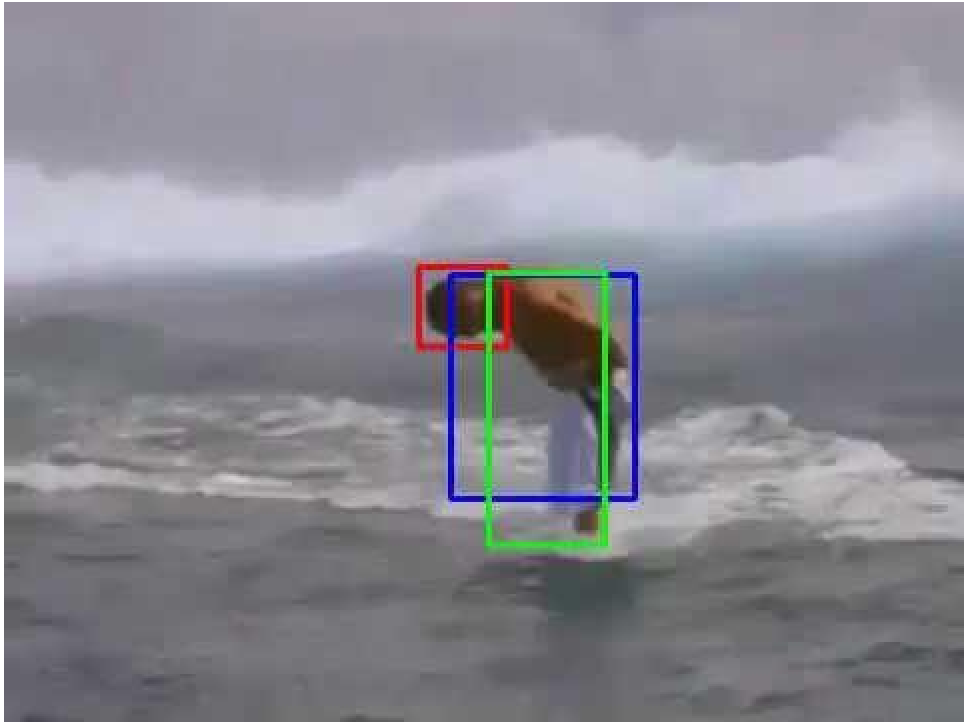
Wang, Naiyan, and Dit-Yan Yeung. "Learning a deep compact image representation for visual tracking." In *Advances in neural information processing systems*, pp. 809-817. 2013 [Project page with code]

Object tracking: ROLO



Ning, Guanghan, Zhi Zhang, Chen Huang, Zhihai He, Xiaobo Ren, and Haohong Wang. [“Spatially Supervised Recurrent Convolutional Neural Networks for Visual Object Tracking.”] arXiv preprint arXiv:1607.05781 (2016)

