Day 6 Lecture 2

Recurrent Neural Networks

Xavier Giro-i-Nieto

@DocXavi
xavier.giro@upc.edu

Associate Professor
Universitat Politècnica de Catalunya

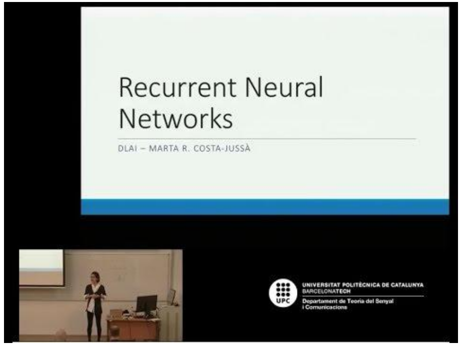
telecos BCN

UPC

Acknowledgments

Santiago Pascual

Marta R. Costa-jussà



UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Language and Speech Technologies

TALP

[Santiago Pascual
UPC TelecomBCN
DLSL 2017]

[Marta R. Costa-Jussà
UPC TelecomBCN
DLAI 2017]

Similar Lecture

[Deep Learning for Artificial Intelligence, UPC TelecomBCN 2020](#)

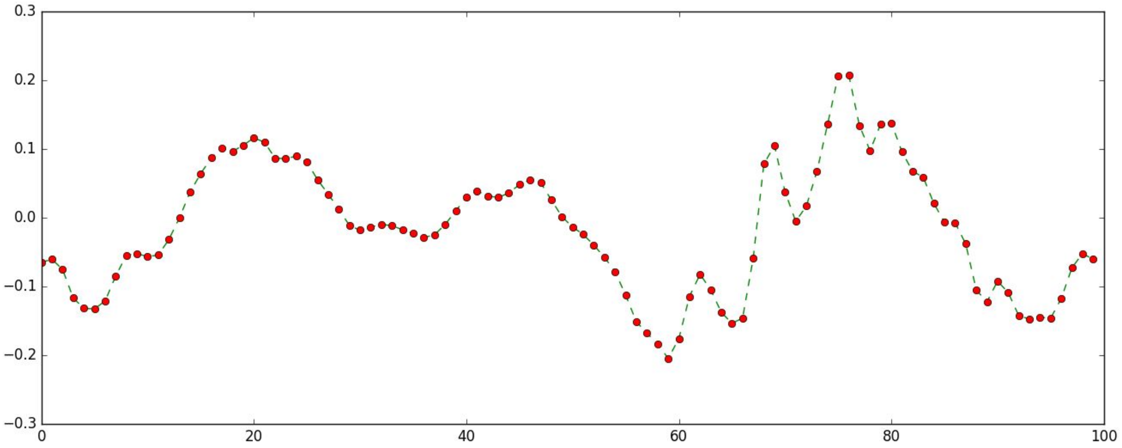
Source: (AI Memes) AI & Deep Learning Memes For Back-propagated Poets

Outline

1. Motivation

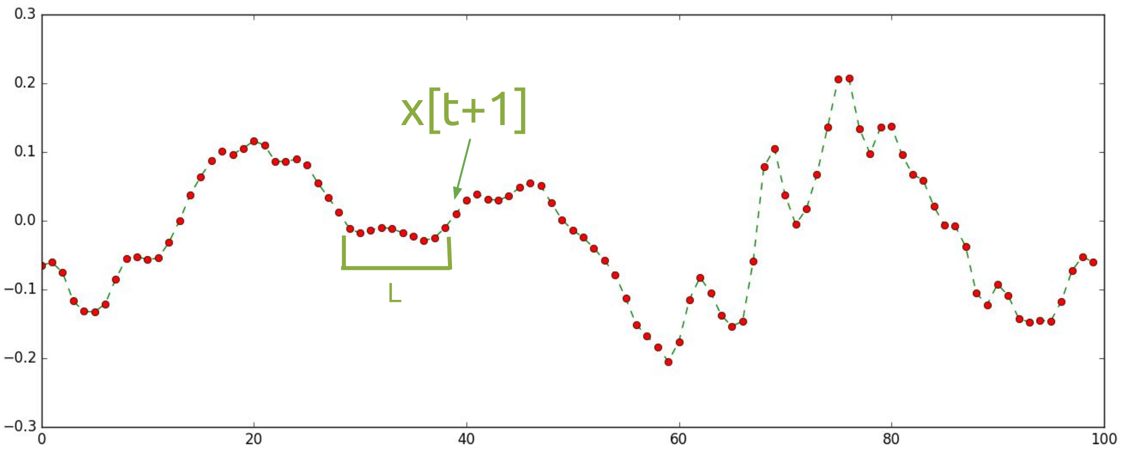
Sequences (a.k.a time series)

Consider a sequence of samples (eg. of temperature):



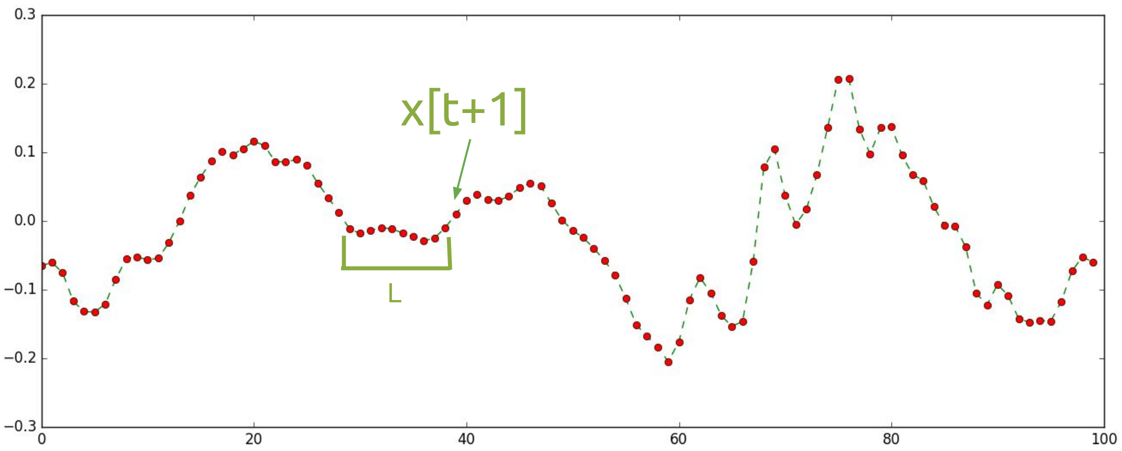
Sequences with Feed Forward MLP

Task:
Predict sample x[t+1] knowing previous values {x[t], x[t-1], x[t-2], ..., x[t-L]}



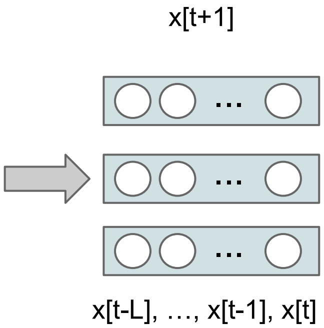
How could you solve this task with a naive feed forward MLP ?

Sequences with Feed Forward MLP

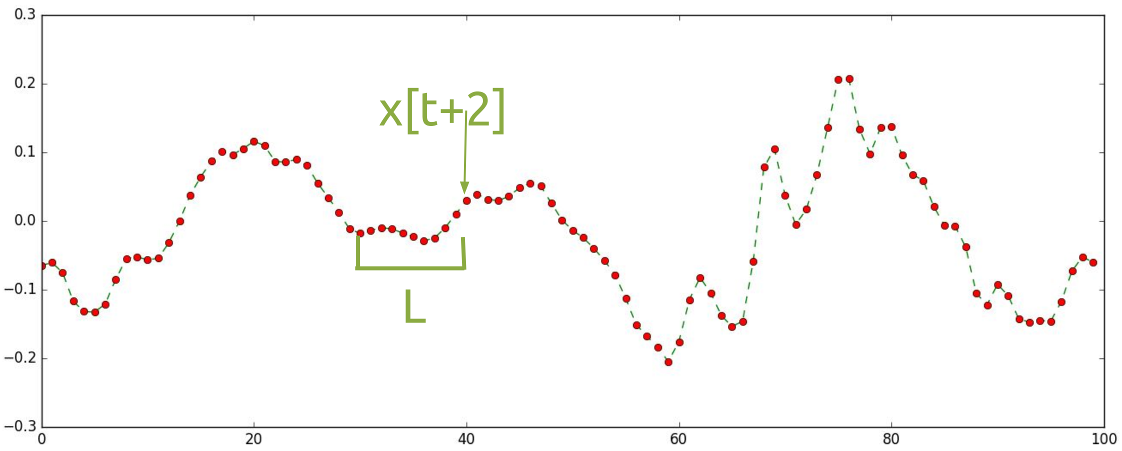


Feed Forward approach:

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

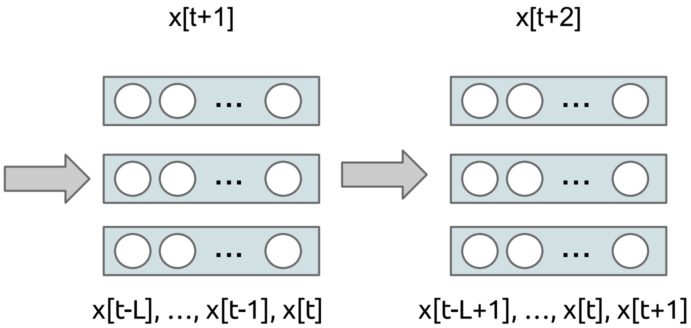


Sequences with Feed Forward MLP

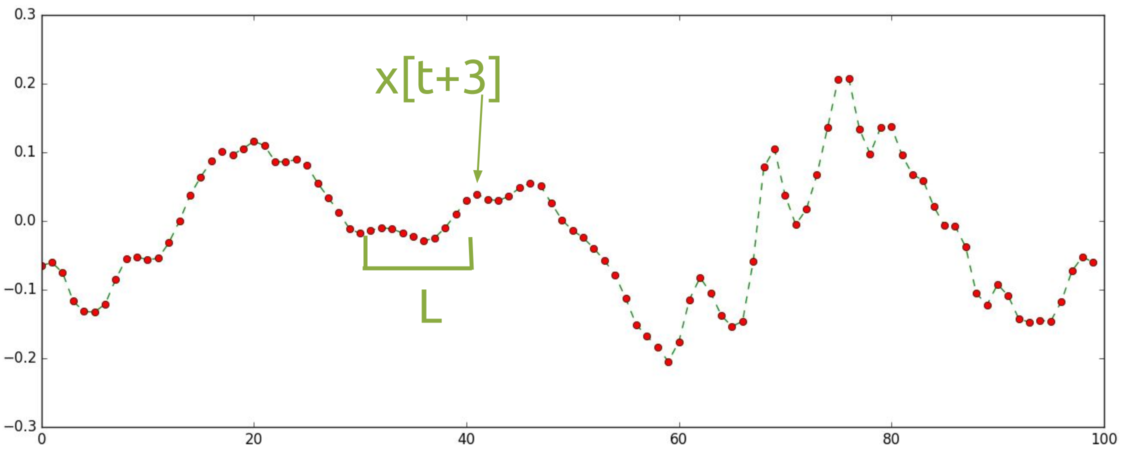


Feed Forward approach:

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

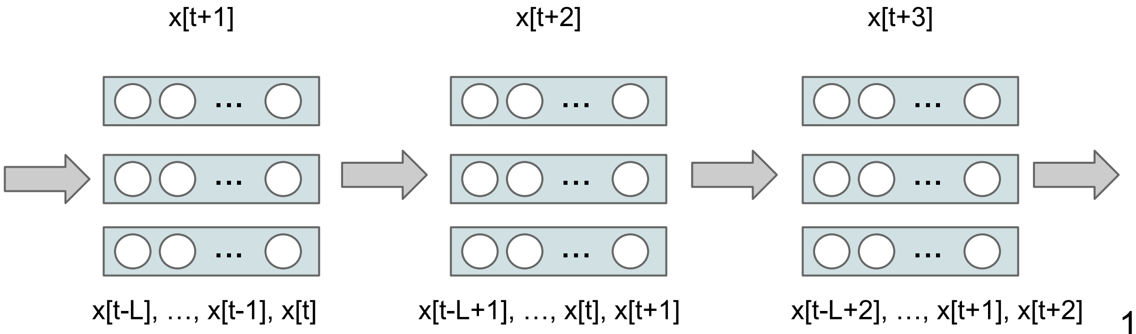


Sequences with Feed Forward MLP



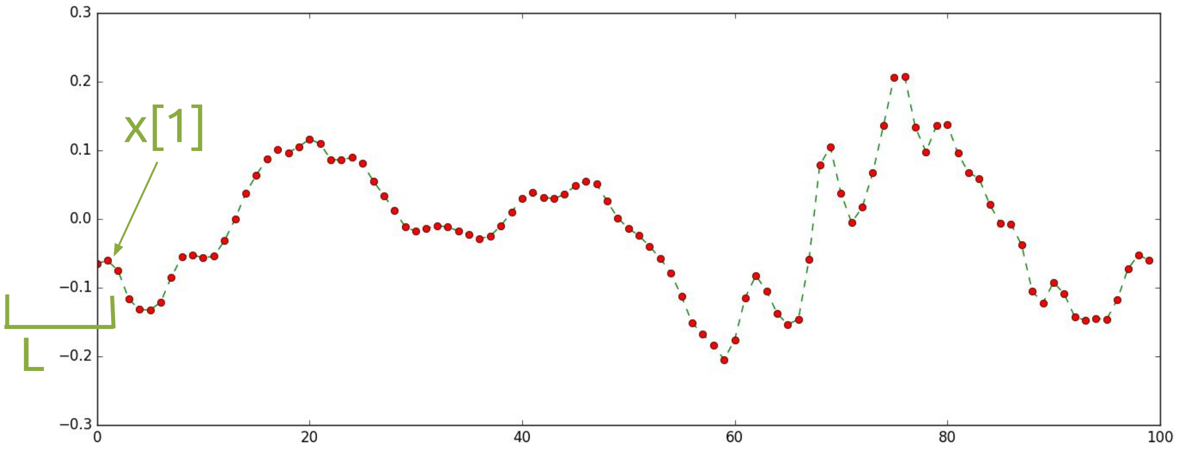
Feed Forward approach:

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



Sequences with Feed Forward MLP

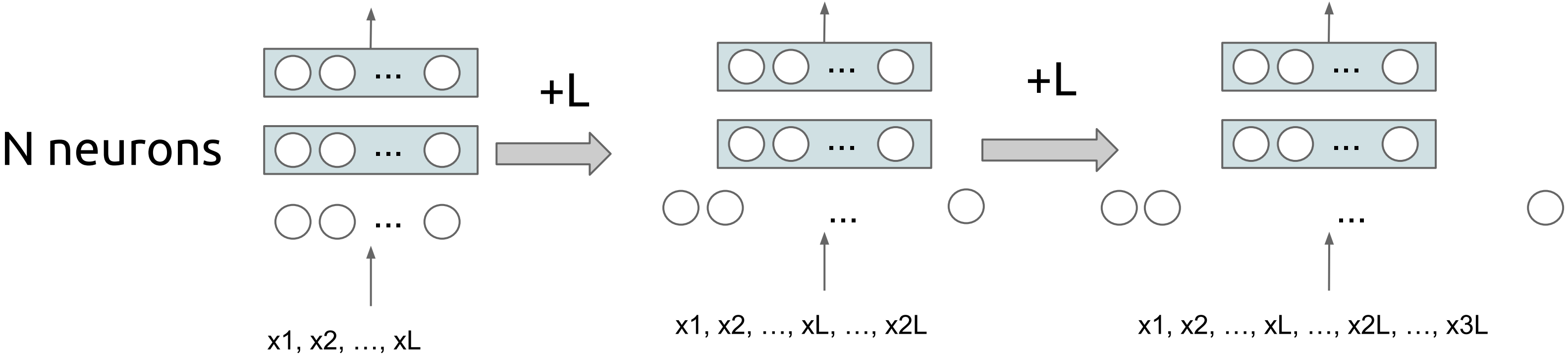
How would you deal with the prediction at the beginning of the sequence ?



By adding (zero) padding for the samples necessary to fill the input layer of the MLP.

Sequences with Feed Forward MLP

In the first layer, how does an increase of the receptive field (L) affect the amount of parameters to learn ?



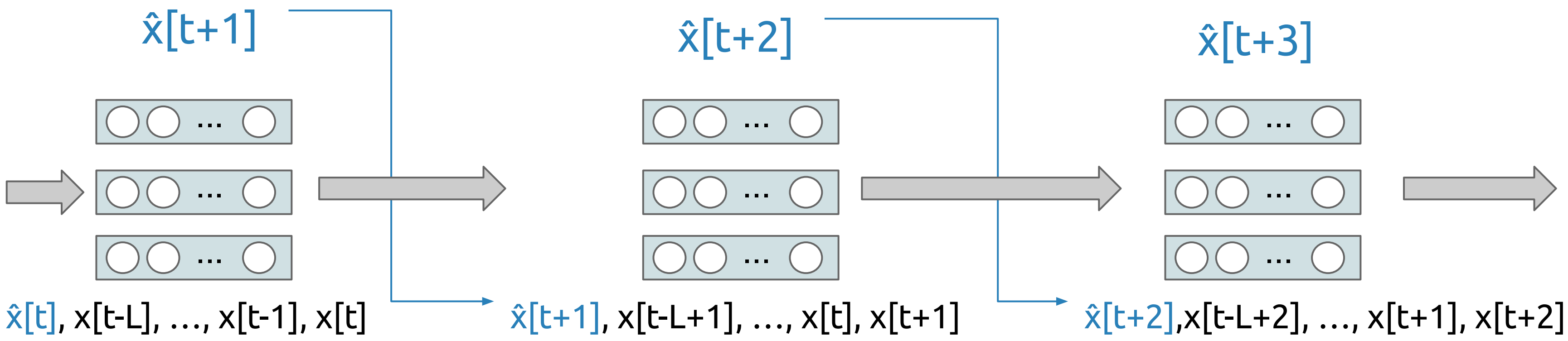
Receptive field	L	2L	3L
#params in 1st layer	N x L + N	N x 2L + N	N x 3L + N

A +1 increase of the receptive field increases the parameters with a factor N.

Sequences with Feed Forward MLP

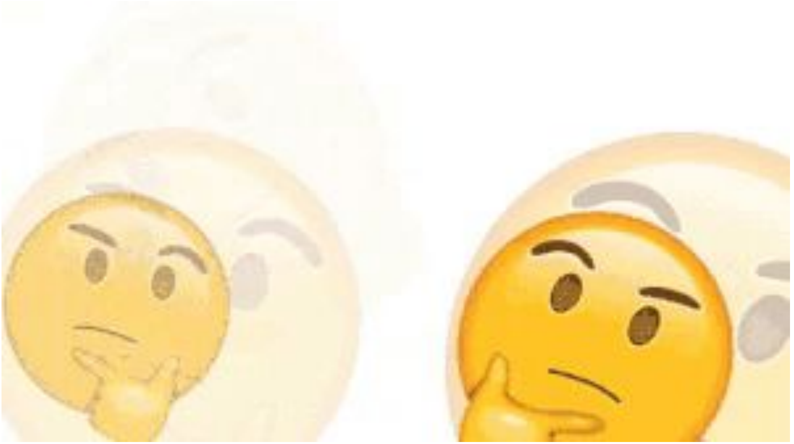
How could the previous decisions condition the future predictions ?

Adopting an **auto-regressive** strategy, by which the previous outputs become inputs in future steps.



Could we find an architecture that:

- adapts to any input length to avoid padding.
- does not increase the complexity with the receptive field.
- can naturally remember what it has seen in previous timesteps ?



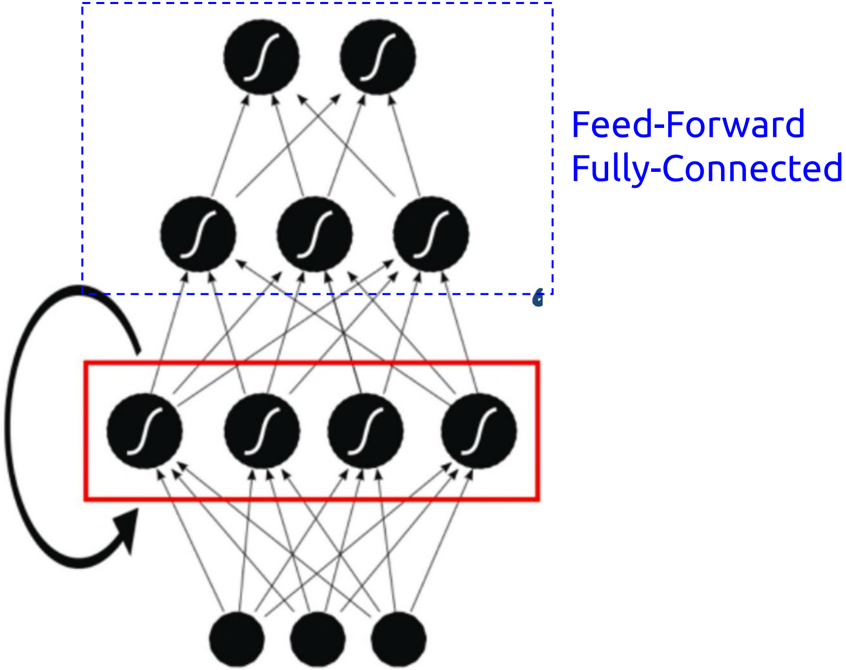
Outline

1. Motivation
2. Recurrent Neural Layer

Recurrent Neural Layer

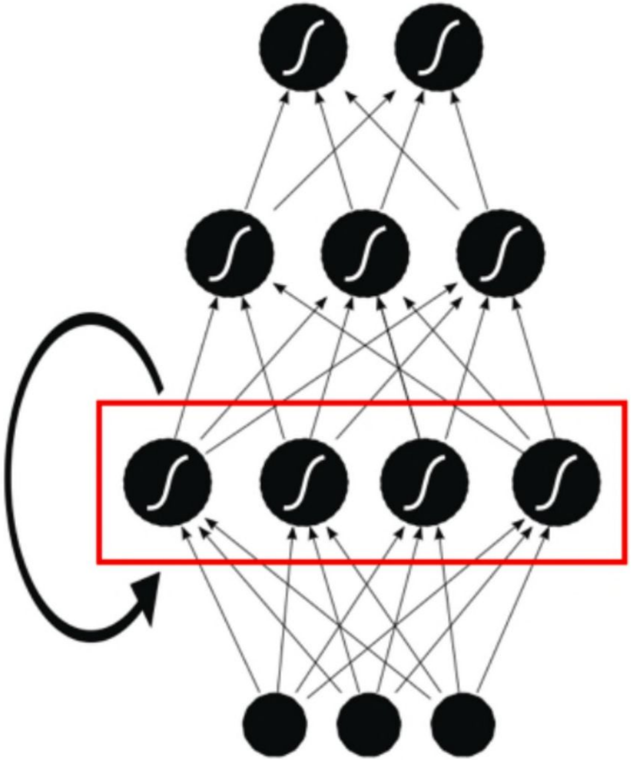
Recurrent layer (RNN)

The hidden state of a recurrent layer h_t also depends from its previous state h_{t-1}.



Elman, J. L. (1990). [Finding structure in time](). Cognitive science, 14(2), 179-211.