Object tracking: ROLO

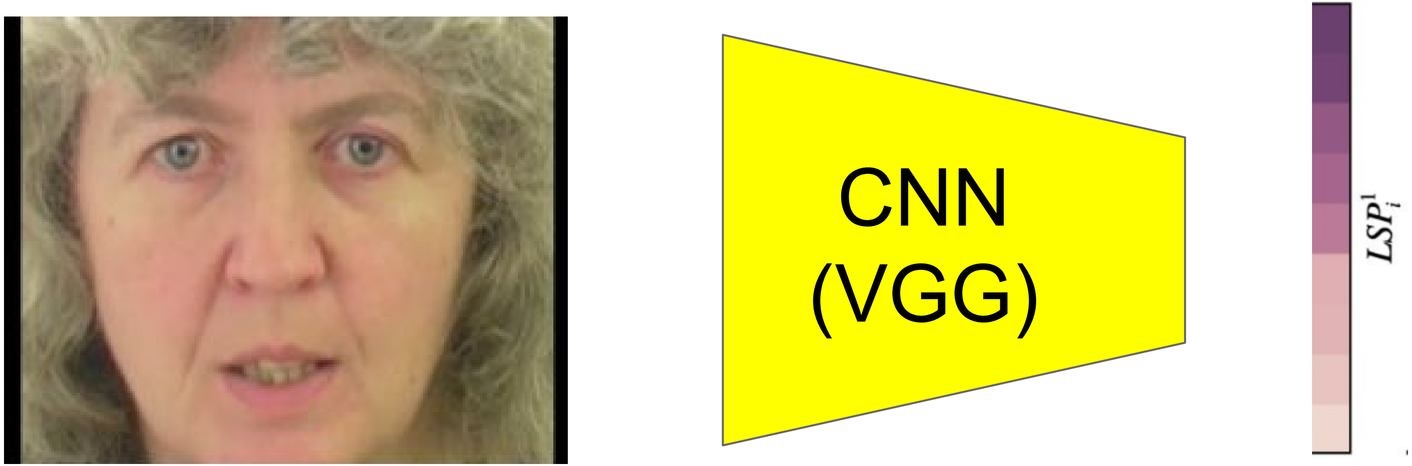


Ning, Guanghan, Zhi Zhang, Chen Huang, Zhihai He, Xiaobo Ren, and Haohong Wang. "Spatially Supervised Recurrent Convolutional Neural Networks for Visual Object Tracking." arXiv preprint arXiv:1607.05781 (2016)

Outline

1. Recurrent Neural Networks
2. Activity Recognition
3. Object Tracking
4. Speech and Video
5. Learn more

Speech and Video: Vid2Speech

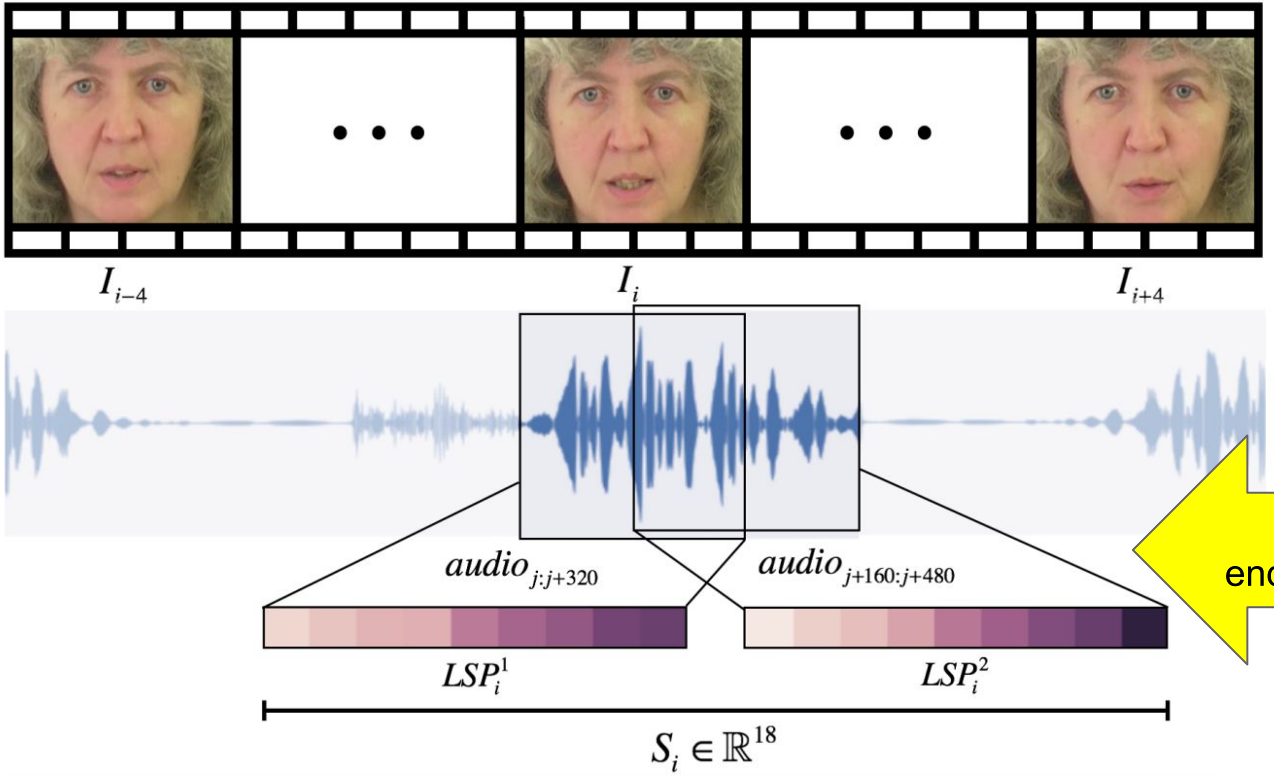


Frame from a
slient video

Audio feature

Ephrat, Ariel, and Shmuel Peleg. ["Vid2speech: Speech Reconstruction from Silent Video."]() *ICASSP 2017*

Speech and Video: Vid2Speech



No
end-to-end

Ephrat, Ariel, and Shmuel Peleg. ["Vid2speech: Speech Reconstruction from Silent Video."]() *ICASSP 2017*

Speech and Video: Vid2Speech

Ephrat, Ariel, and Shmuel Peleg. "Vid2speech: Speech Reconstruction from Silent Video." *ICASSP 2017*

Speech and Video: LipNet

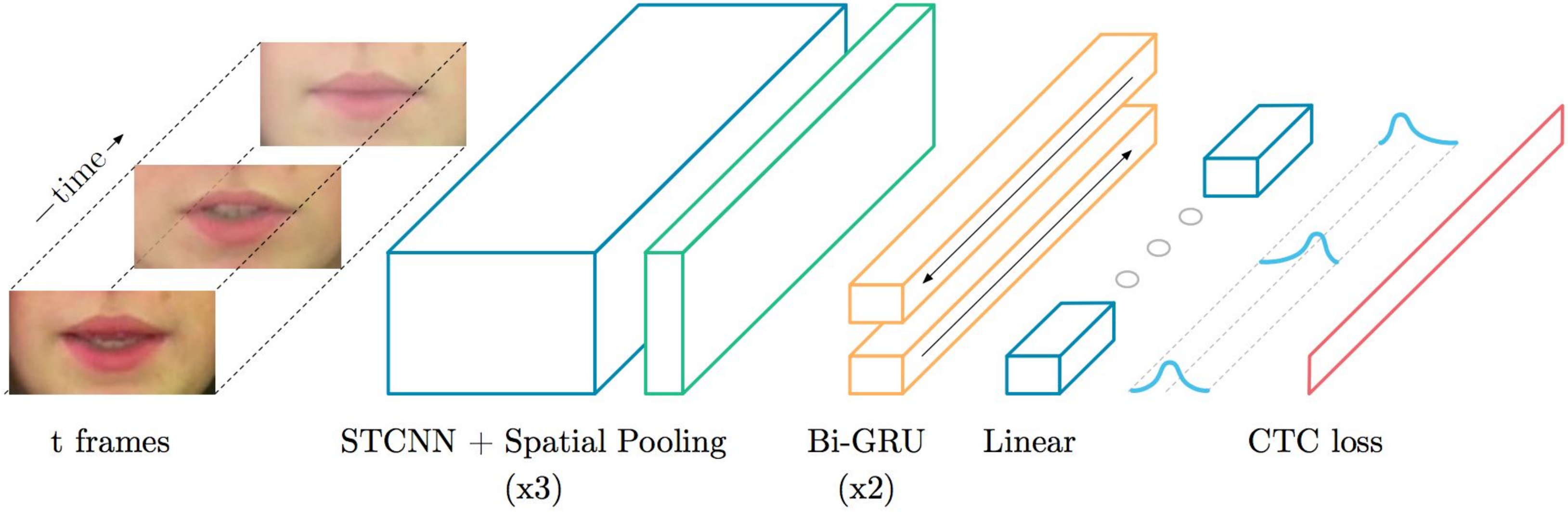


Figure 1: LipNet architecture. A sequence of T frames is used as input, and is processed by 3 layers of STCNN, each followed by a spatial max-pooling layer. The features extracted are processed by 2 Bi-GRUs; each time-step of the GRU output is processed by a linear layer and a softmax. This end-to-end model is trained with CTC.

Assael, Yannis M., Brendan Shillingford, Shimon Whiteson, and Nando de Freitas. ["LipNet: Sentence-level Lipreading."]() *arXiv preprint arXiv:1611.01599* (2016).

Speech and Video: LipNet

Assael, Yannis M., Brendan Shillingford, Shimon Whiteson, and Nando de Freitas. ["LipNet: Sentence-level Lipreading."]() *arXiv preprint arXiv:1611.01599* (2016).