Recurrent Neural Layer



Feed-forward Weights (W)

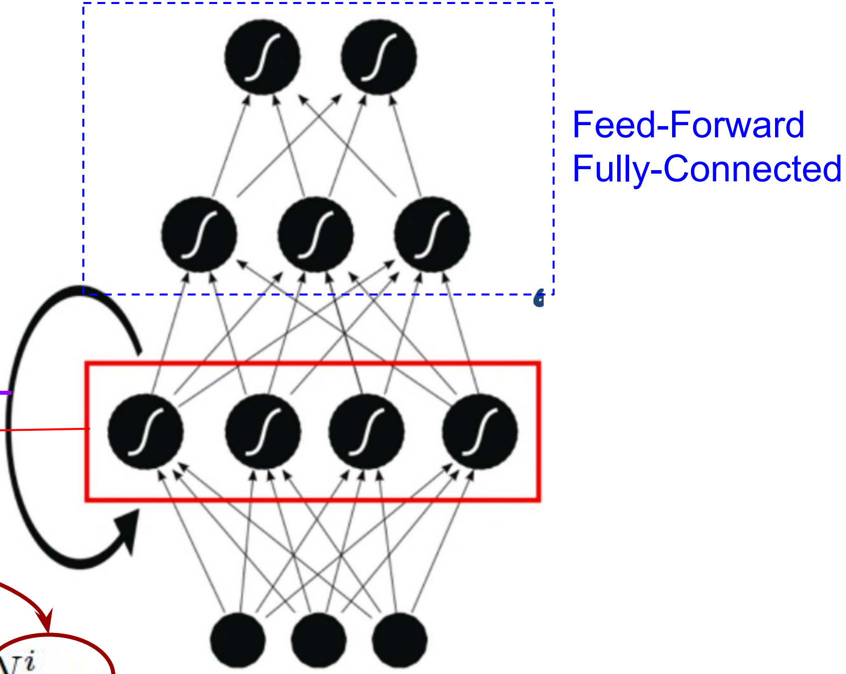
$$h_t = f(\boxed{W} \cdot x_t + \boxed{U} \cdot h_{t-1} + b)$$

Recurrent Weights (U)

Elman, J. L. [Finding structure in time](). Cognitive science (1990).

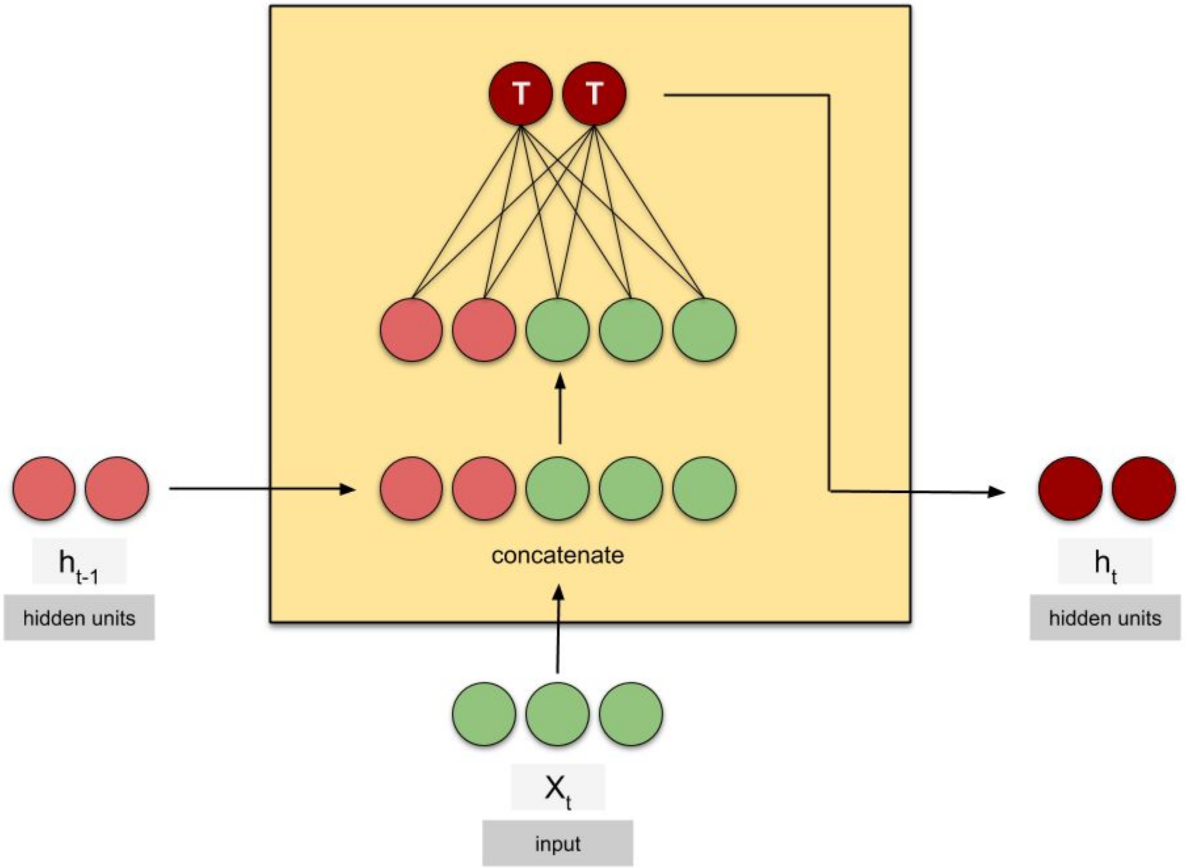
Recurrent Neural Layer

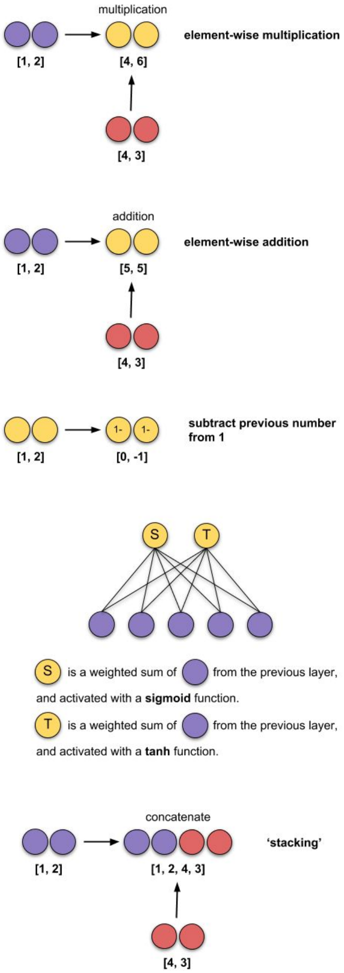
The amount of learnable parameters increases with the additional U matrix.



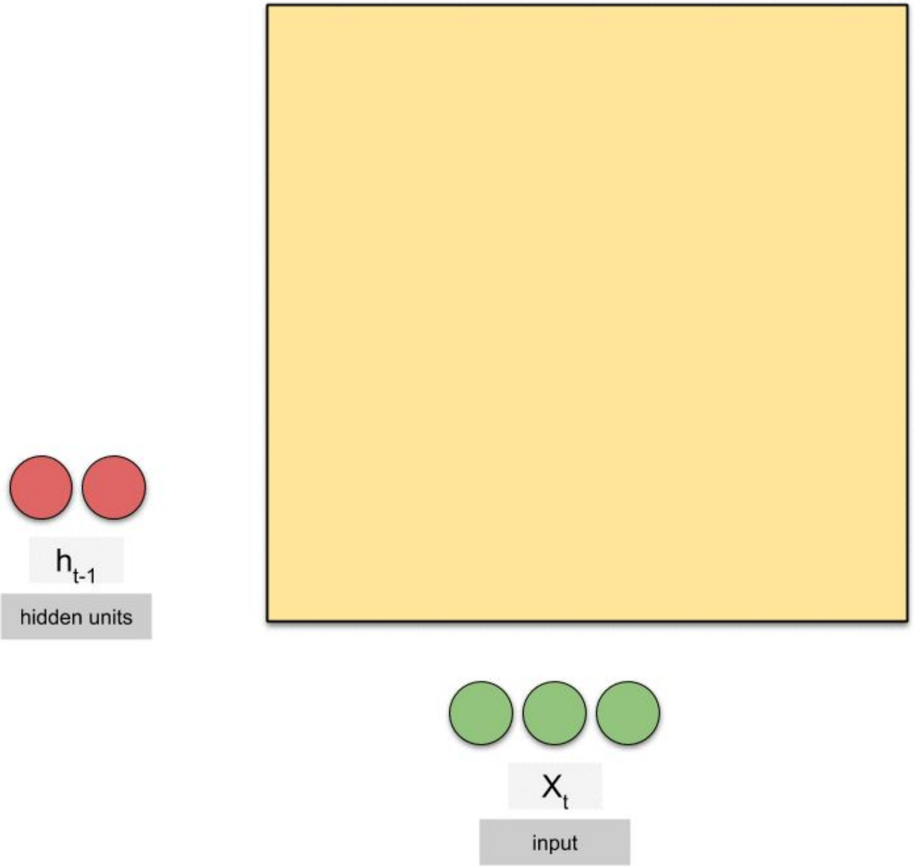
$$h_t = f(W \cdot x_t + U \cdot h_{t-1} + b)$$

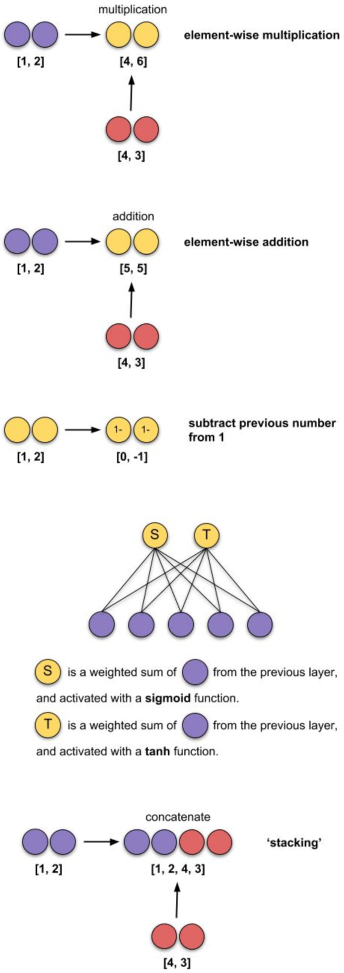
$$N^i_{params} = N^i_{inputs} \times N^i_{units} + N^i_{units} \times N^i_{units} + N^i_{units}$$





Raimi Karim, "Animated RNN, LSTM and GRU". Towards Data Science 2018.



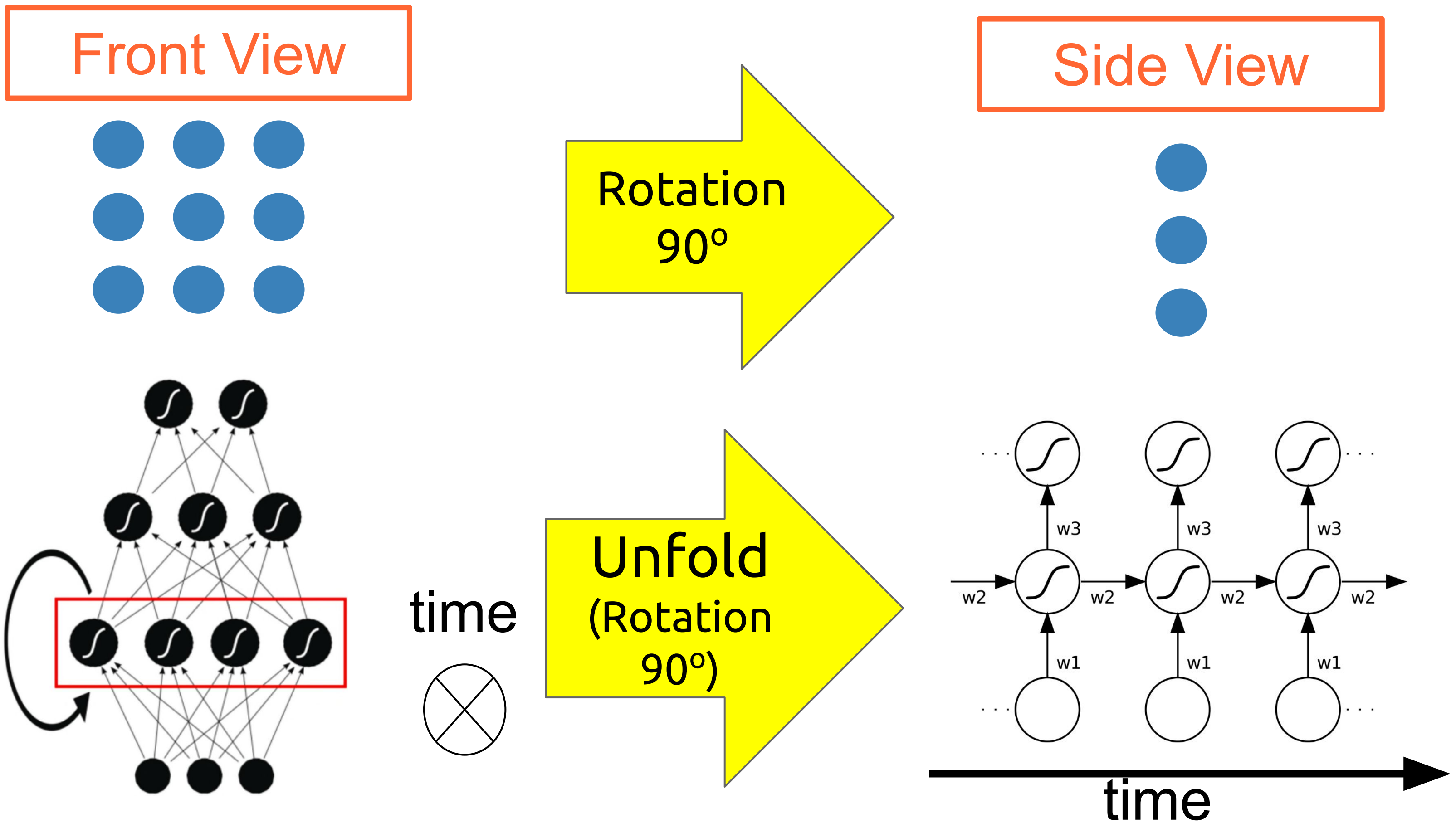


Raimi Karim, [“Animated RNN, LSTM and GRU”](). Towards Data Science 2018.

Outline

1. Motivation
2. Recurrent Neural Layer
3. Forward & Backward Passes Through Time

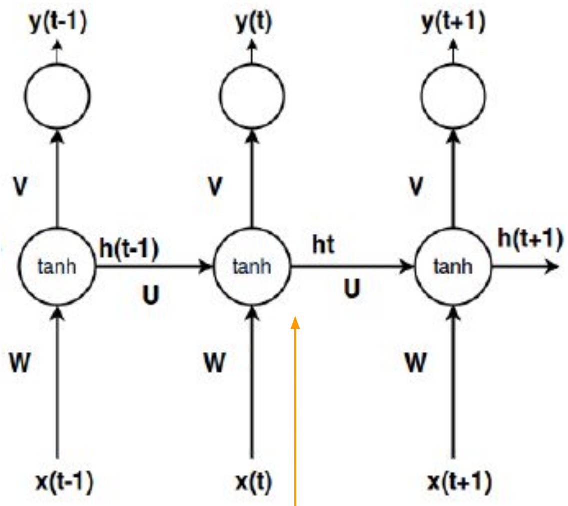
Forward Pass in RNN



Forward Pass in RNN

Hence we have two data flows: **Forward in neural layers + time** propagation

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



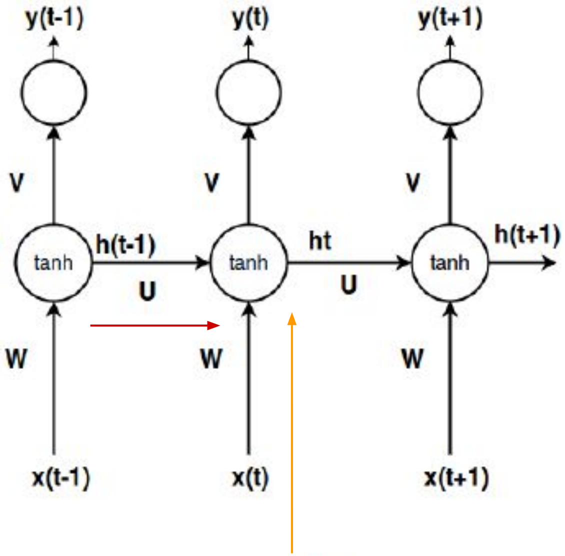
$\mathbf{x}_t \in \mathbb{R}^n$

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

Forward Pass in RNN

Hence we have two data flows: **Forward in layers + time** propagation

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



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

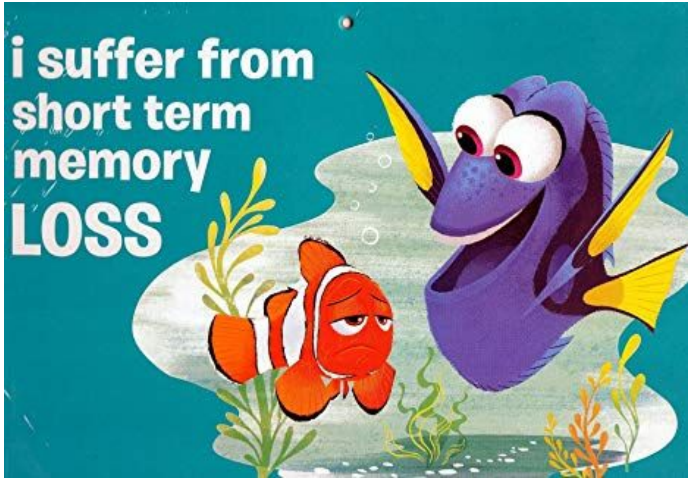
$$\mathbf{h}_t = g(\mathbf{W} \cdot \mathbf{x}_t + \mathbf{U} \cdot \mathbf{h}_{t-1} + \mathbf{b}_h)$$

Forward Pass in RNN

Main problem:

- **Long-term memory** (remembering quite far time-steps) **vanishes quickly** because of the recursive operation with non-linearities **g(·)** and **U**.

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



Backpropagation Through Time (BPTT)

Back Propagation Through Time (BPTT): The training method has to take into account the time operations:

Total error at the output is the sum of errors at each time-step t

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

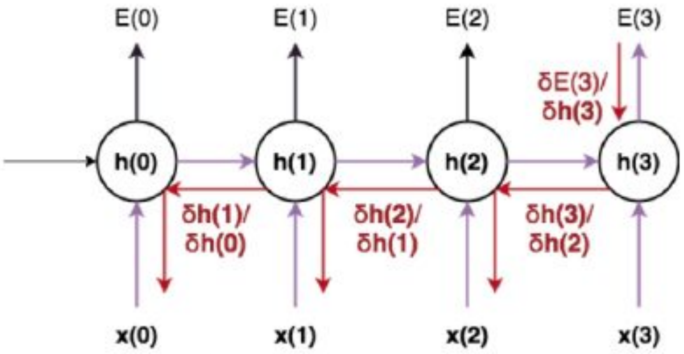
Total gradient is the sum of gradients at each time-step t

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

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

Backpropagation Through Time (BPTT)

Main problems:

- **Exploding / vanishing gradients:** During training gradients explode/vanish easily because of depth-in-time.

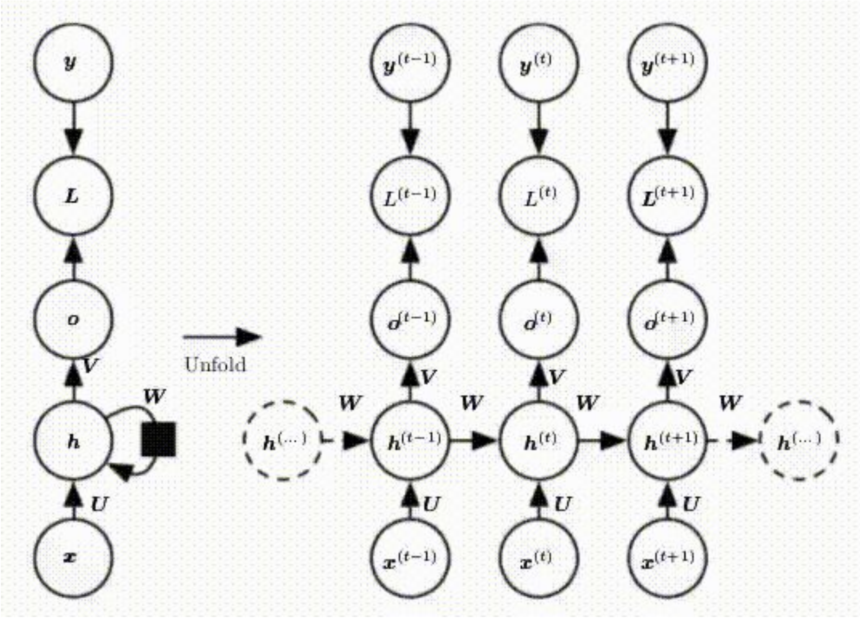


Figure: Jordi Pons

Outline

1. Motivation
2. Recurrent Neural Layer
3. Forward & Backward Passes Through Time
4. Gated Recurrent Neurons
    a. LSTM
    b. GRU

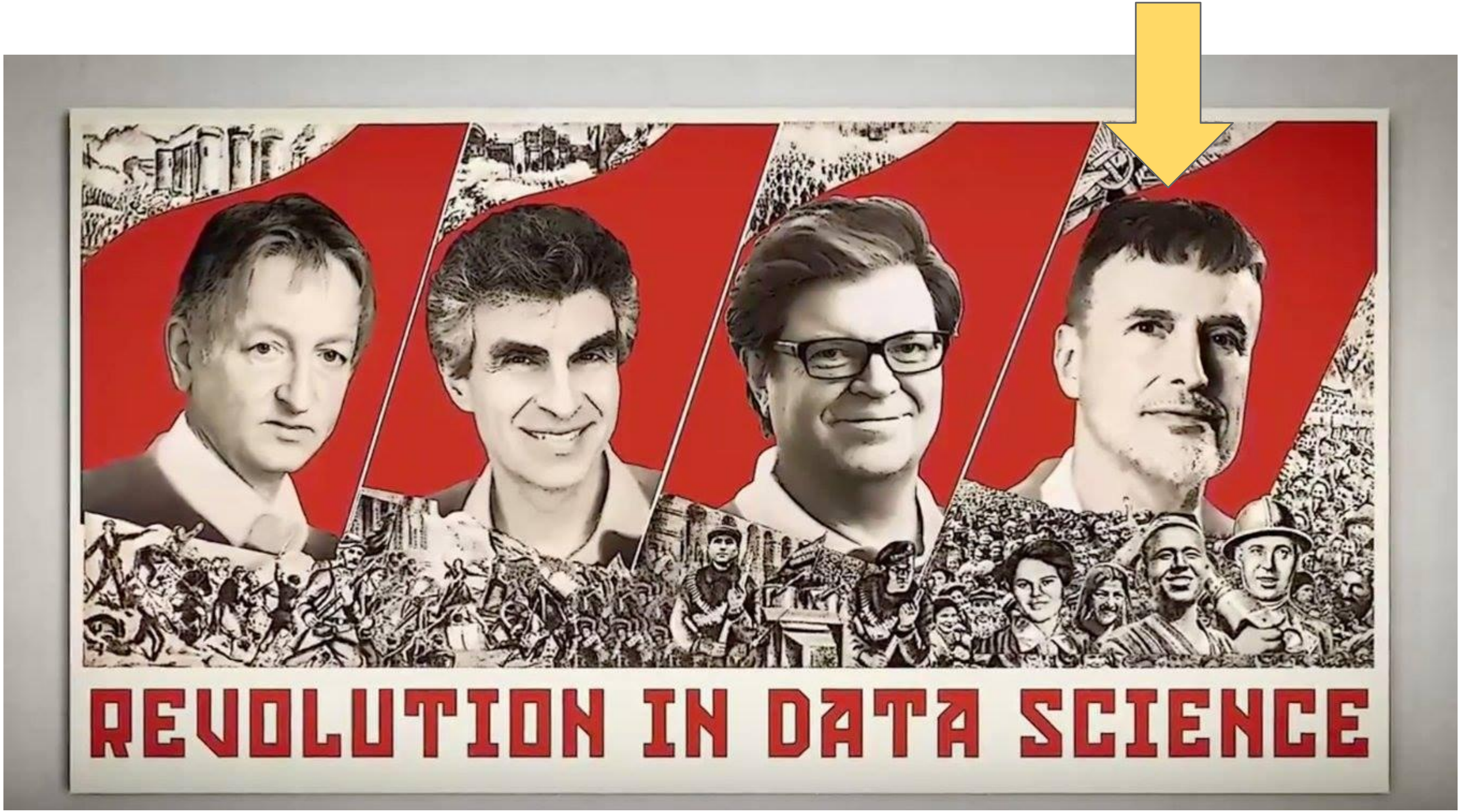


Figure: Michael Bronstein, “Geometric Deep Learning” (ICLR 2021)