Speech and Video: Watch, Listen, Attend & Spell

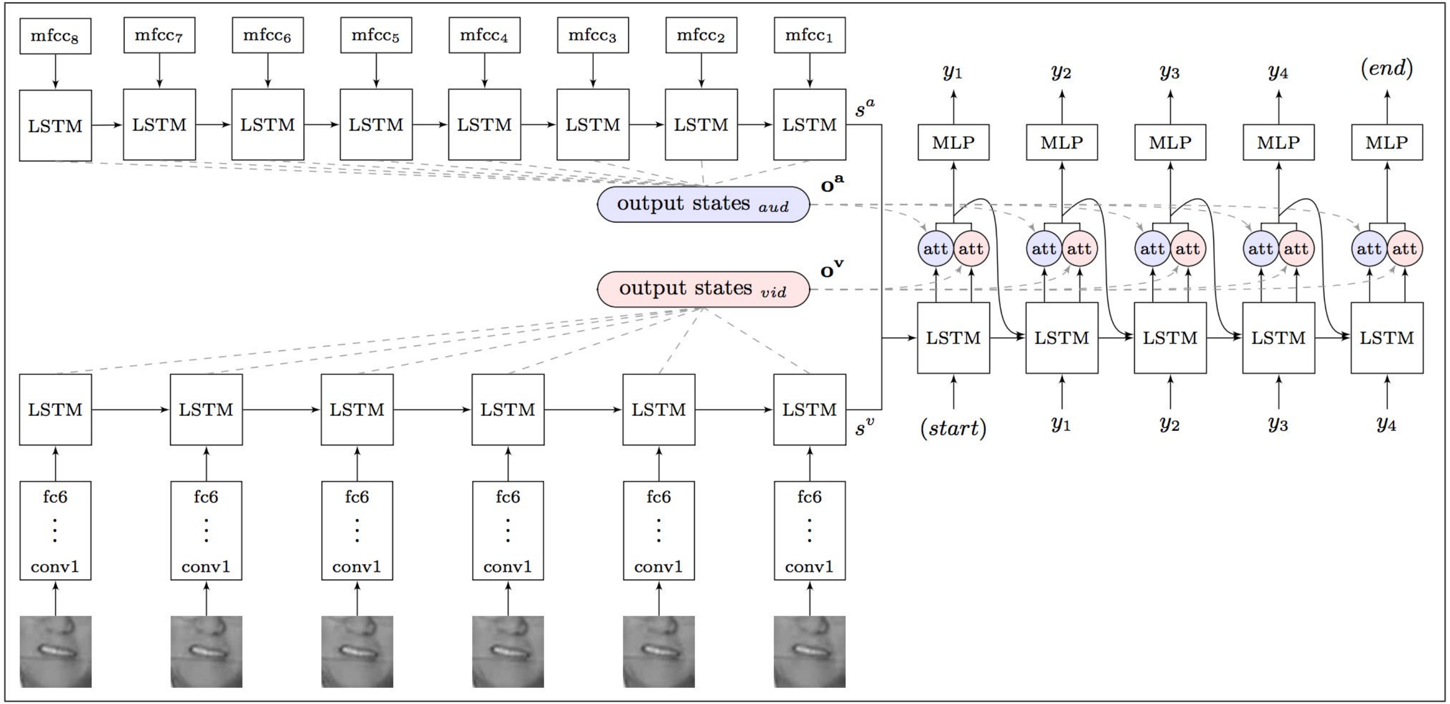


Figure 1. *Watch, Listen, Attend and Spell* architecture. At each time step, the decoder outputs a character y_i, as well as two attention vectors. The attention vectors are used to select the appropriate period of the input visual and audio sequences.

Chung, Joon Son, Andrew Senior, Oriol Vinyals, and Andrew Zisserman. ["Lip reading sentences in the wild."](#) *arXiv preprint arXiv:1611.05358* (2016).

Speech and Video: Watch, Listen, Attend & Spell

Chung, Joon Son, Andrew Senior, Oriol Vinyals, and Andrew Zisserman. ["Lip reading sentences in the wild."]() *arXiv preprint arXiv:1611.05358* (2016).