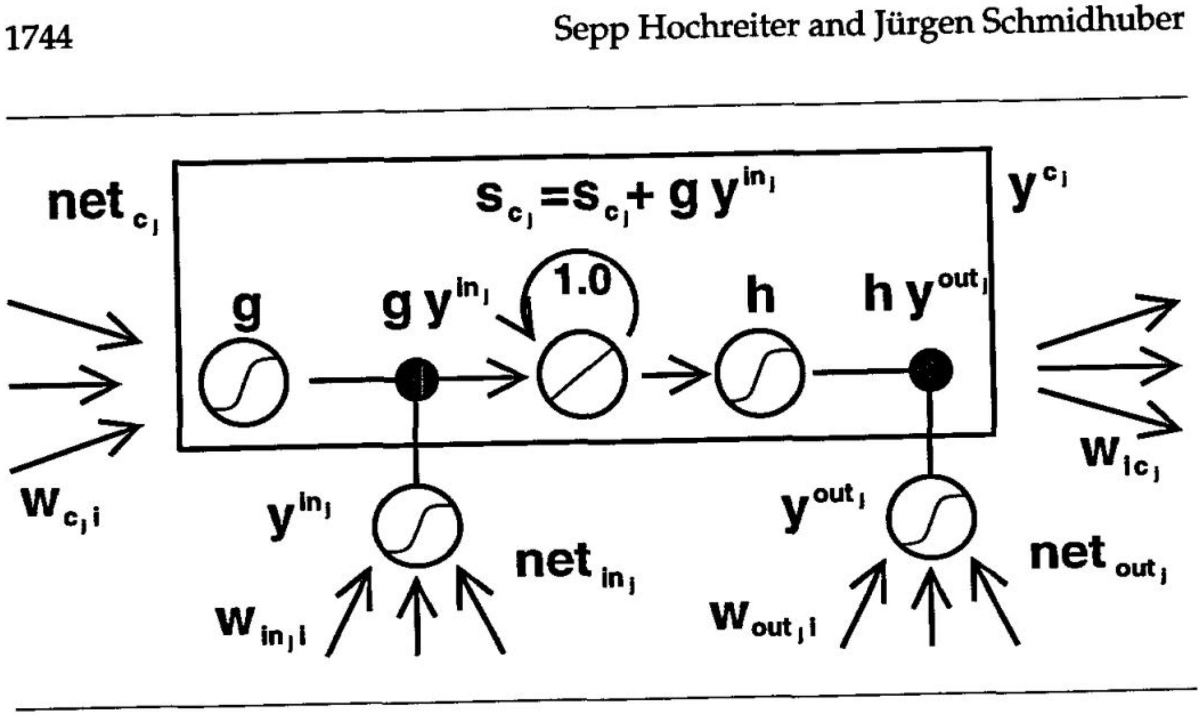
Long Short-Term Memory (LSTM)

The New York Times, “When A.I. Matures, It May Call Jürgen Schmidhuber ‘Dad’” (November 2016)

USI/SUPSI
Istituto
Dalle Molle
di studi
sull’intelligenza
artificiale
IDSIA

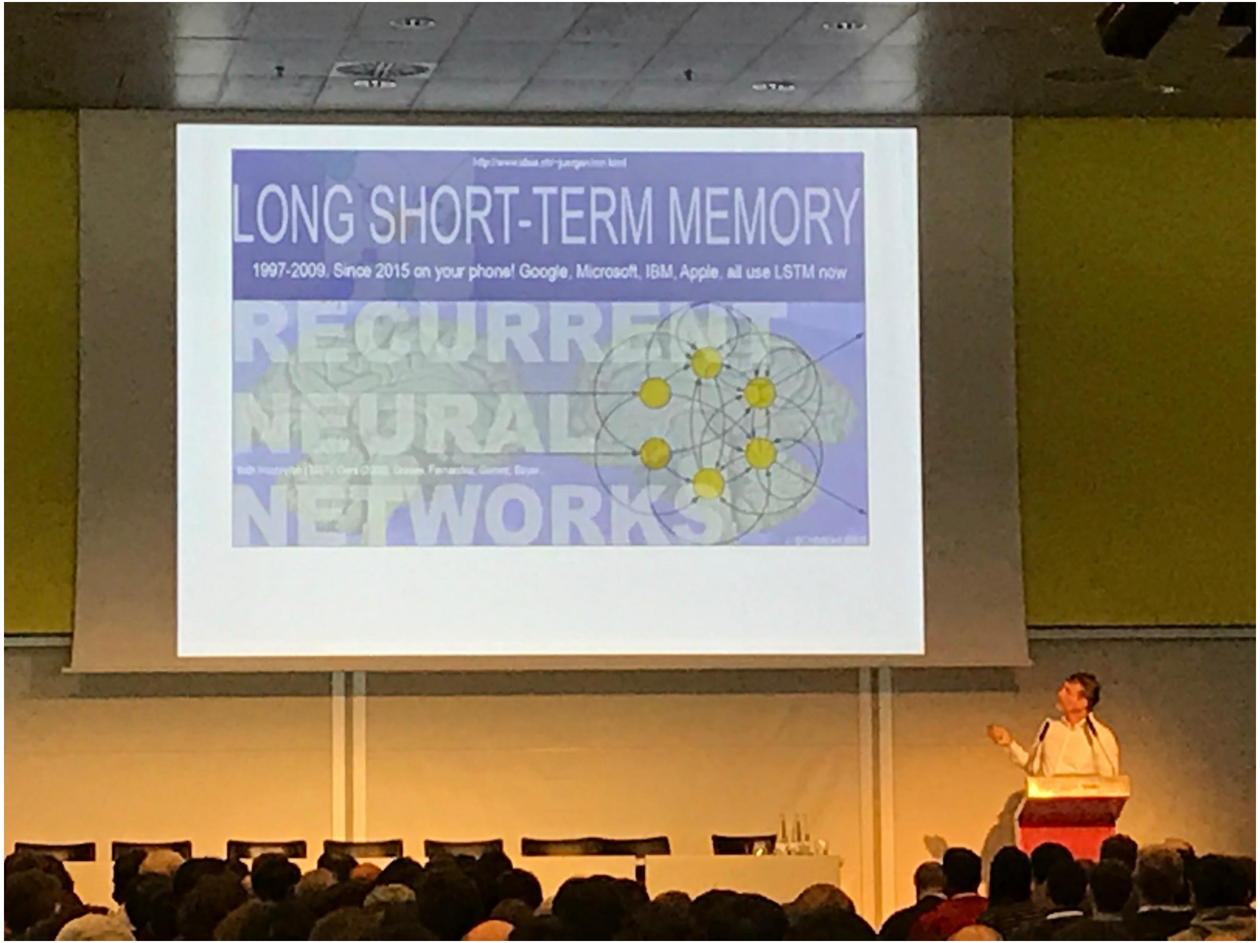
Technische Universität München

Long Short-Term Memory (LSTM)



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

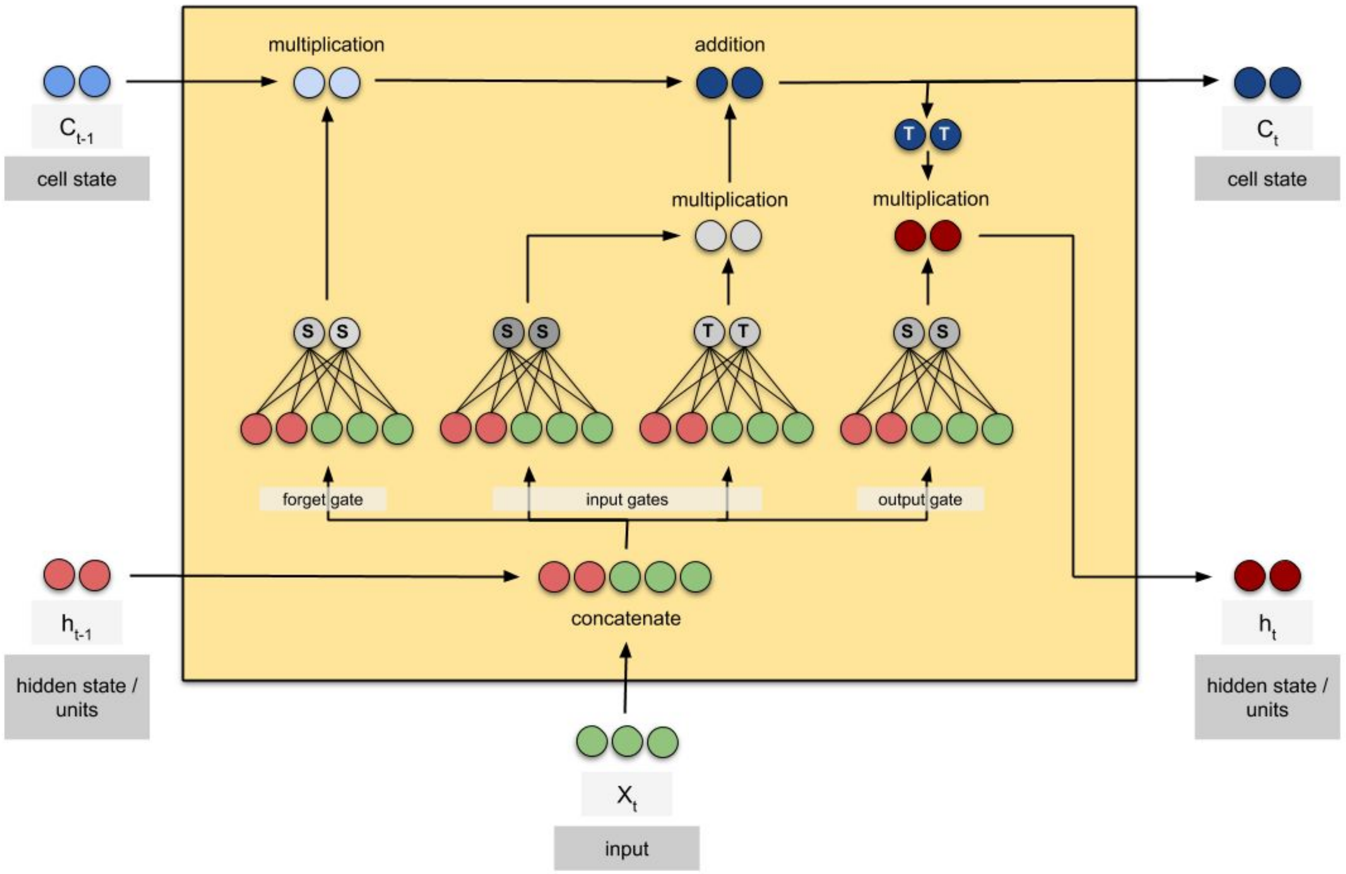
Long Short-Term Memory (LSTM)

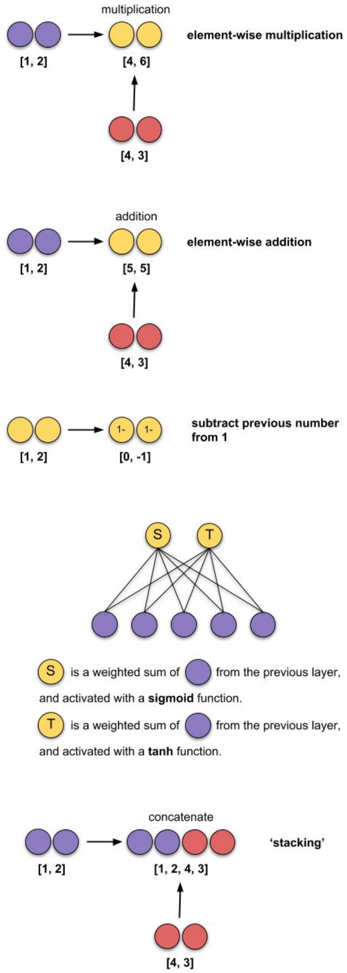


Jürgen Schmidhuber @ NIPS 2016 Barcelona

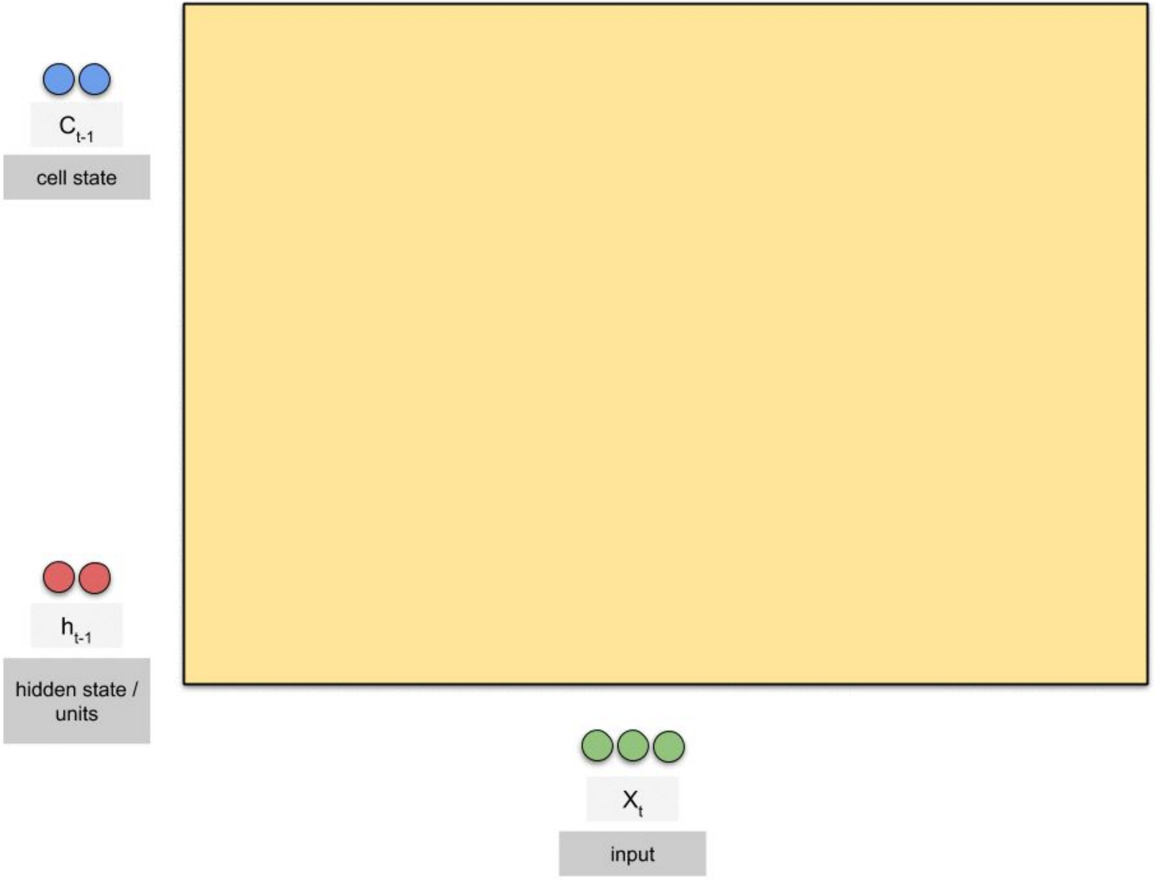
Long Short-Term Memory (LSTM)

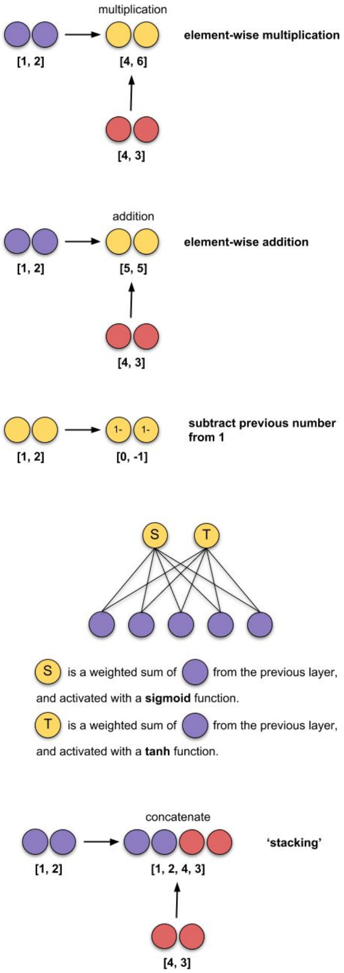
[Jürgen Schmidhuber](#)
@ NIPS 2016 Barcelona





Raimi Karim, [“Animated RNN, LSTM and GRU”](). Towards Data Science 2018.





Raimi Karim, [“Animated RNN, LSTM and GRU”](). Towards Data Science 2018.

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 **forget whatever may not be useful anymore** by clearing that info from the cell state (optimized clearing mechanism)
3. Make every RNN unit able to decide whether **the current time-step information matters or not**, to accept or discard (optimized reading 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)

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

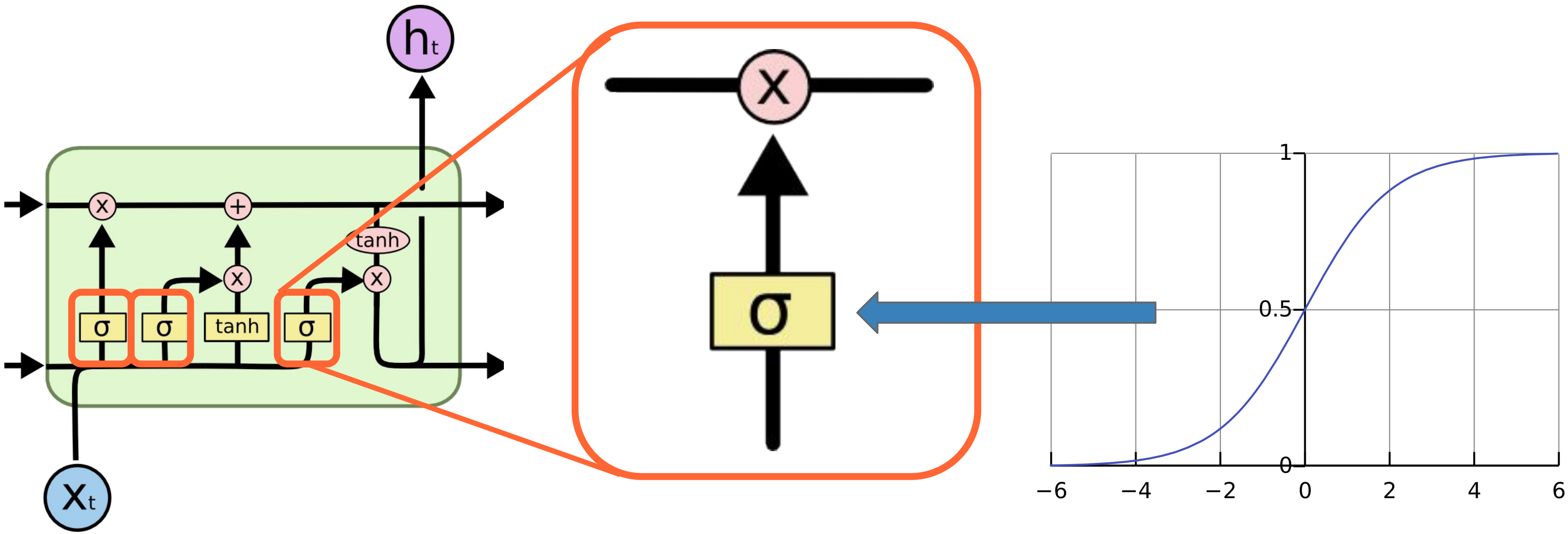
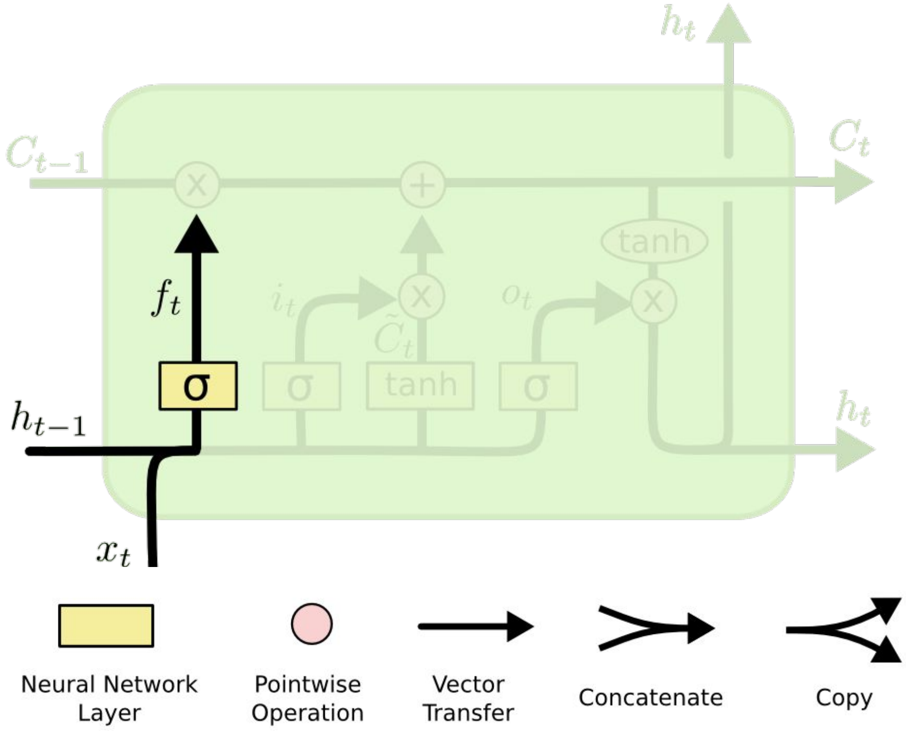


Figure: Cristopher Olah, [“Understanding LSTM Networks”]() (2015)

Long Short-Term Memory (LSTM)

Make every RNN unit able to **forget whatever may not be useful anymore** by clearing that info from the cell state (optimized clearing mechanism)



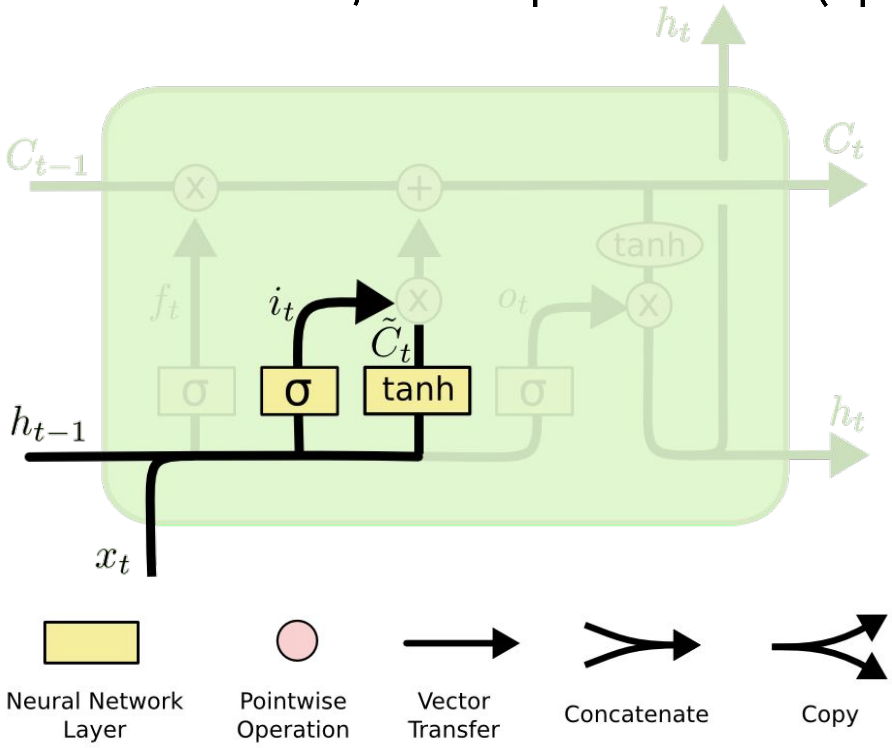
Forget Gate:

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

Figure: Cristopher Olah, "Understanding LSTM Networks" (2015)

Long Short-Term Memory (LSTM)

Make every RNN unit able to decide whether **the current time-step information matters or not**, to accept or discard (optimized reading mechanism)



Input Gate Layer

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

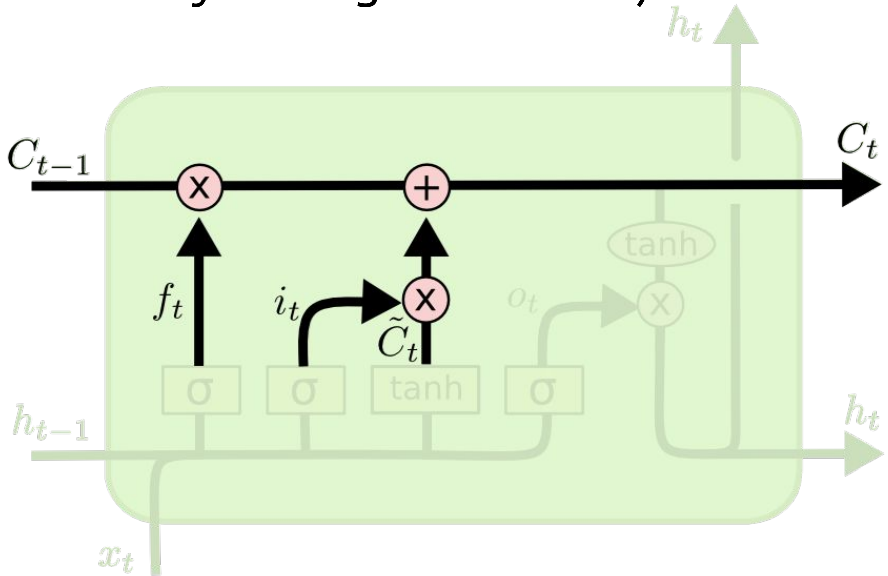
New contribution to cell state

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

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

Long Short-Term Memory (LSTM)

Make every RNN unit able to decide how to **to update the cell state** (optimized memory writing mechanism)



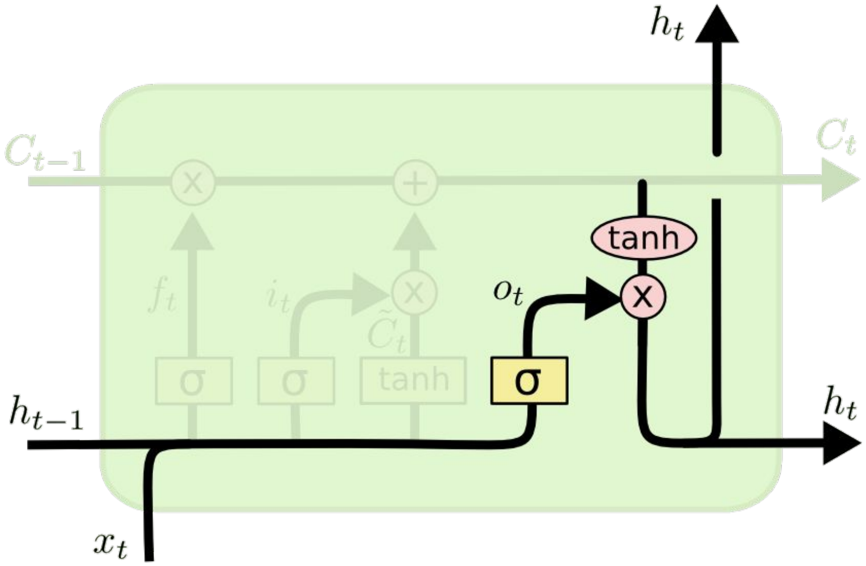
Forget + Input Gates = Update Cell State (memory):

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

Figure: Cristopher Olah, "Understanding LSTM Networks" (2015)

Long Short-Term Memory (LSTM)

Make every RNN unit able to **output the decisions whenever it is ready to do so** (optimized output mechanism)



Output Gate Layer

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

Output to next layer & timestep

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

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

Long Short-Term Memory (LSTM)

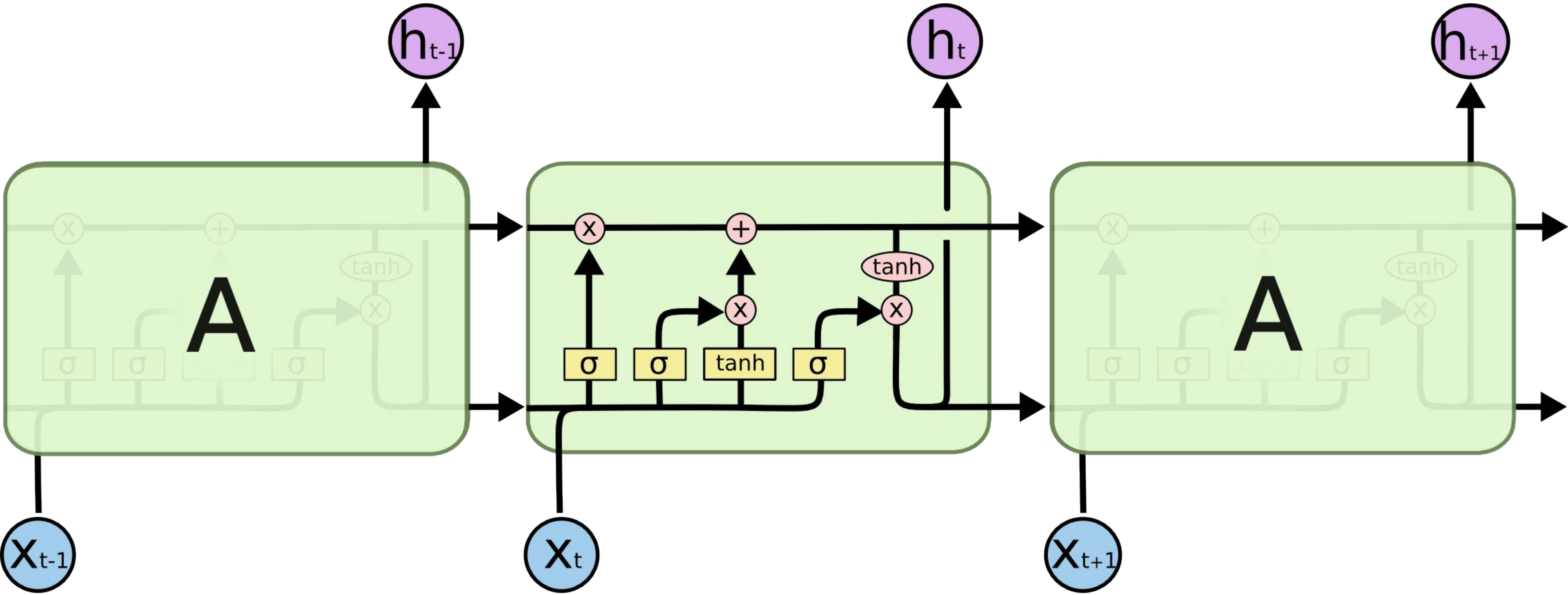
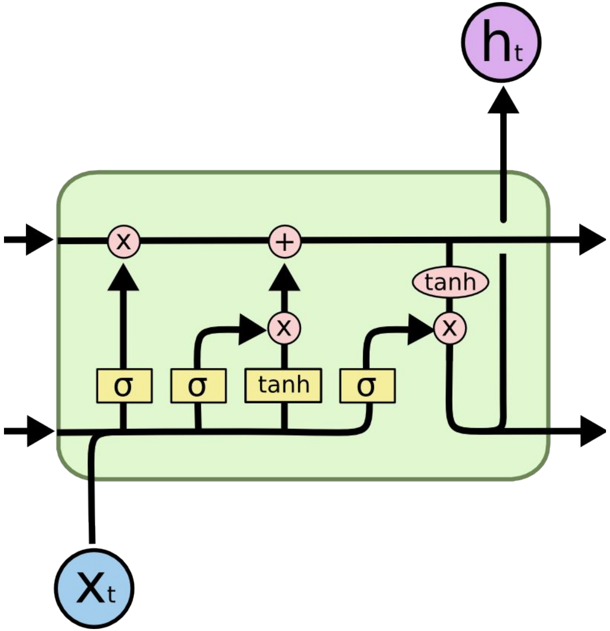


Figure: Cristopher Olah, "Understanding LSTM Networks" (2015)

Long Short-Term Memory (LSTM)



Compared to a non-gated RNN, an LSTM has four times more parameters because of the additional neurons that govern the gates:

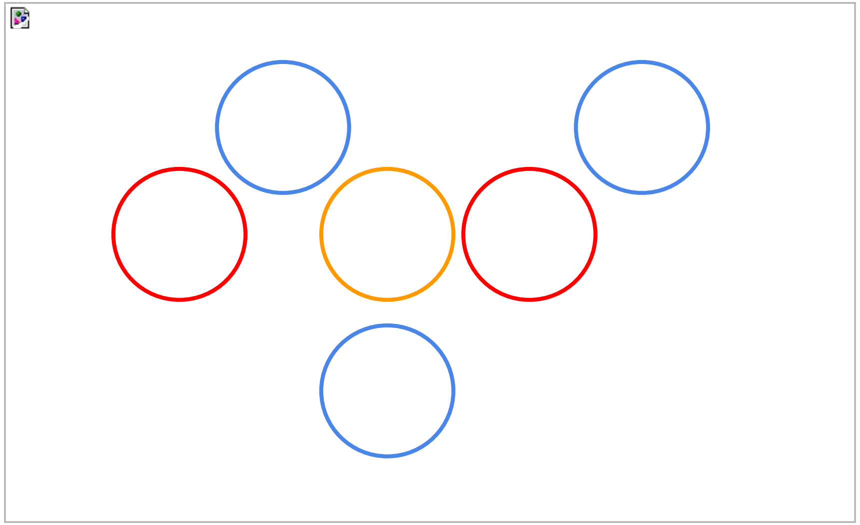
$$N^i_{params} = 4 \times (N^i_{inputs} \times N^i_{units} + N^i_{units} \times N^i_{units} + N^i_{units})$$

3 sigmoid gates +
input activation (tanh in the figure)

Long Short Term Memory (LSTM) cell

Updating an LSTM cell requires 6 computations:

1. Gates
2. Activation units
3. Cell state



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

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

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

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

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

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

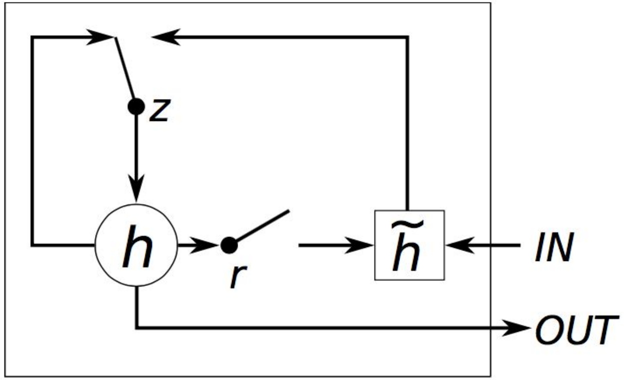
Computation Flow

Outline

1. Motivation
2. Recurrent Neural Layer
3. Forward & Backward Passes Through Time
4. Gated Recurrent Neurons
    a. LSTM
    b. GRU

Gated Recurrent Unit (GRU)

GRU obtain a similar performance as LSTM with one gate less.



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

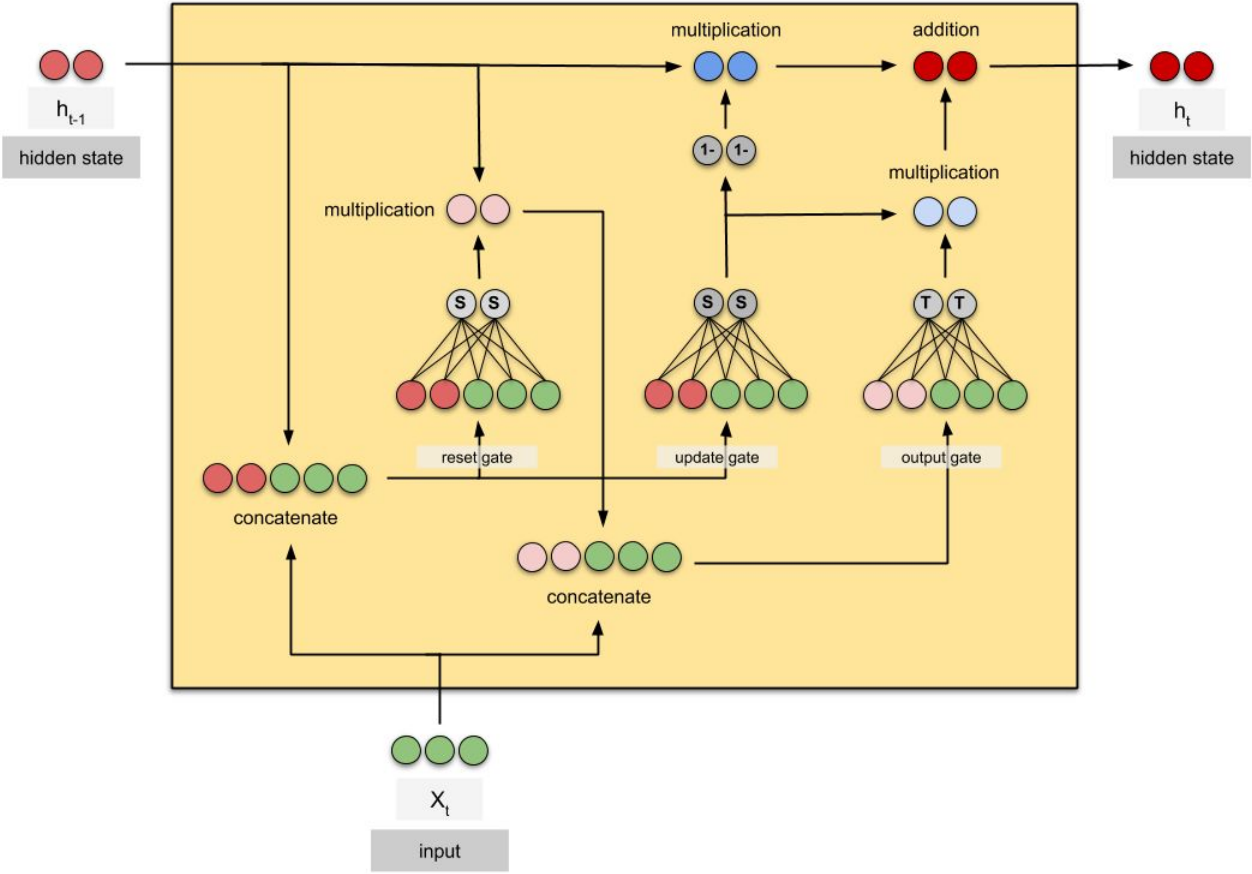
$$r_i = \sigma\left(W^{(r)}x_i + U^{(r)}h_{i-1} + b^{(r)}\right) \quad (2)$$

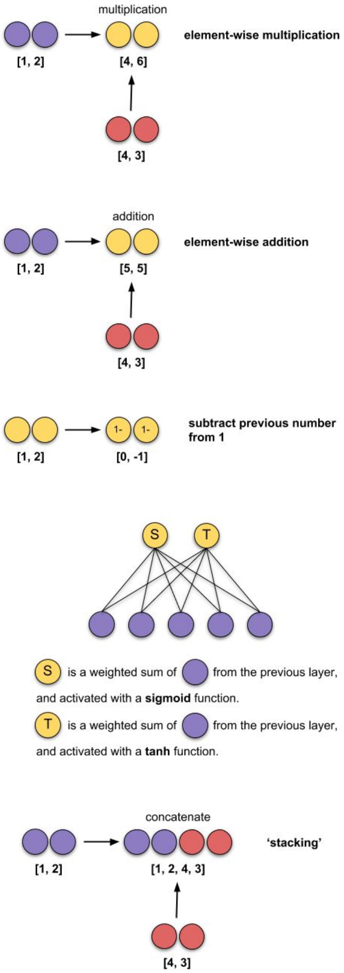
$$\tilde{h}_i = \tanh\left(Wx_i + r_i \circ Uh_{i-1} + b^{(h)}\right) \quad (3)$$

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

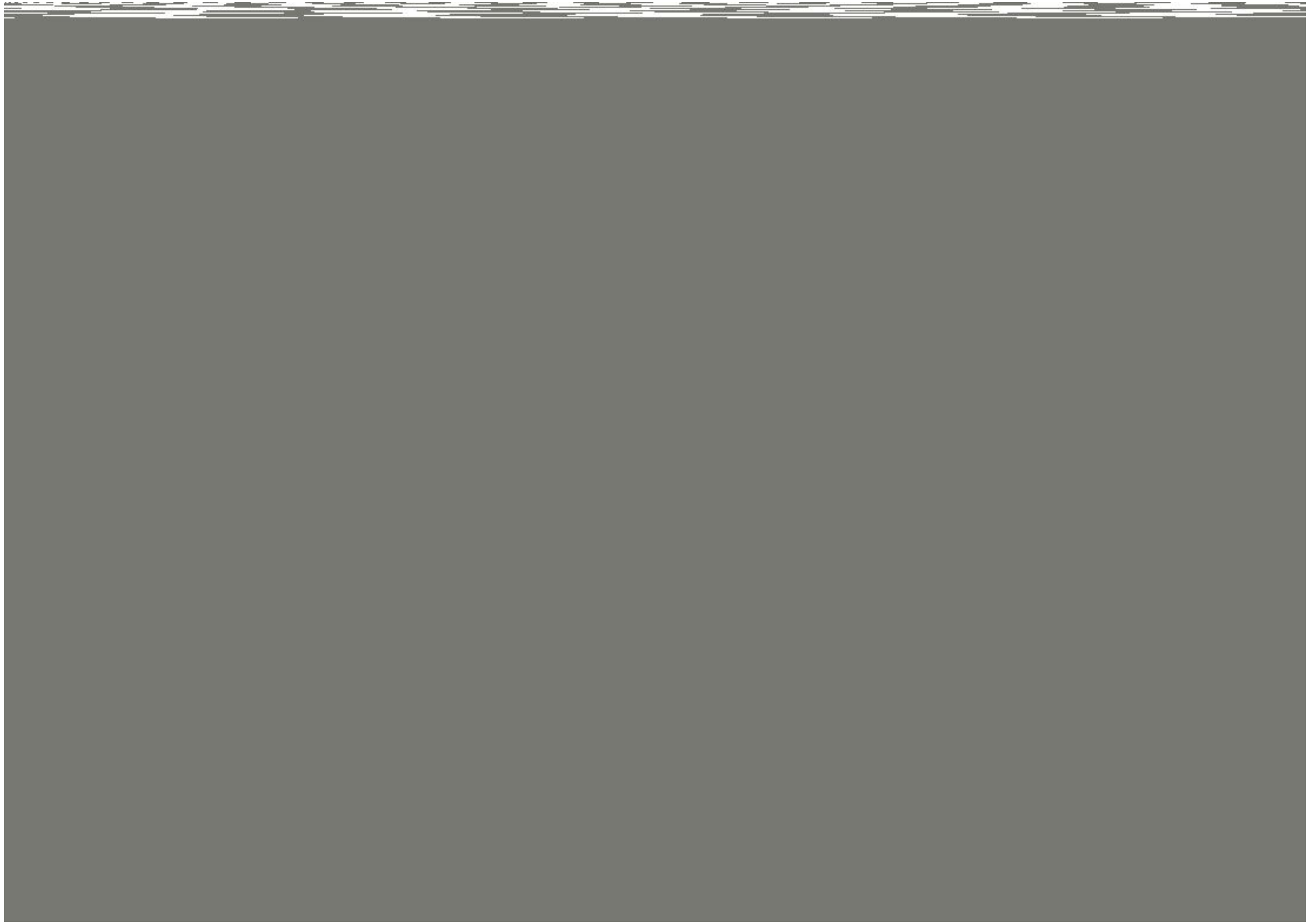
$$N^i_{params} = 3 \times (N^i_{inputs} \times N^i_{units} + N^i_{units} \times N^i_{units} + N^i_{units})$$

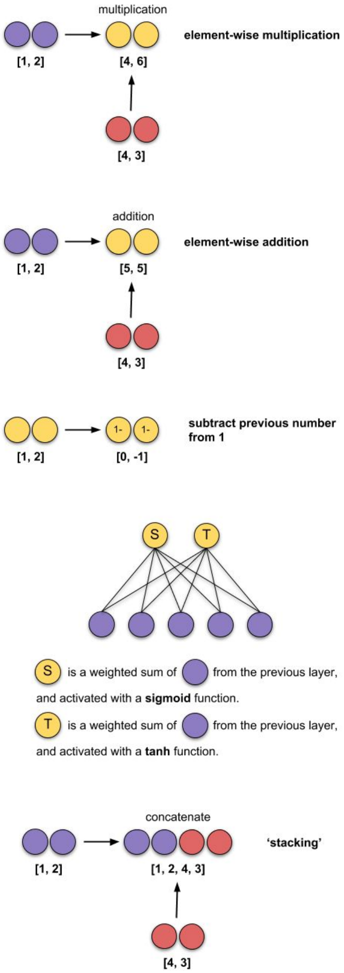
#GRU Cho, Kyunghyun, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." EMNLP 2014.





Raimi Karim, ["Animated RNN, LSTM and GRU"](). Towards Data Science 2018.





Raimi Karim, [“Animated RNN, LSTM and GRU”](). Towards Data Science 2018.

Outline

1. Motivation
2. Recurrent Neural Layer
3. Forward & Backward Passes Through Time
4. Gated Recurrent Neurons
    a. LSTM
    b. GRU
5. Stacked RNNs

Deep (stacked) RNNs

Recurrent layers can be stacked as any non-recurrent layer.