Outline

1. Recurrent Neural Networks
2. Activity Recognition
3. Object Tracking
4. Speech and Video
5. Learn more

Deep Learning: Software

Keras http://keras.io/

Tensor Flow https://www.tensorflow.org/

Caffe http://caffe.berkeleyvision.org/

Torch (Overfeat) http://torch.ch/

Theano http://deeplearning.net/software/theano/

MatconvNet (VLFeat) http://www.vlfeat.org/matconvnet/

CNTK (Mcrosoft) http://www.cntk.ai/

MxNet: https://github.com/dmlc/mxnet

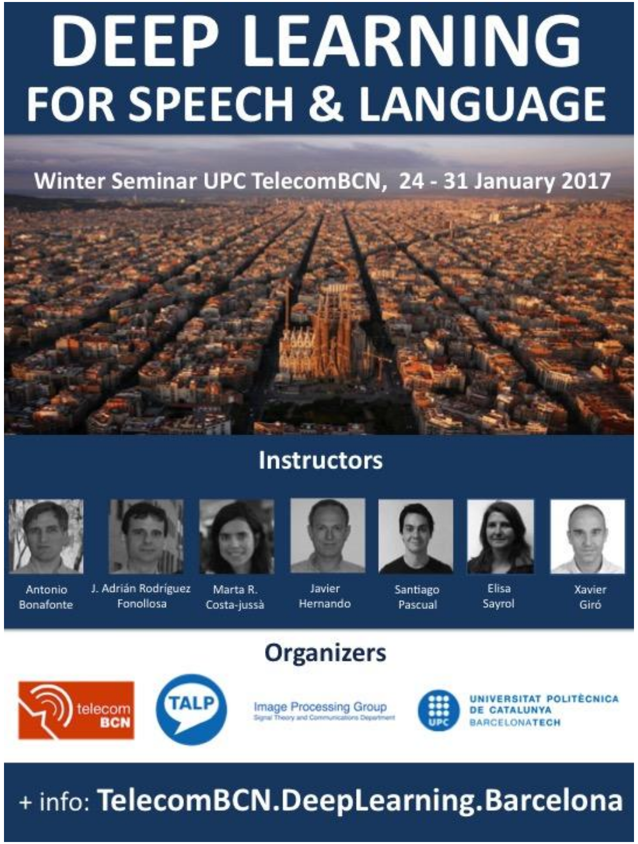
Learn more

Jordi Pont-Tuset (ETH Zurich), “One-shot Video Segmentation” on Monday 13 February at 12pm at IRI UPC

ETH zürich

Learn more

[Summer 2016] [Summer 2017]