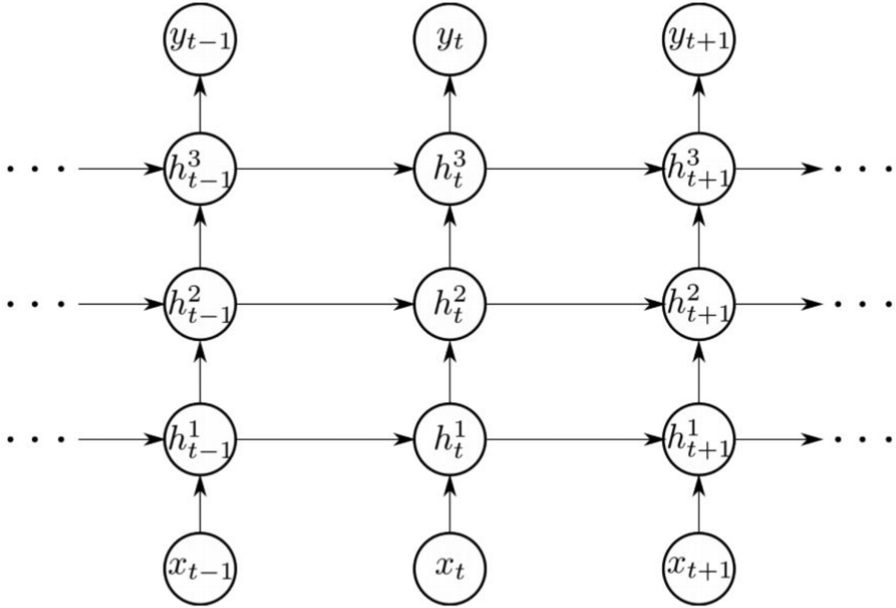


Figure: Lambert (Stanford University).

Pascanu, R., Gulcehre, C., Cho, K., & Bengio, Y. (2013). How to construct deep recurrent neural networks. arXiv preprint arXiv:1312.6026.

Bidirectional RNN (BRNN)

Sequences can be process in both directions by stacking two RNN layers.

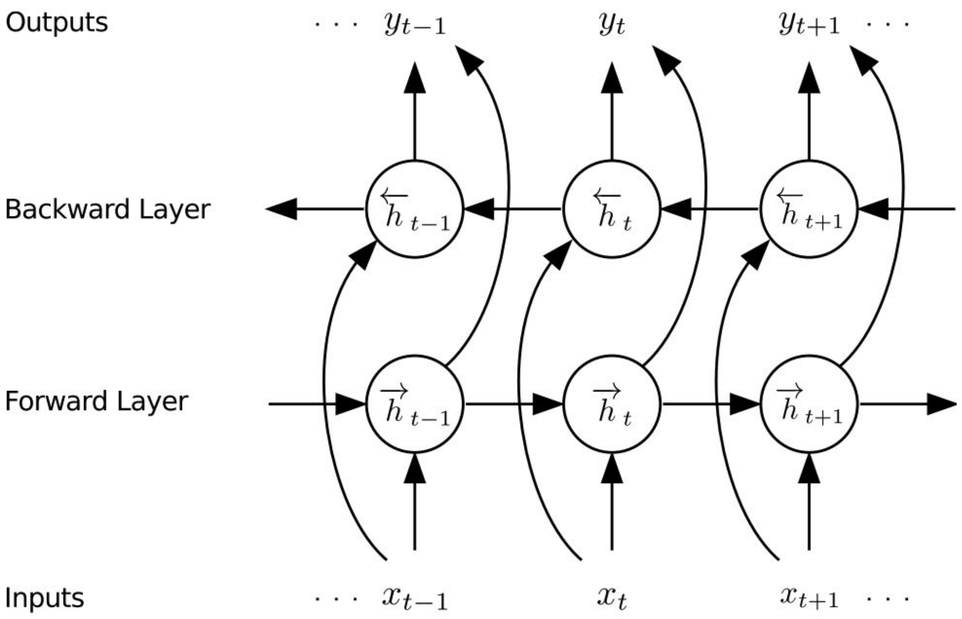


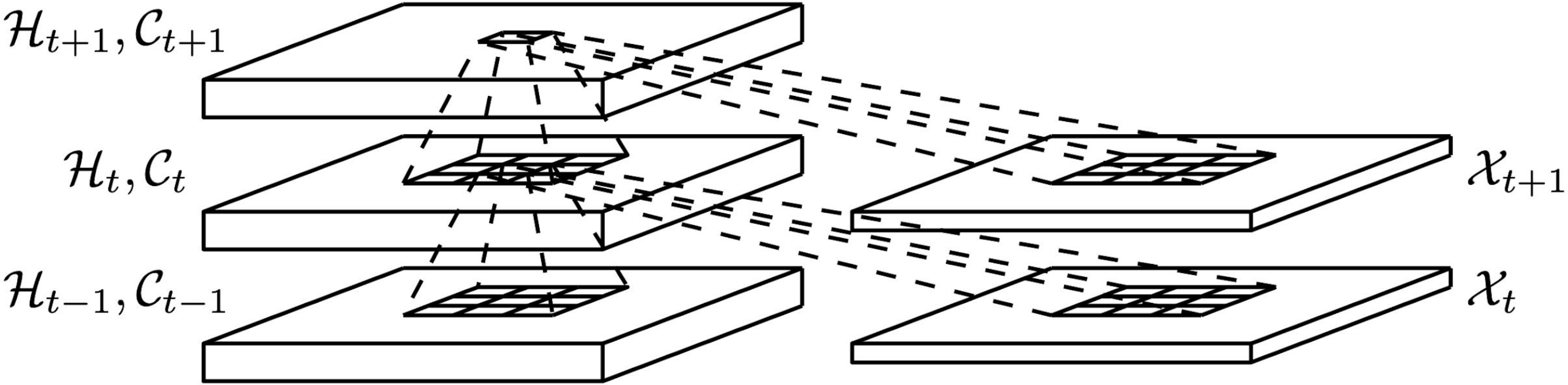
Figure:
Graves et al. 2013.

#BRNN Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. IEEE transactions on Signal Processing, 45(11), 2673-2681.

Outline

1. Motivation
2. Recurrent Neural Layer
3. Forward & Backward Passes Through Time
4. Gated Recurrent Neurons
   a. LSTM
   b. GRU
5. Stacked RNNs
6. Advanced architectures

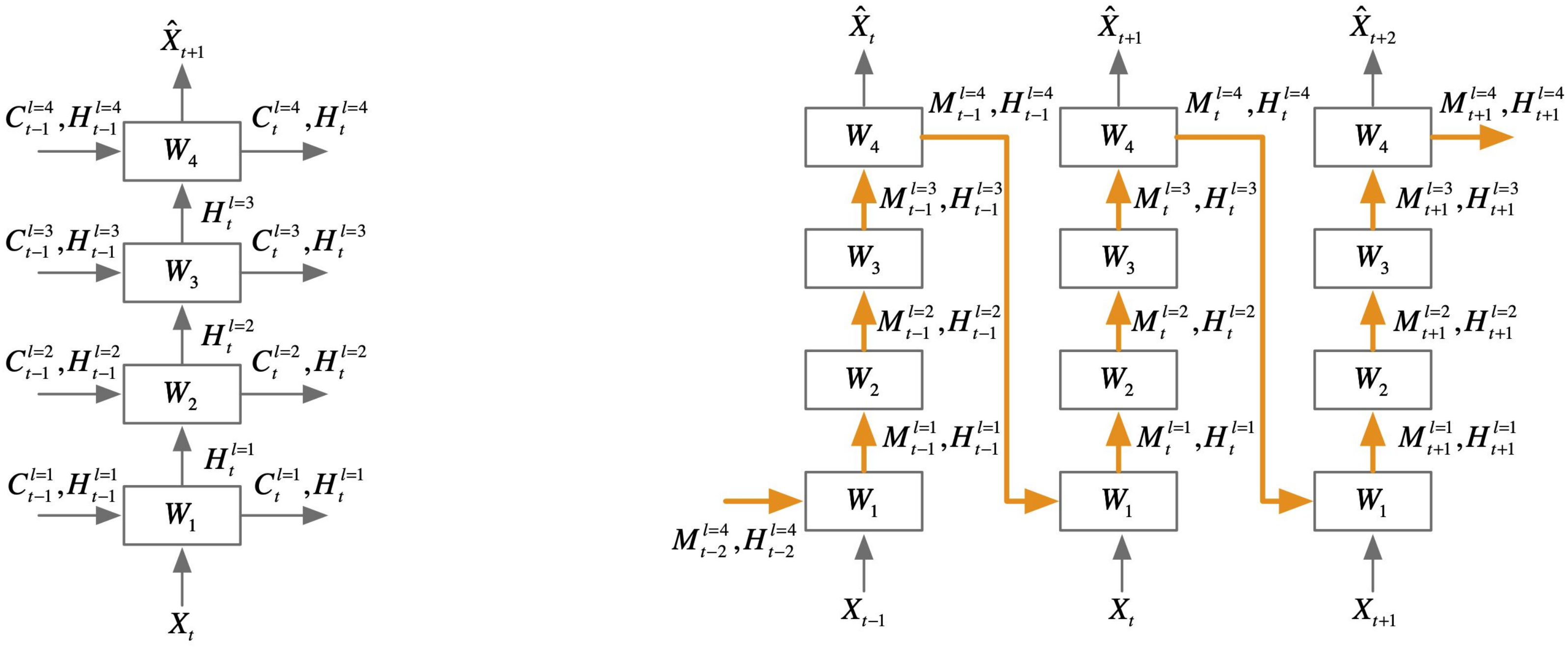
Conv-LSTM



#ConvLSTM Xingjian, S. H. I., Chen, Z., Wang, H., Yeung, D. Y., Wong, W. K., & Woo, W. C. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. NIPS 2015.

Spatiotemporal LSTM (ST-LSTM)

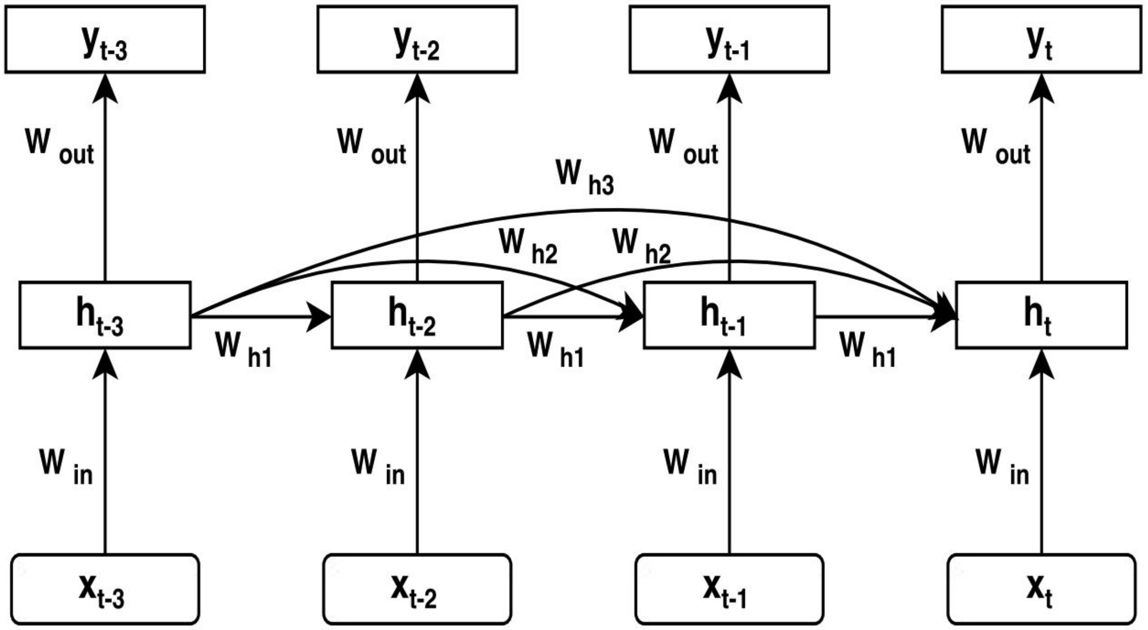
Cell states are no longer constrained inside each LSTM unit. Instead, they are allowed to zigzag in two directions: across stacked RNN layers vertically and through all RNN states horizontally



#ST-LSTM Wang, Y., Long, M., Wang, J., Gao, Z., & Philip, S. Y. [Predrnn: Recurrent neural networks for predictive learning using spatiotemporal LSTMs](). NIPS 2017.

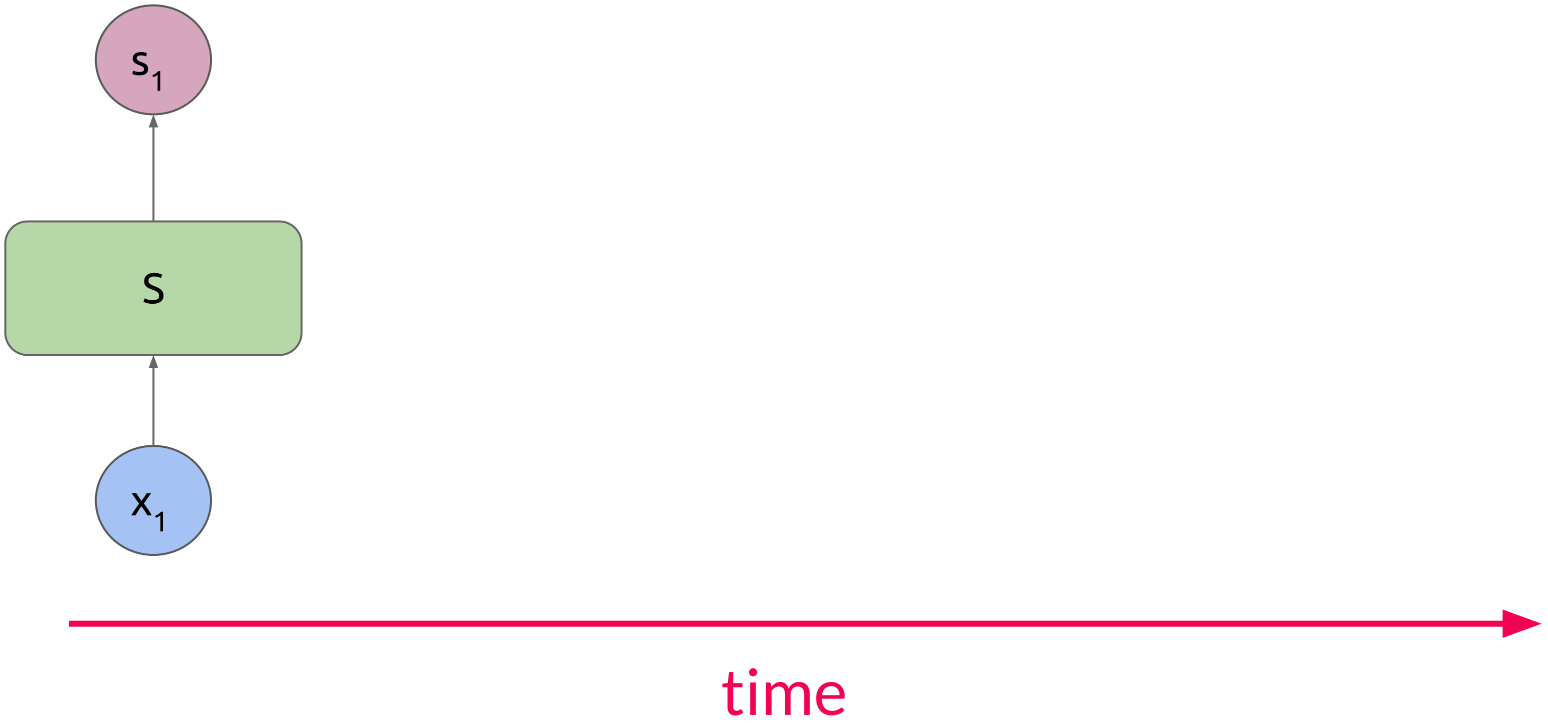
Higher Order RNN

More memory units can be added to connect with further temporal states.



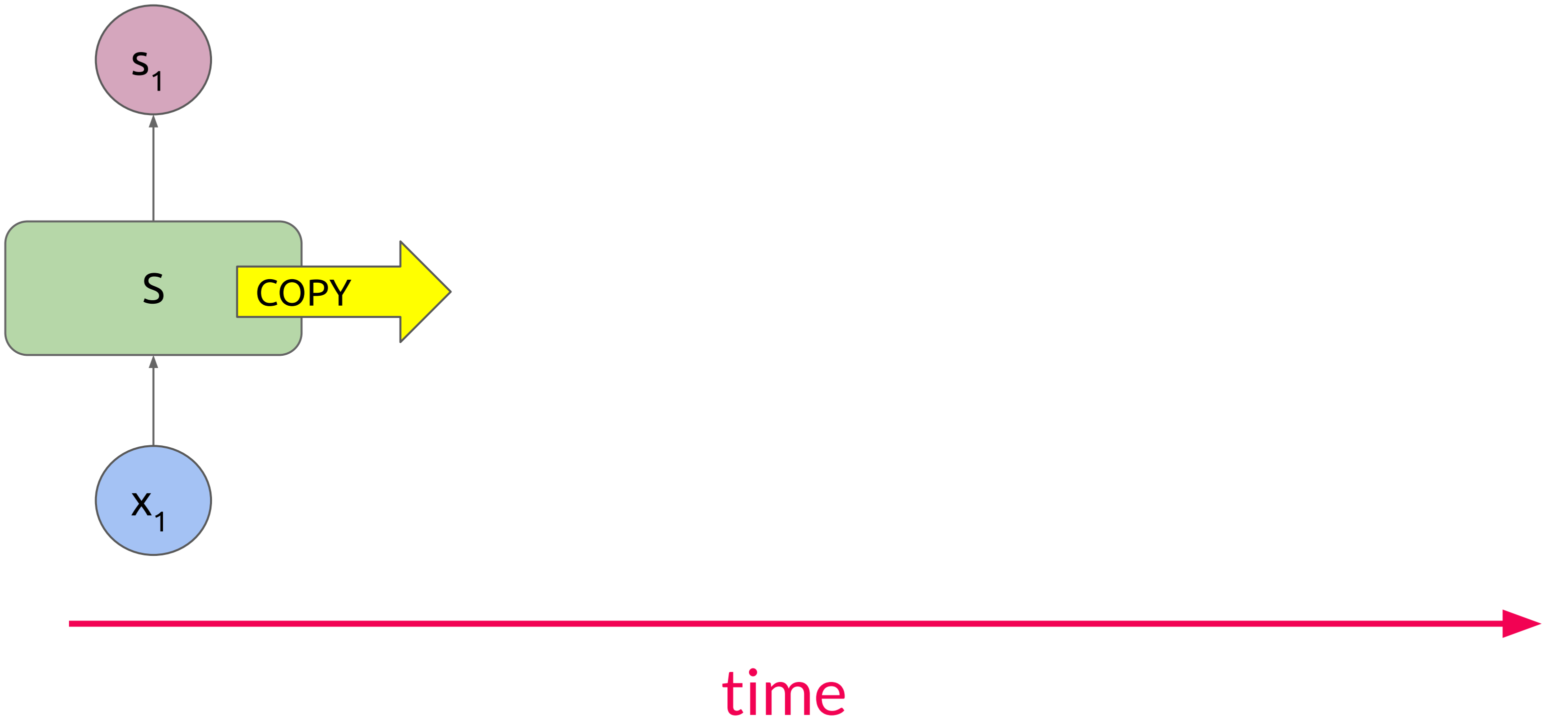
#HORNN Soltani, Rohollah, and Hui Jiang. ["Higher order recurrent neural networks."]() arXiv preprint arXiv:1605.00064 (2016).

SkipRNN



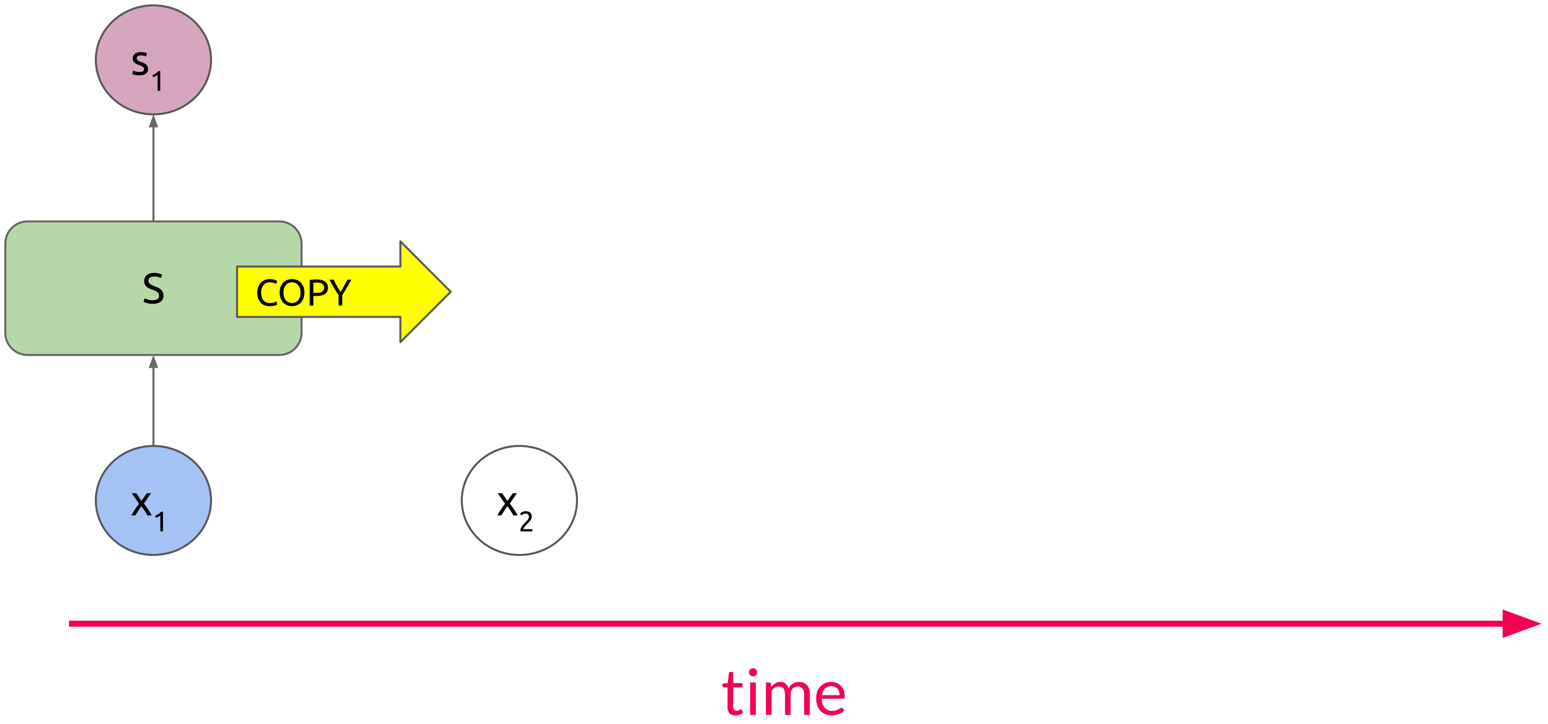
#SkipRNN Victor Campos, Brendan Jou, Xavier Giro-i-Nieto, Jordi Torres, and Shih-Fu Chang. "Skip RNN: Learning to Skip State Updates in Recurrent Neural Networks", ICLR 2018.