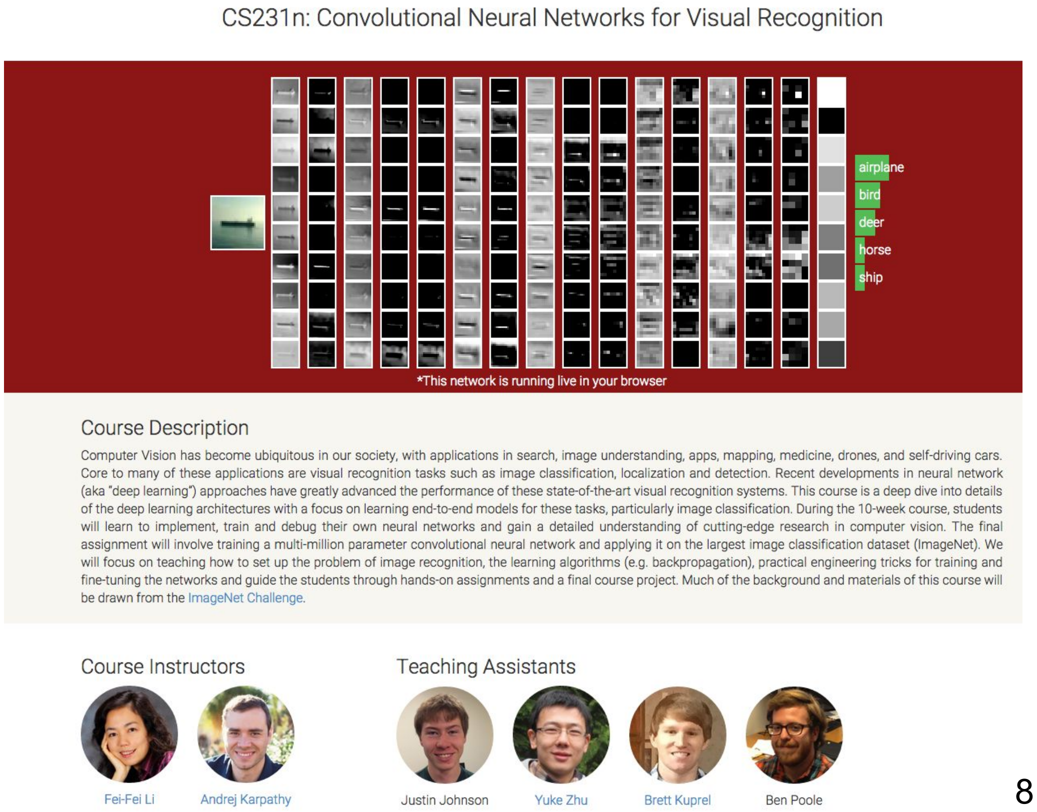


[course site]

Learn more

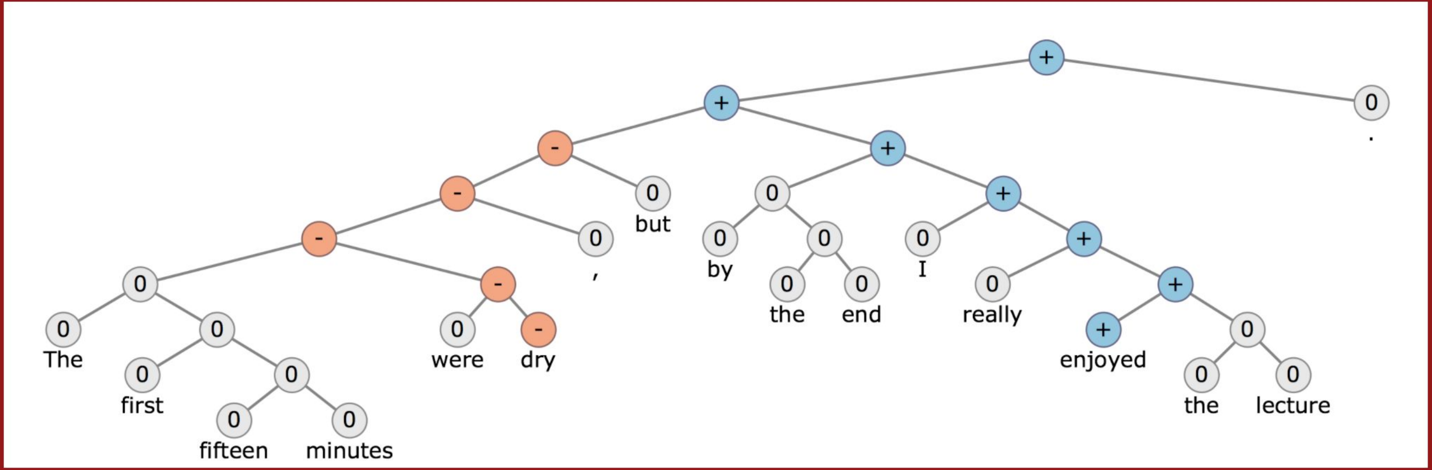
Stanford course:

[CS231n: Convolutional Neural Networks for Visual Recognition]



Learn more

CS224n: Natural Language Processing with Deep Learning



Chris Manning

Richard Socher

http://cs224n.stanford.edu/

Learn more

Online course:

[Deep Learning]

[Taking machine learning to the next level]

ConvNets: Learn more

[ReadCV seminar]

Friendly reviews of SoA papers

Spring 2016:

Tuesdays at 11am

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Department of Signal Theory
and Communications
Image Processing Group

UNIVERSITAT DE BARCELONA

Deep Learning: Learn more

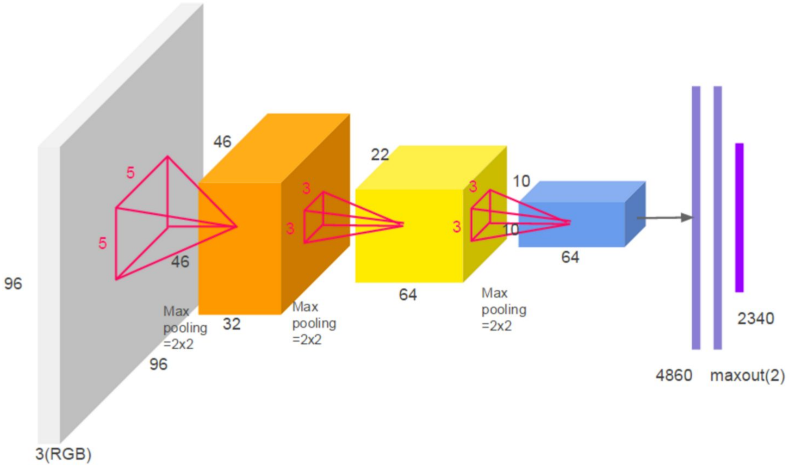
Summer course

[Deep Learning for Computer Vision]