SkipRNN



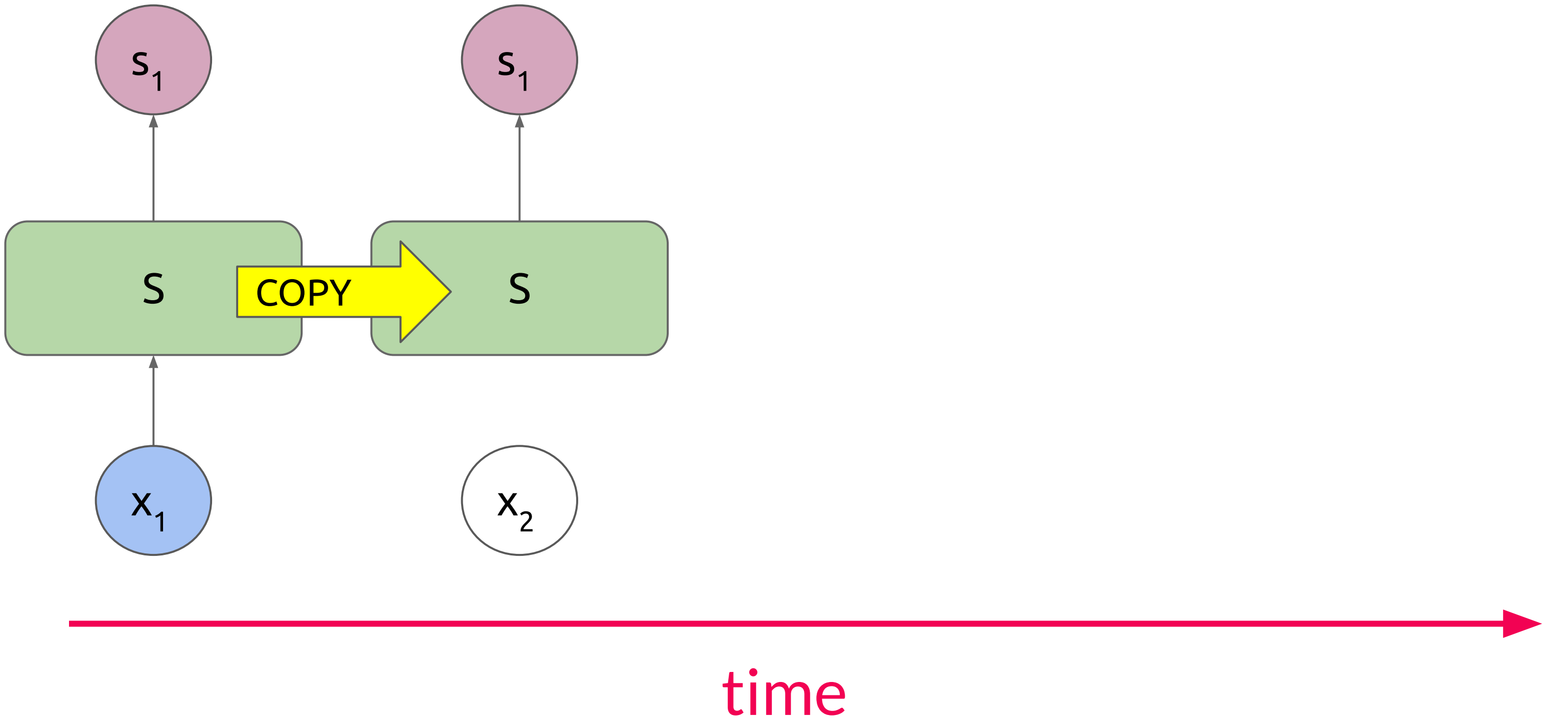
#SkipRNN Victor Campos, Brendan Jou, Xavier Giro-i-Nieto, Jordi Torres, and Shih-Fu Chang. [“Skip RNN: Learning to Skip State Updates in Recurrent Neural Networks”](), ICLR 2018.

SkipRNN



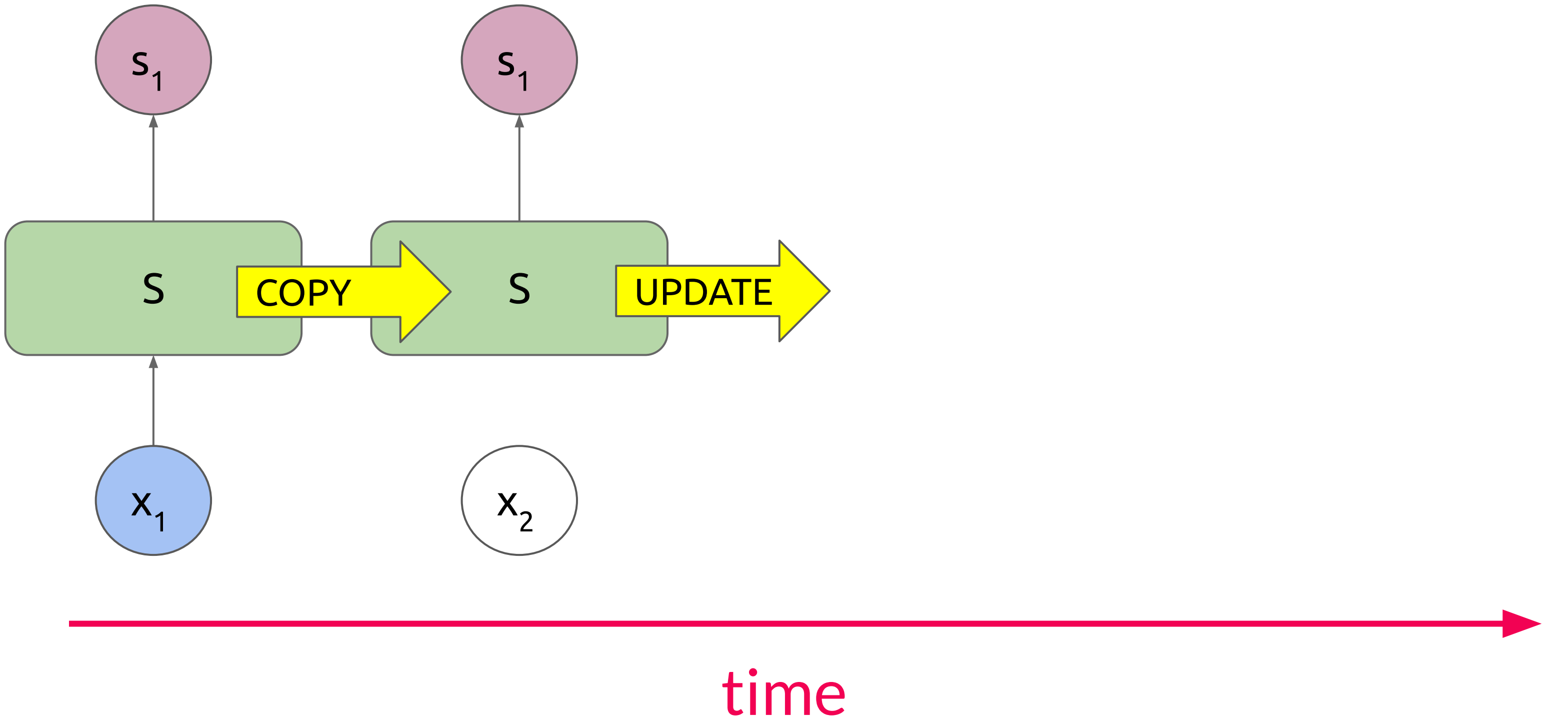
#SkipRNN Victor Campos, Brendan Jou, Xavier Giro-i-Nieto, Jordi Torres, and Shih-Fu Chang. "Skip RNN: Learning to Skip State Updates in Recurrent Neural Networks", ICLR 2018.

SkipRNN



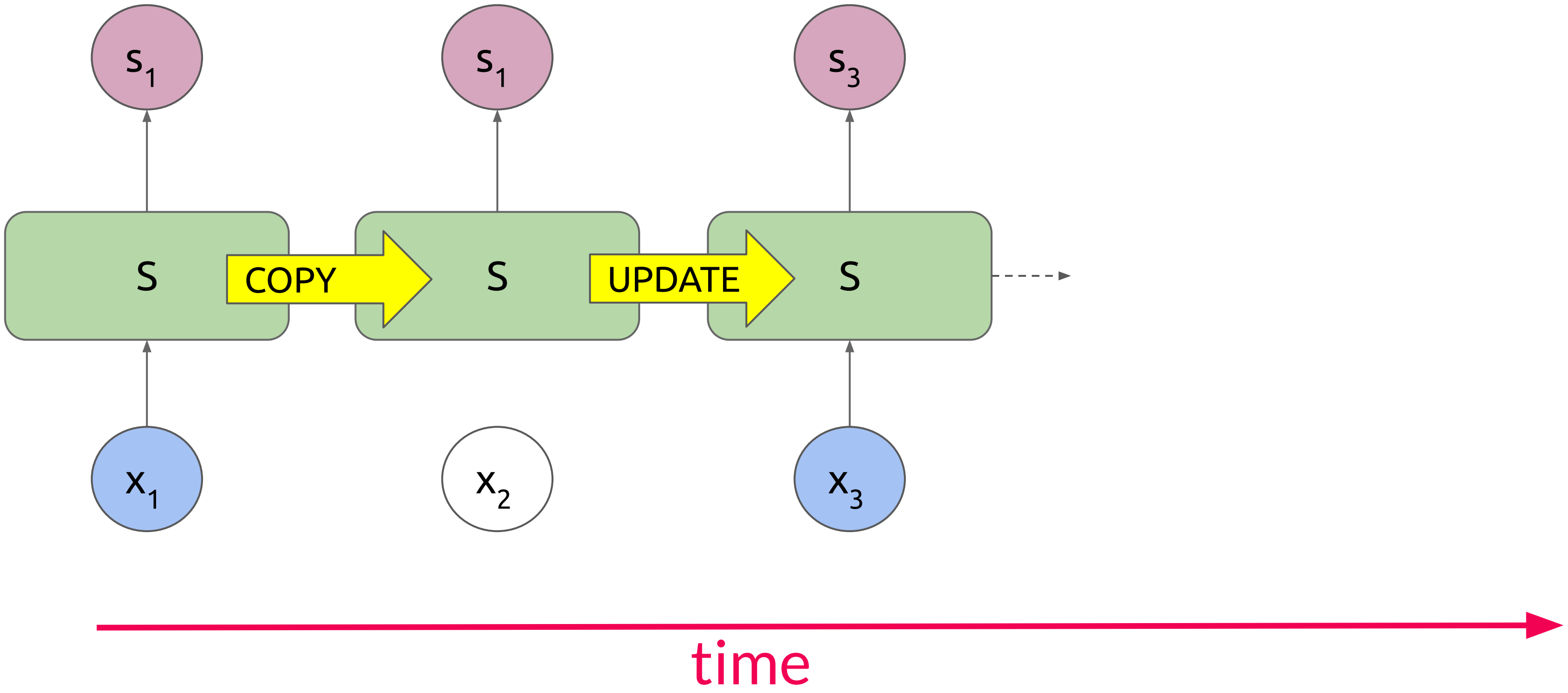
#SkipRNN Victor Campos, Brendan Jou, Xavier Giro-i-Nieto, Jordi Torres, and Shih-Fu Chang. "Skip RNN: Learning to Skip State Updates in Recurrent Neural Networks", ICLR 2018.

SkipRNN



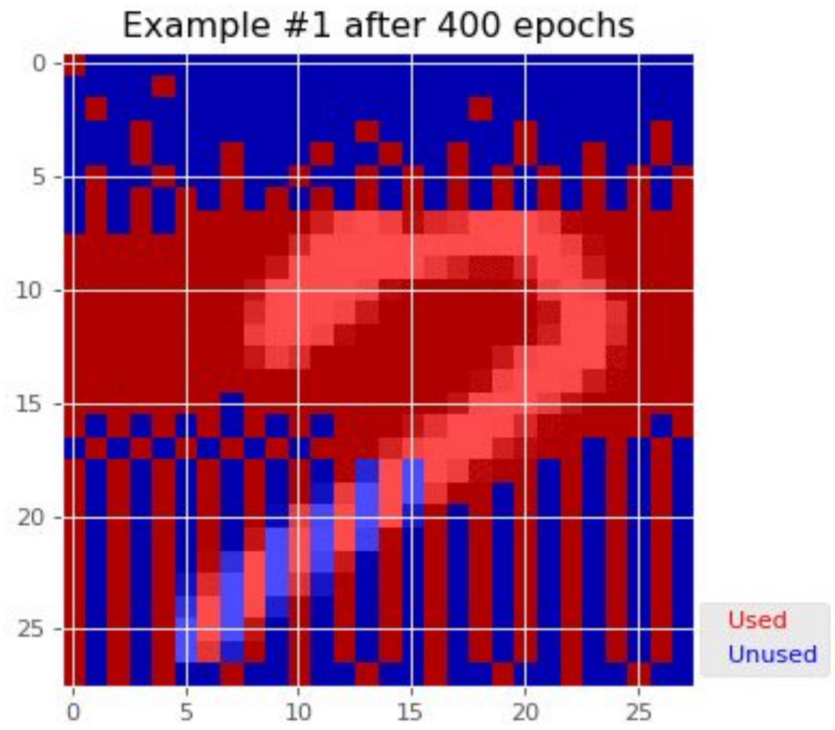
#SkipRNN Victor Campos, Brendan Jou, Xavier Giro-i-Nieto, Jordi Torres, and Shih-Fu Chang. "Skip RNN: Learning to Skip State Updates in Recurrent Neural Networks", ICLR 2018.

SkipRNN



#SkipRNN Victor Campos, Brendan Jou, Xavier Giro-i-Nieto, Jordi Torres, and Shih-Fu Chang. ["Skip RNN: Learning to Skip State Updates in Recurrent Neural Networks"](), ICLR 2018.

SkipRNN

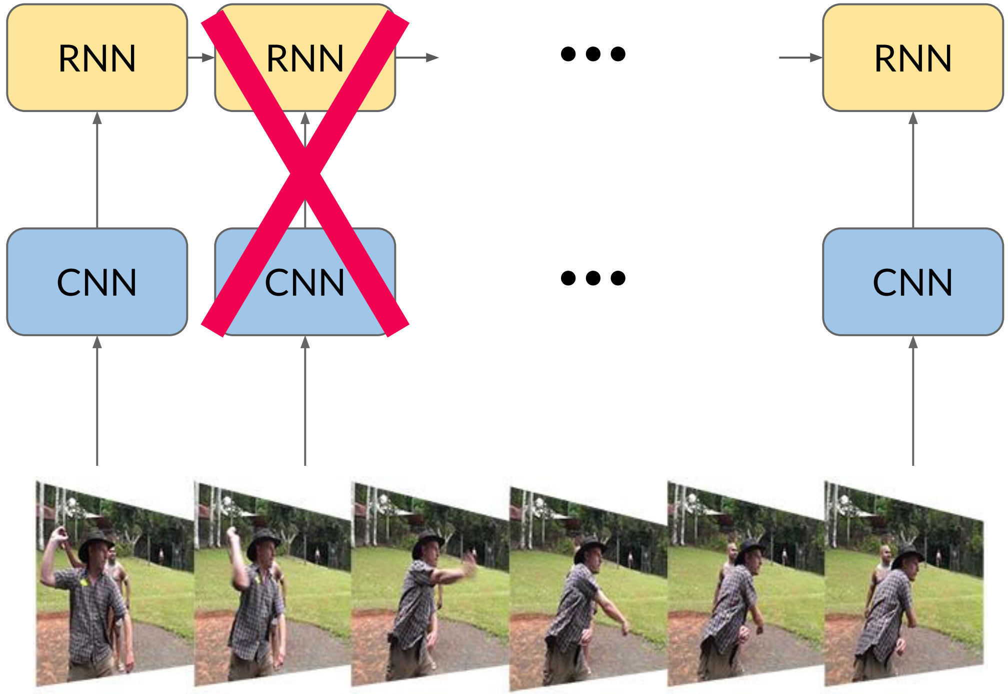


~95% acc

Used
Unused

#SkipRNN Victor Campos, Brendan Jou, Xavier Giro-i-Nieto, Jordi Torres, and Shih-Fu Chang. [“Skip RNN: Learning to Skip State Updates in Recurrent Neural Networks”](), ICLR 2018.

SkipRNN



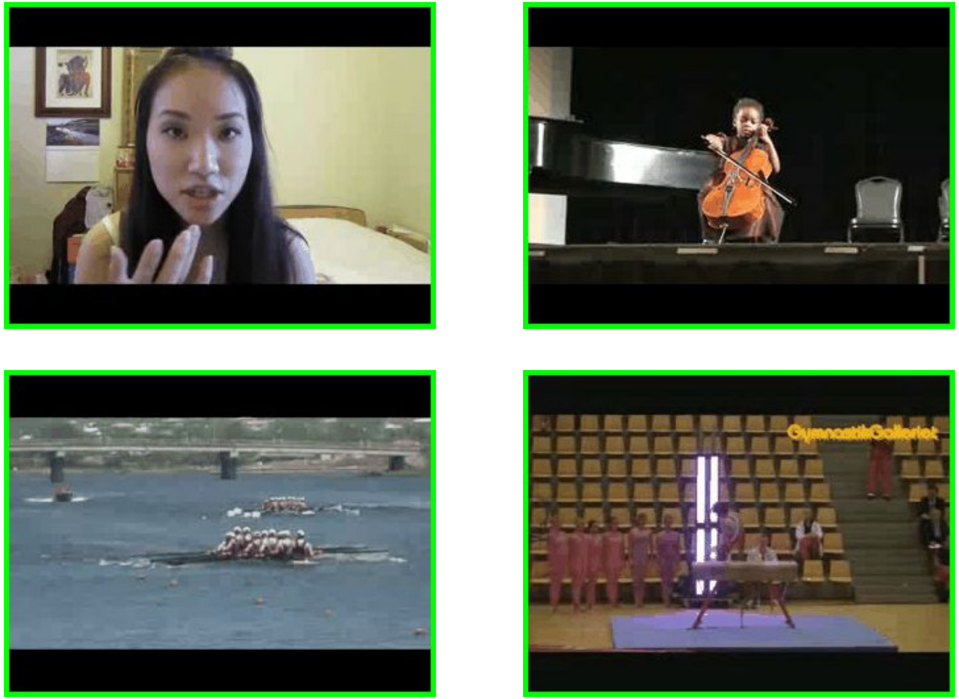
After processing a frame, let the RNN decide how many future frames can be skipped

In skipped frames, simply copy the output and state from the previous time step

There is no ground truth for which frames can be skipped. The RNN **learns** it by itself during training!

#SkipRNN Victor Campos, Brendan Jou, Xavier Giro-i-Nieto, Jordi Torres, and Shih-Fu Chang. "Skip RNN: Learning to Skip State Updates in Recurrent Neural Networks", ICLR 2018.

SkipRNN



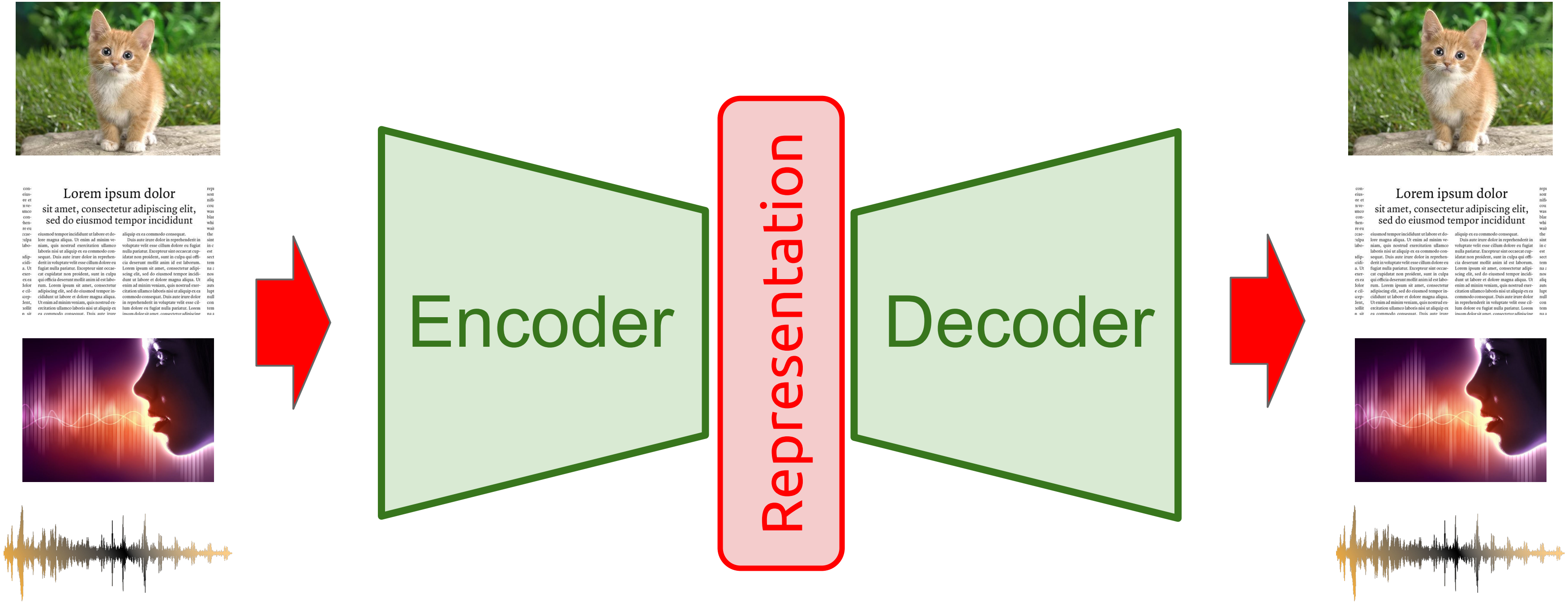
Used
Unused

#SkipRNN Victor Campos, Brendan Jou, Xavier Giro-i-Nieto, Jordi Torres, and Shih-Fu Chang. "Skip RNN: Learning to Skip State Updates in Recurrent Neural Networks", ICLR 2018.