June 21-27, 3-7pm

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Department of Signal Theory
and Communications
Image Processing Group

Dublin City University
Ollscoil Chathair Bhaile Átha Cliath



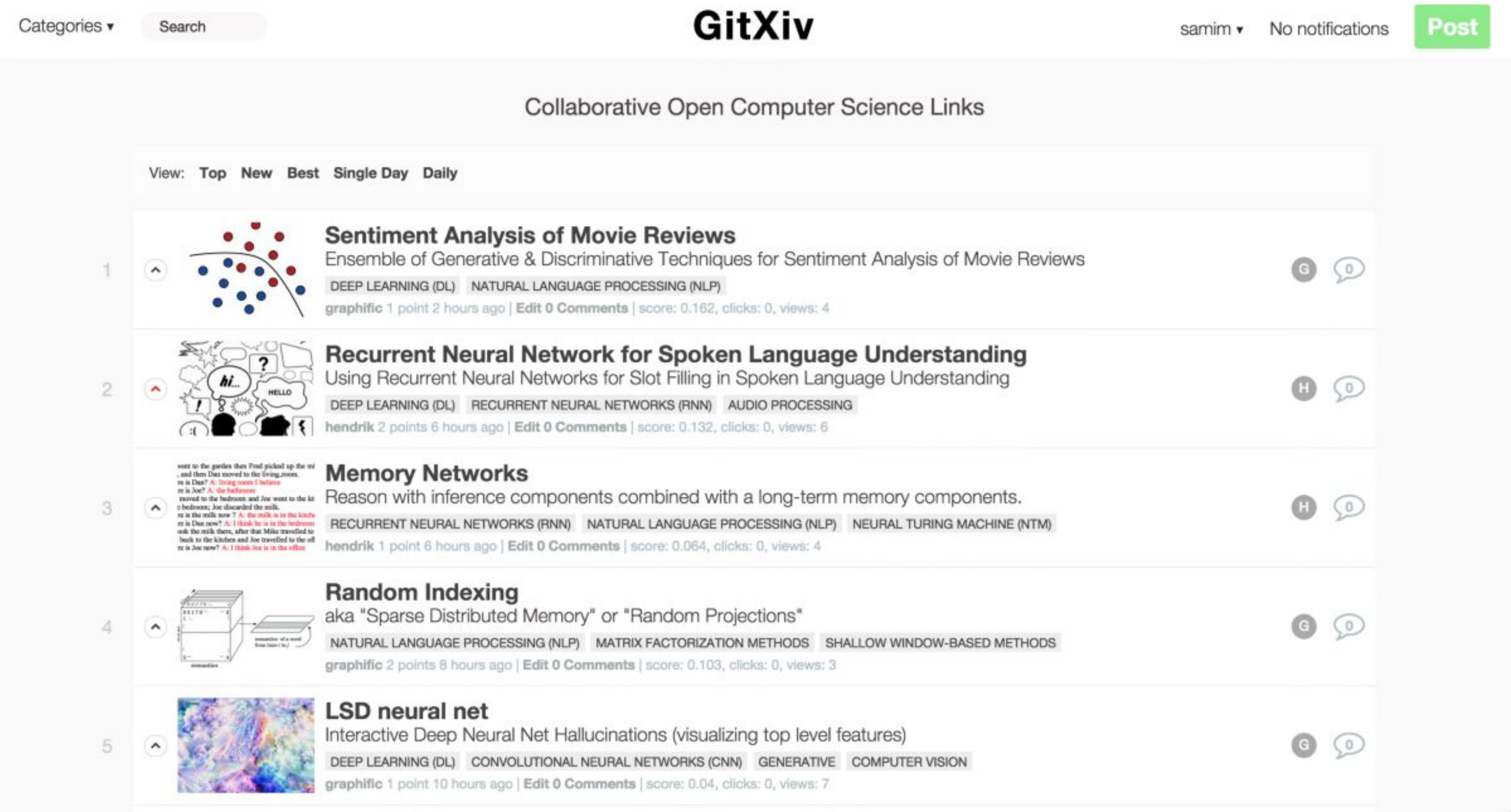
ConvNets: Learn more

- Deep learning methos for vision (CVPR 2012)
- Tutorial on deep learning for vision (CVPR 2014)
- Kyunghyun Cho, “Deep Learning: Past, Present & Future”

ConvNets: Learn more

“Machine learning” sub-Reddit.

ConvNets: Learn more



ConvNets: Learn more

Check profile requirements for Summer internship (disclaimer: offered to Phd students by default)

Company	Avg Salary / hour	Avg Salary / month
Yahoo	$43	($43x160=$6,880)
Apple	$37	($37x160=$5,920)
Google	$29.54-$31.32	$7,151
Facebook	$22.92	$6,150-$7,378
Microsoft	$22.63	$6,506-$7,171

glassdoor.com

Source: Glassdoor.com (internships in California. No stipends included)

Learn more

Video: Cristian Canton’s talk “From Catalonia to America: notes on how to achieve a successful post-Phd career ”@ ACMCV 2015 & UPC

Learn more

Li Fei-Fei, [“How we’re teaching computers to understand pictures”]() TEDTalks 2014.

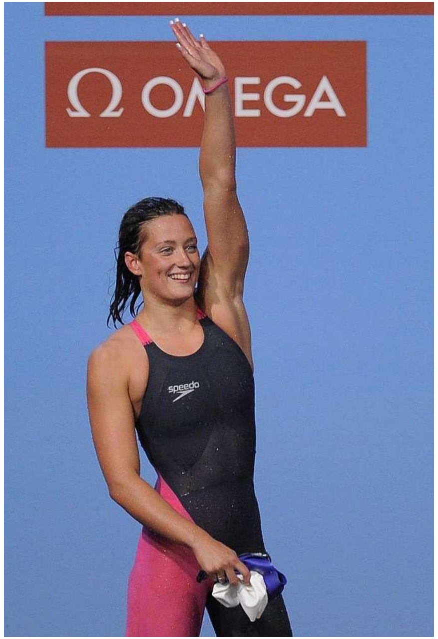
Learn more

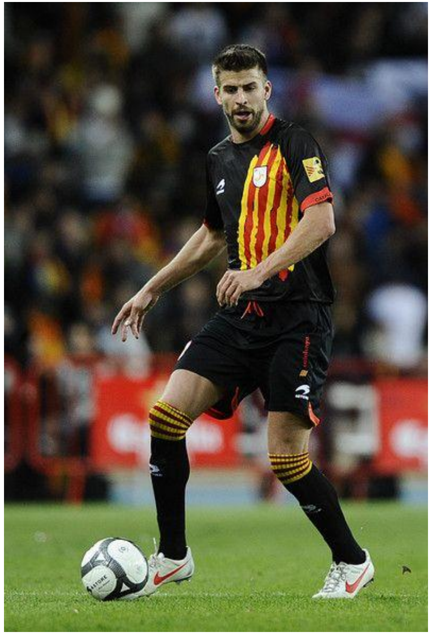
Jeremy Howard, “The wonderful and terrifying implications of computers that can learn”, TEDTalks 2014.

Learn more

- Neil Lawrence, OpenAI won't benefit humanity without open data sharing (The Guardian, 14/12/2015)

Sports: Do you know them ?





Deep Learning: Do you know them ?

Antonio Torralba, MIT
(former UPC)

Oriol Vinyals, Google
(former UPC)

Jose M Álvarez, NICTA
(former URL & UAB)

Joan Bruna, Berkeley
(former UPC)

...and MANY MORE I am missing in the page (apologies).

Where you are studying

VisioCat dinner
@ CVPR 2015

Where you are studying

VisioCat dinner
@ CVPR 2016

ConvNets: Discussion

Is Computer Vision solved ?

Thank you !

Slides available on and .

https://imatge.upc.edu/web/people/xavier-giro

http://bitsearch.blogspot.com

https://twitter.com/DocXavi

https://www.facebook.com/ProfessorXavi

xavier.giro@upc.edu