Outline

1. Motivation
2. Recurrent Neural Layer
3. Forward & Backward Passes Through Time
4. Gated Recurrent Neurons
    a. LSTM
    b. GRU
5. Advanced Architectures
6. Encoder-Decoder Architectures

Encoder-Decoder Applications



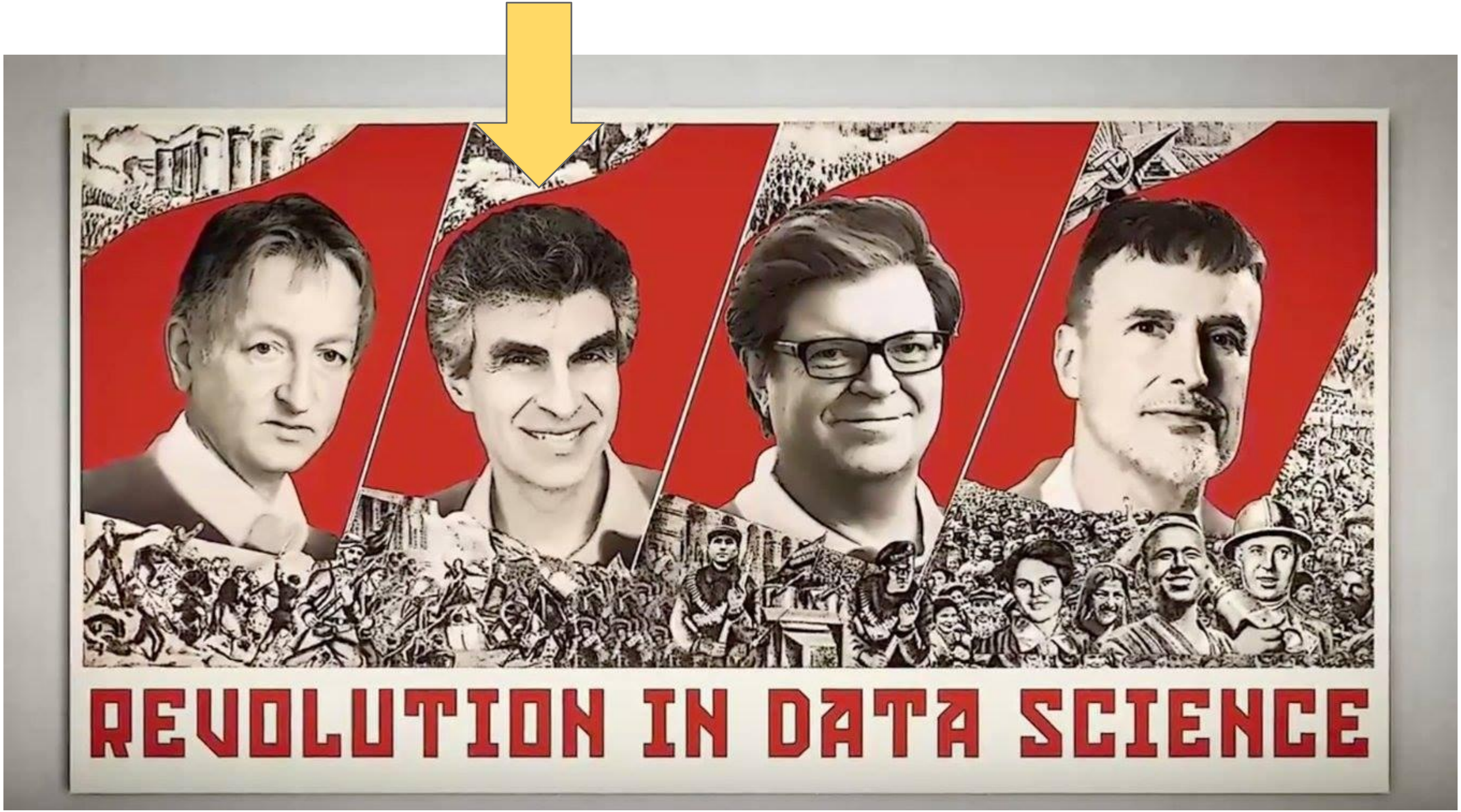


Figure: Michael Bronstein, "Geometric Deep Learning" (ICLR 2021)

Encoding of a Sequence of Words

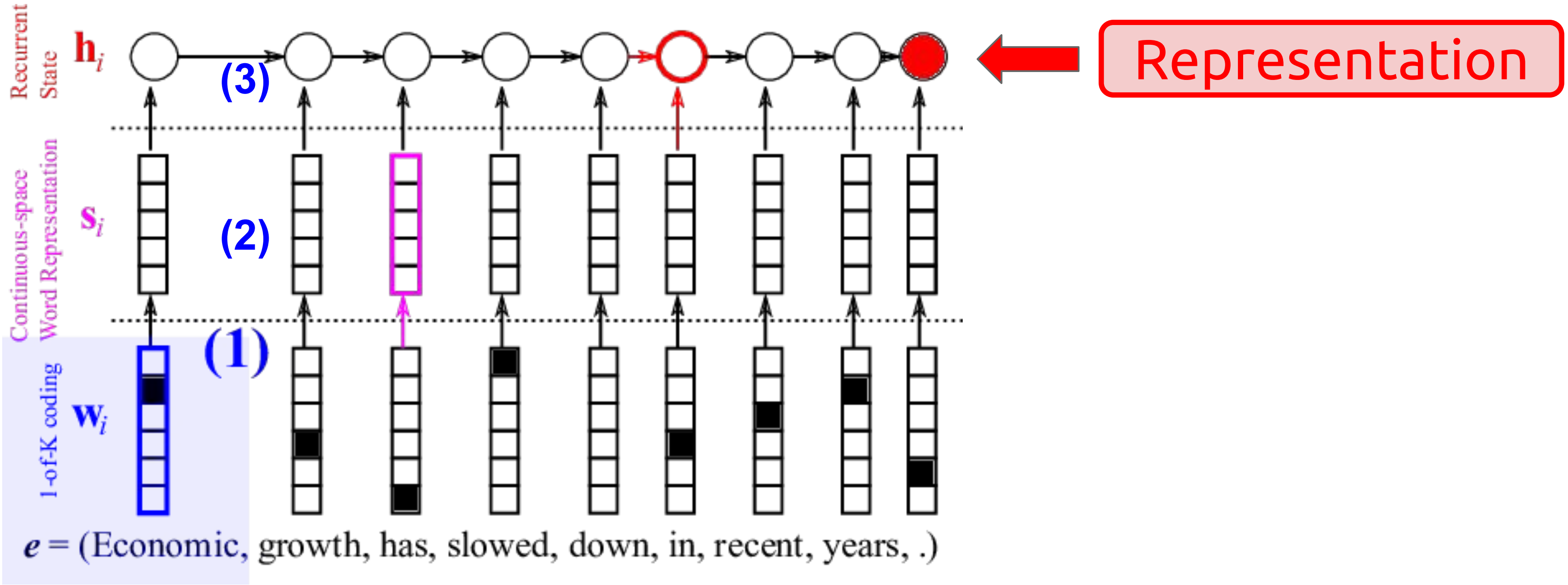
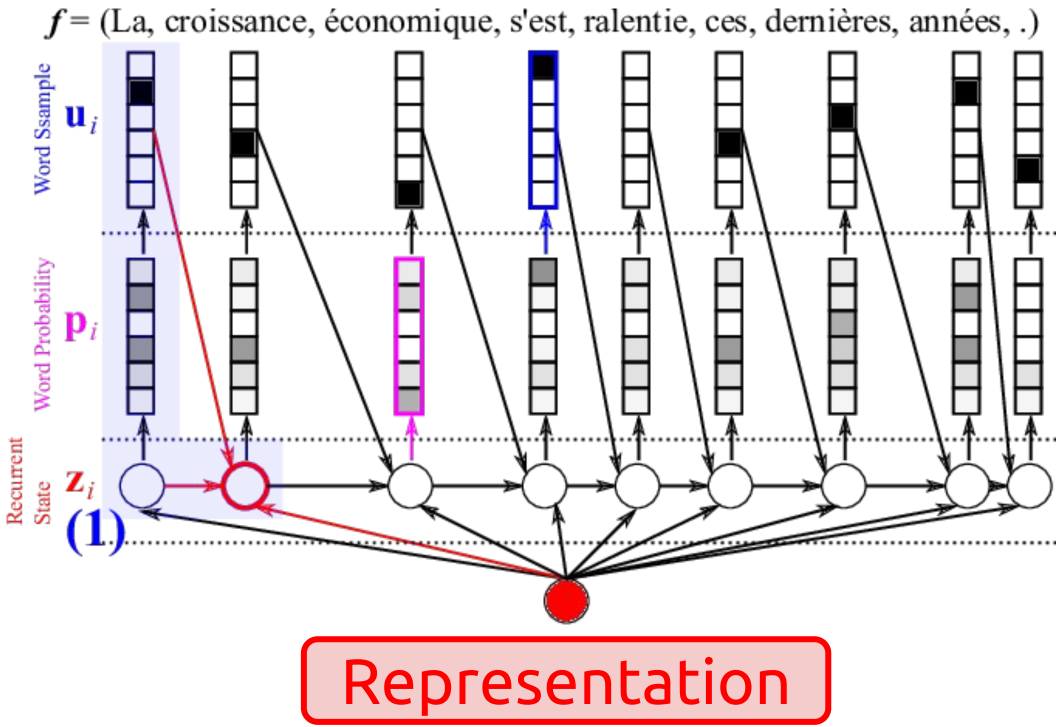


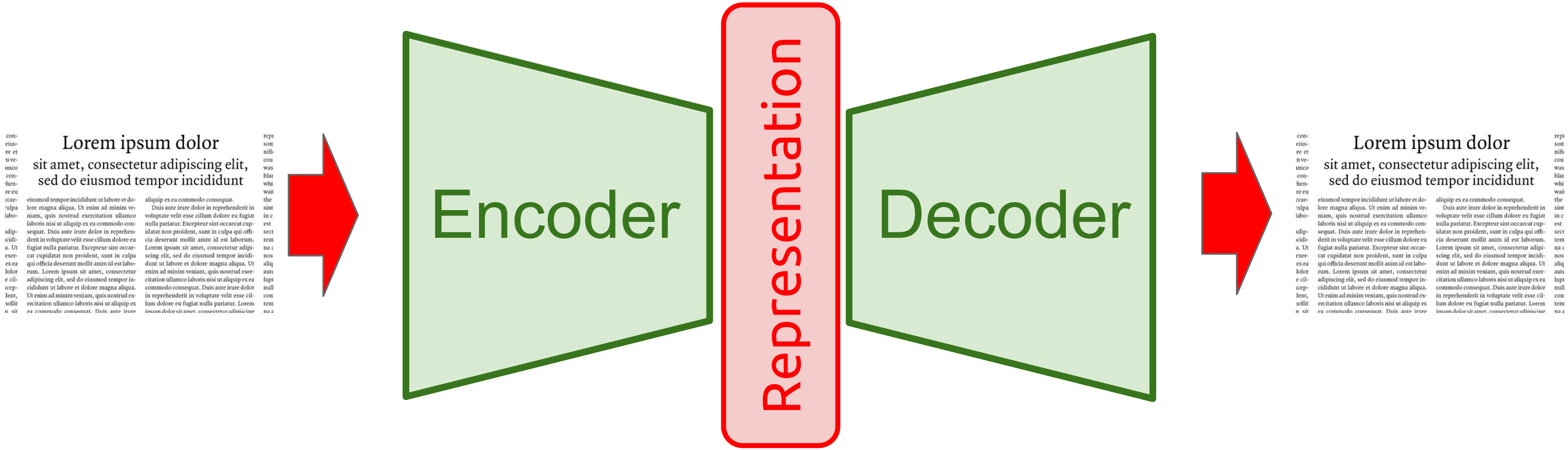
Fig: Kyunghyun Cho, "Introduction to Neural Machine Translation with GPUs" (2015)

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." EMNLP 2014.

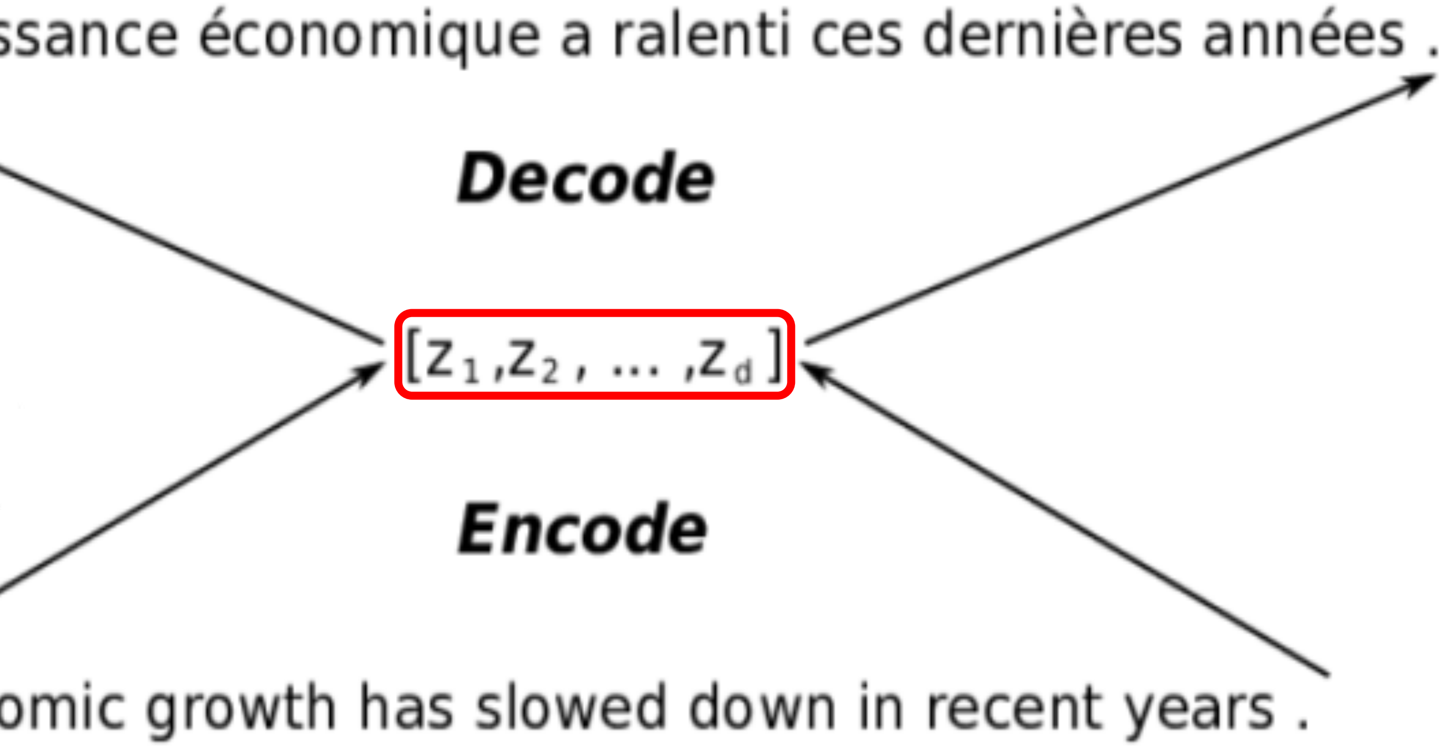
Decoding of a Sequence of Words



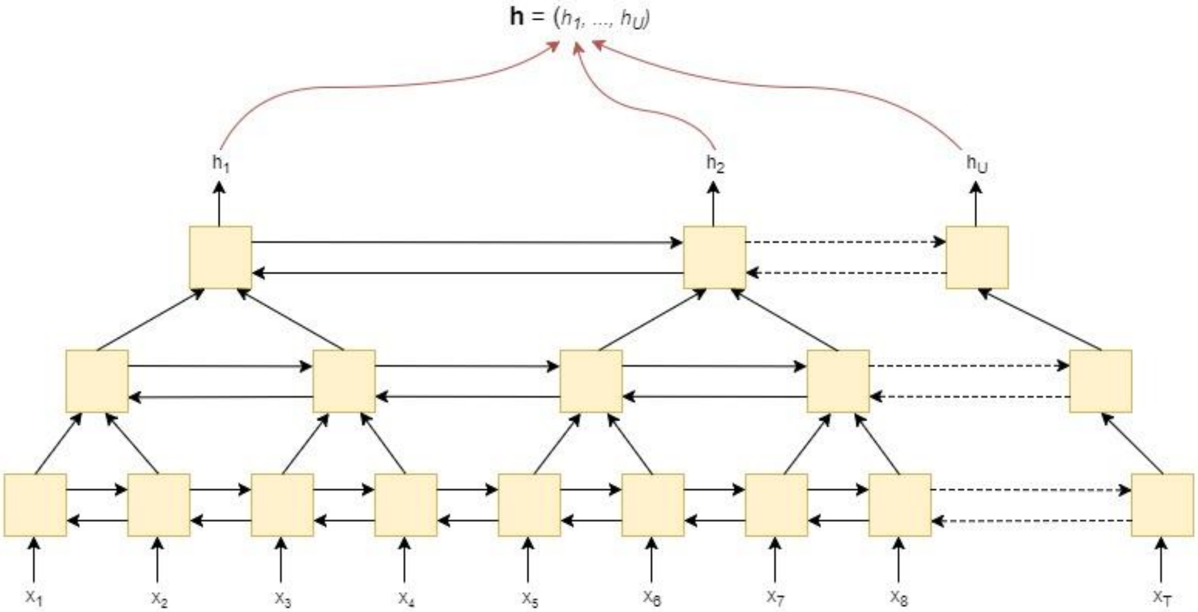
Kyunghyun Cho, [“Introduction to Neural Machine Translation with GPUs”](#) (2015)


Lorem ipsum dolor
sit amet, consectetur adipiscing elit,
sed do eiusmod tempor incididunt
Encoder
Representation
Decoder
Lorem ipsum dolor
sit amet, consectetur adipiscing elit,
sed do eiusmod tempor incididunt

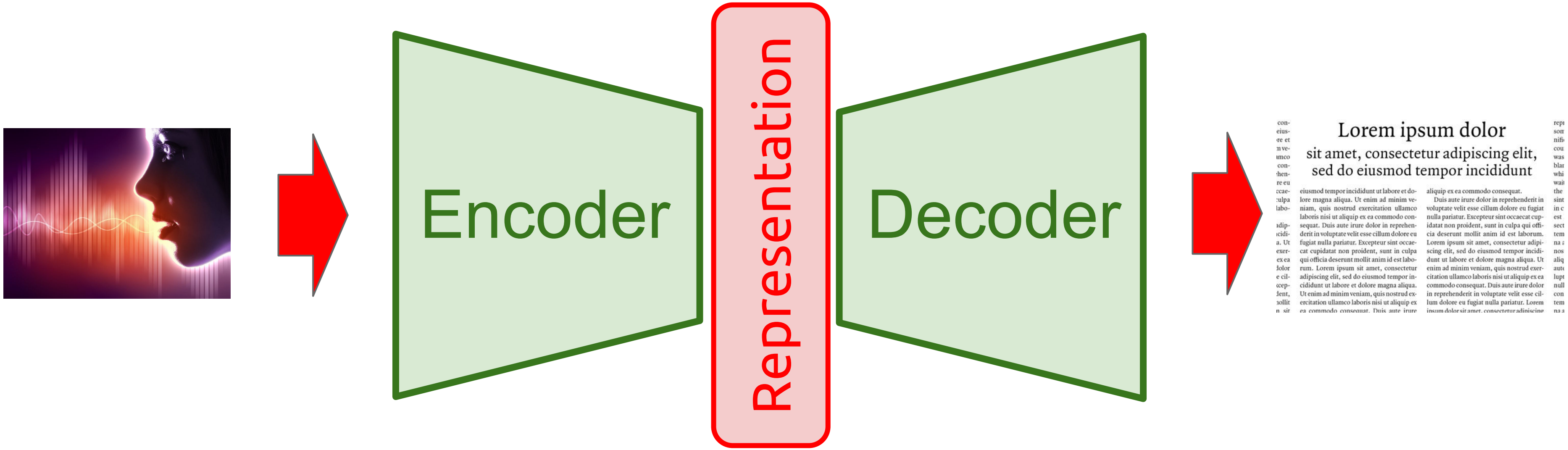
Neural Machine Translation (NMT)



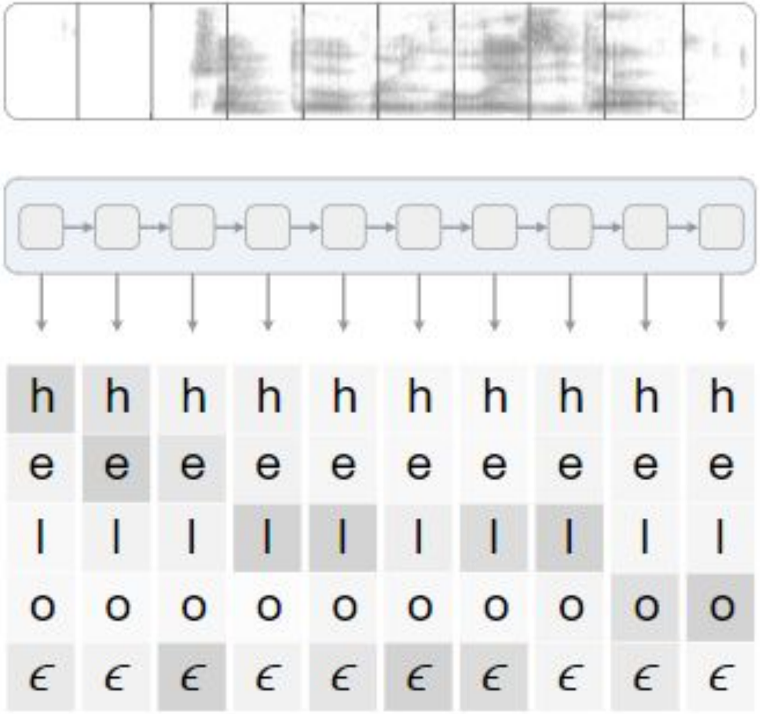
Speech Encoding



Chan, William, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition." ICASSP 2016.


Encoder
Representation
Decoder
Lorem ipsum dolor
sit amet, consectetur adipiscing elit,
sed do eiusmod tempor incididunt

Automatic Speech Recognition (ASR)

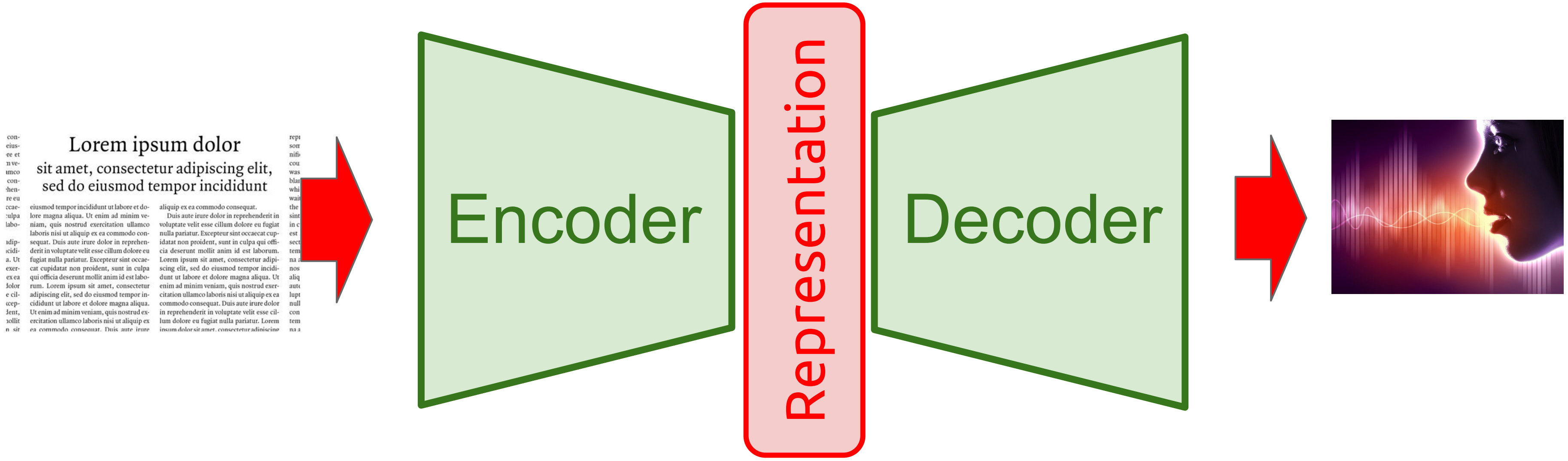


We start with an input sequence, like a spectrogram of audio.

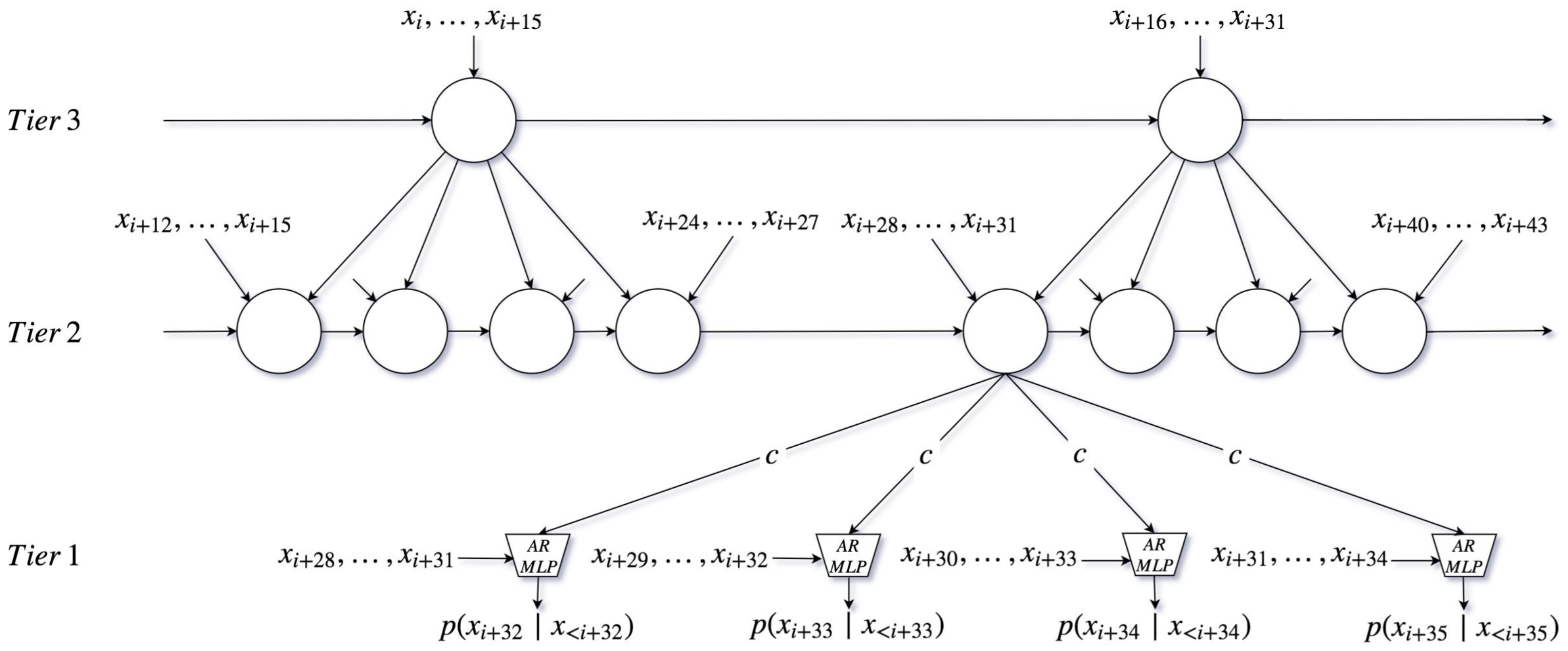
The input is fed into an RNN, for example.

The network gives $p_t(a \mid X)$, a distribution over the outputs $\{h, e, l, o, \epsilon\}$ for each input step.

Slide: Hannun, Awni. "Sequence Modeling with CTC." *Distill* 2.11 (2017): e8.


Lorem ipsum dolor
sit amet, consectetur adipiscing elit,
sed do eiusmod tempor incididunt
Encoder
Representation
Decoder

Speech Synthesis



#SampleRNN Mehri, Soroush, Kundan Kumar, Ishaan Gulrajani, Rithesh Kumar, Shubham Jain, Jose Sotelo, Aaron Courville, and Yoshua Bengio. ["SampleRNN: An unconditional end-to-end neural audio generation model."]() ICLR 2017.

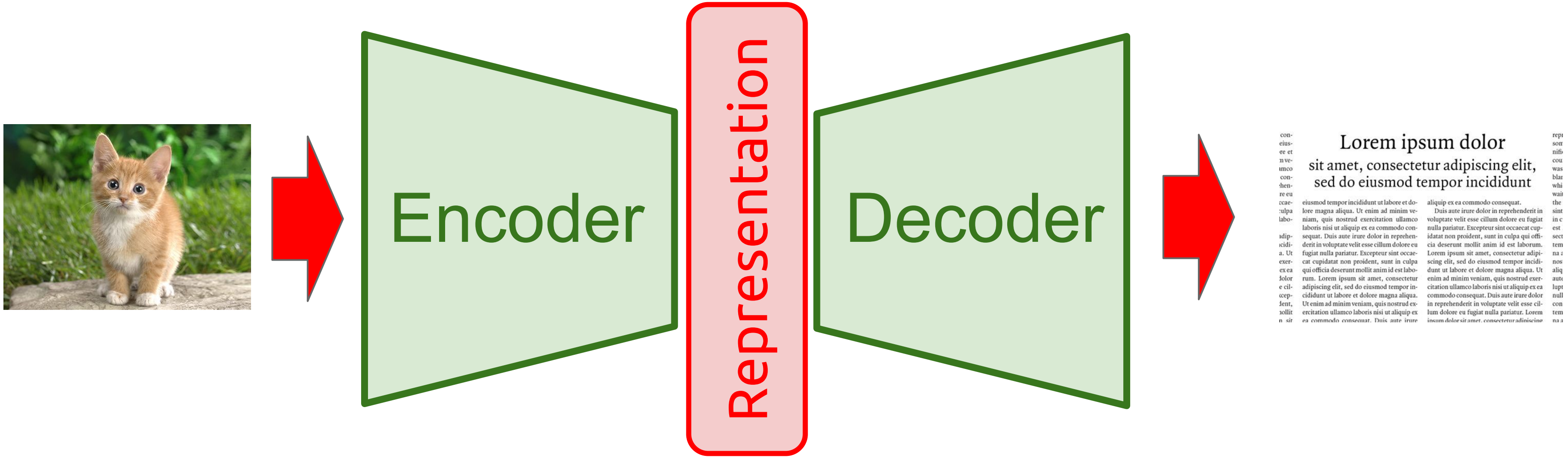
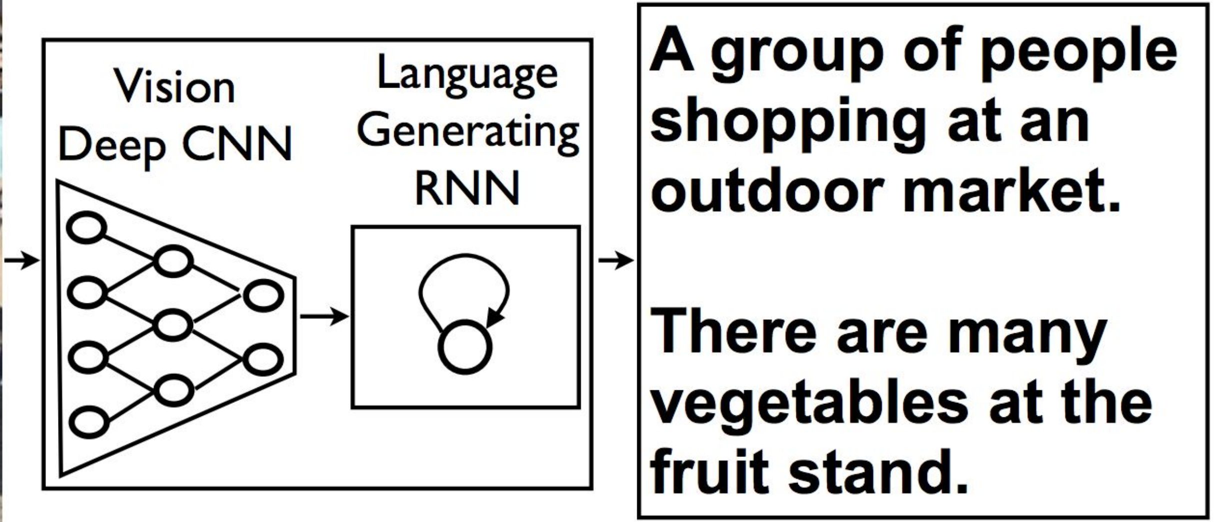
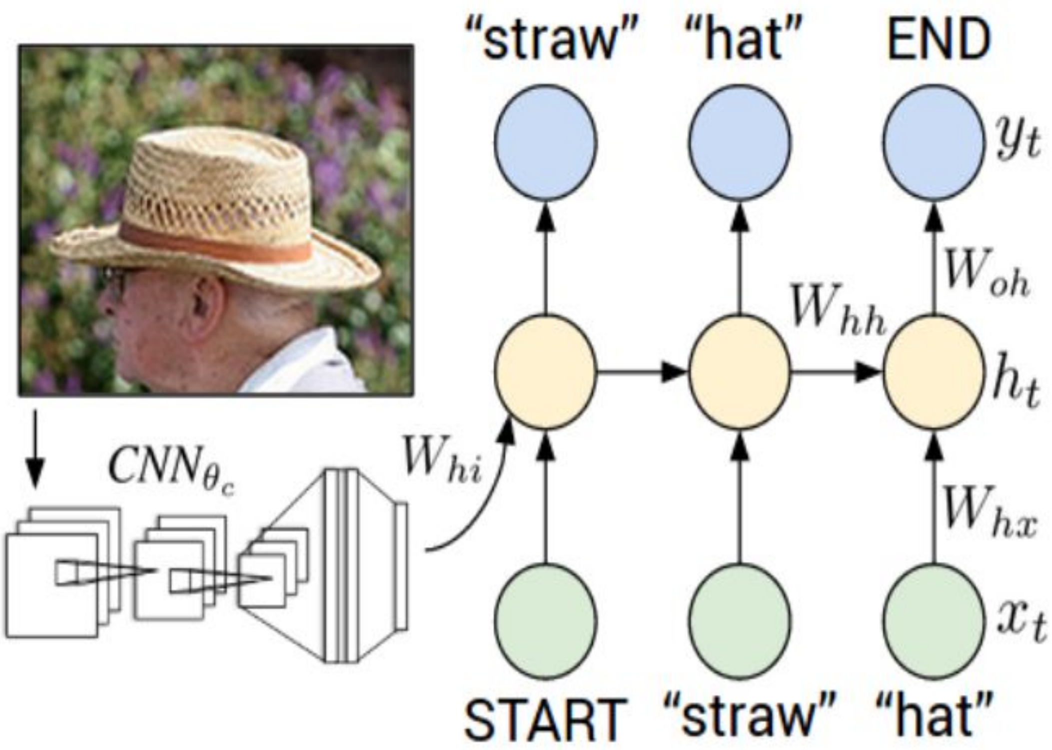

Encoder
Representation
Decoder
Lorem ipsum dolor
sit amet, consectetur adipiscing elit,
sed do eiusmod tempor incididunt

Image Captioning with RNN

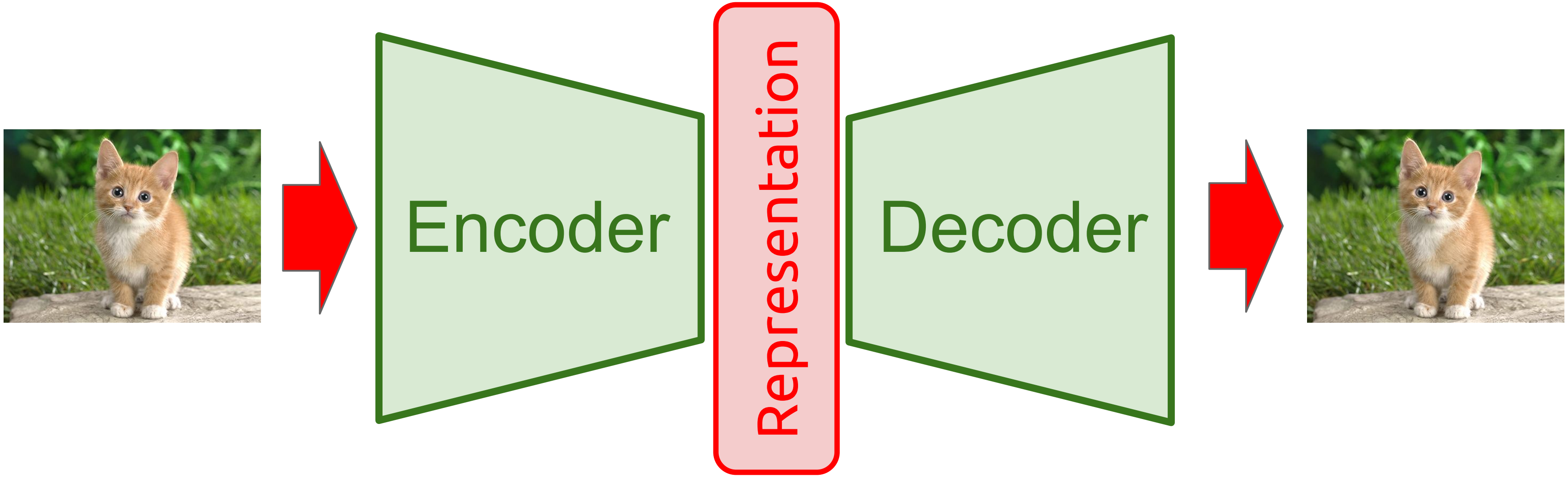


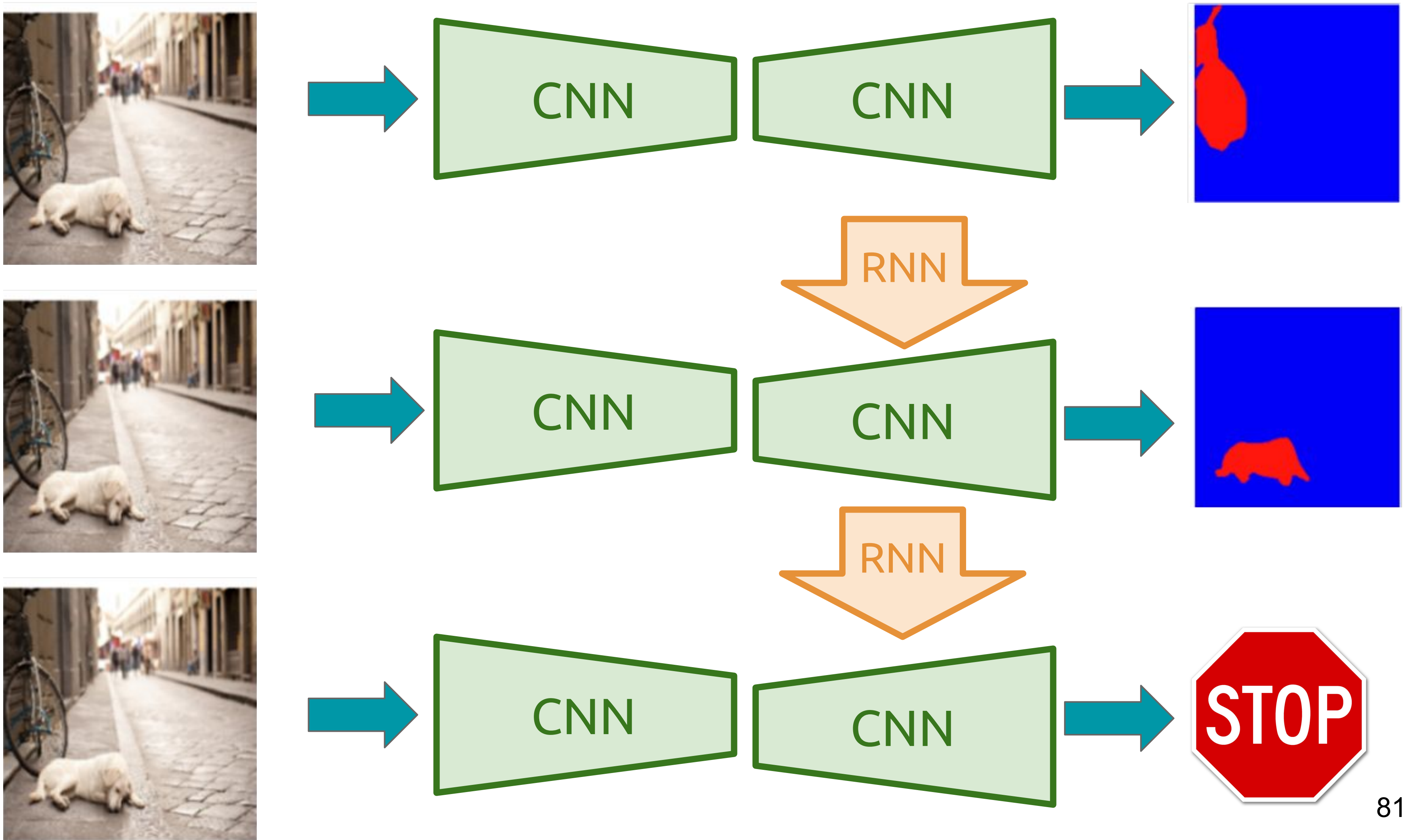
#ShowAndTell Vinyals, Oriol, Alexander Toshev, Samy Bengio, and Dumitru Erhan. ["Show and tell: A neural image caption generator."](#) CVPR 2015.

Image Captioning with RNN

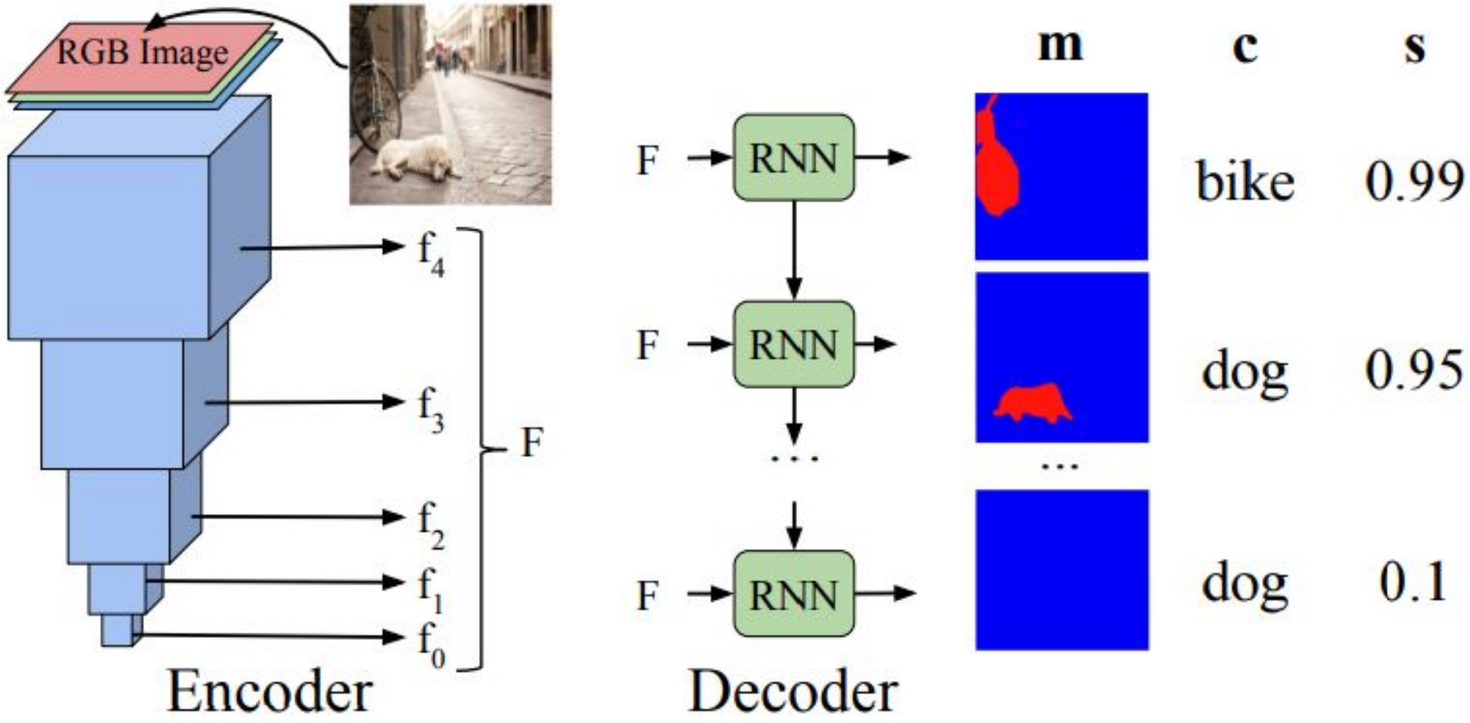


#DeepImageSent Karpathy, Andrej, and Li Fei-Fei. "Deep visual-semantic alignments for generating image descriptions." CVPR 2015 (Slides by Marc Bolaños)


Encoder
Representation
Decoder

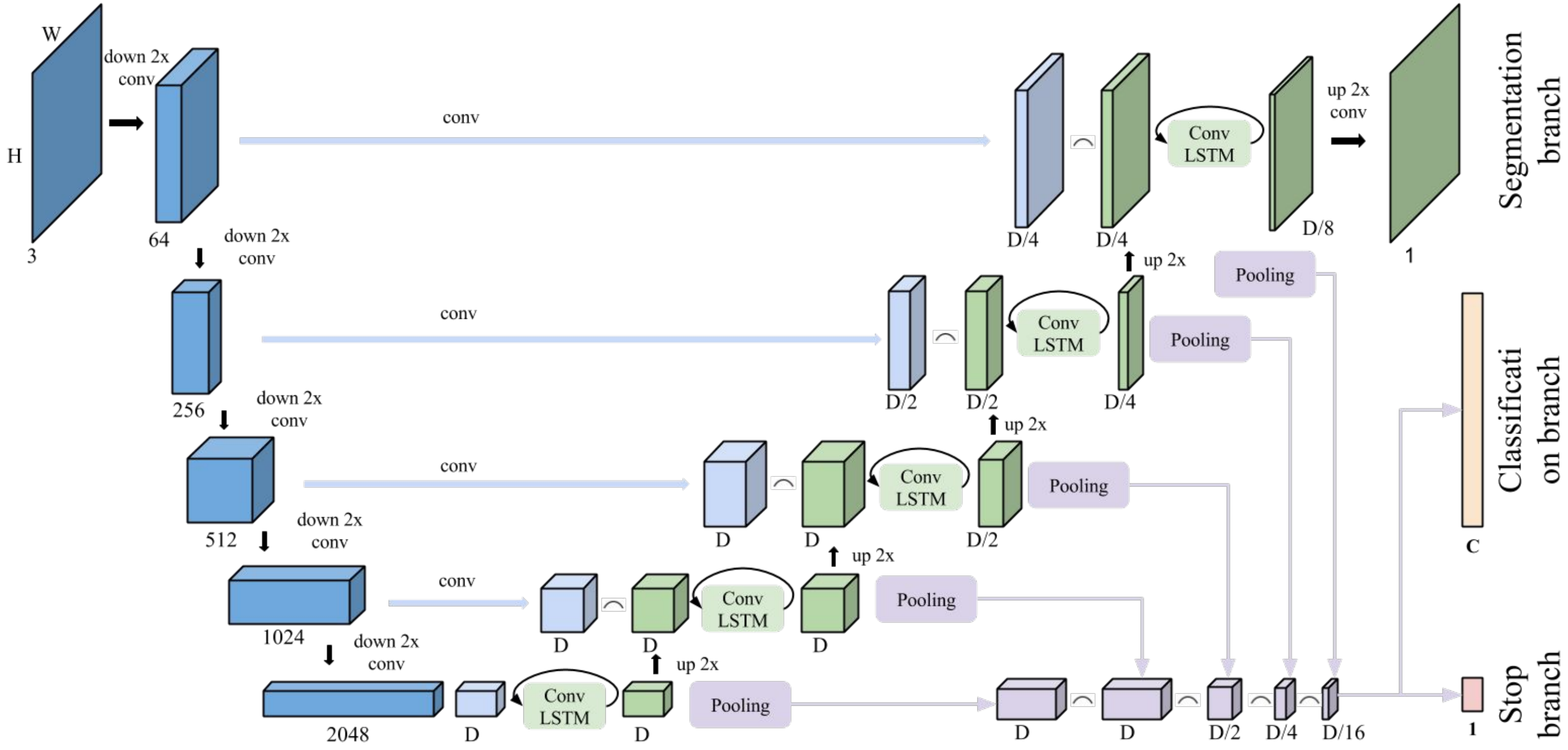

CNN
CNN
RNN
CNN
CNN
RNN
CNN
CNN
STOP

Recurrent Instance Segmentation



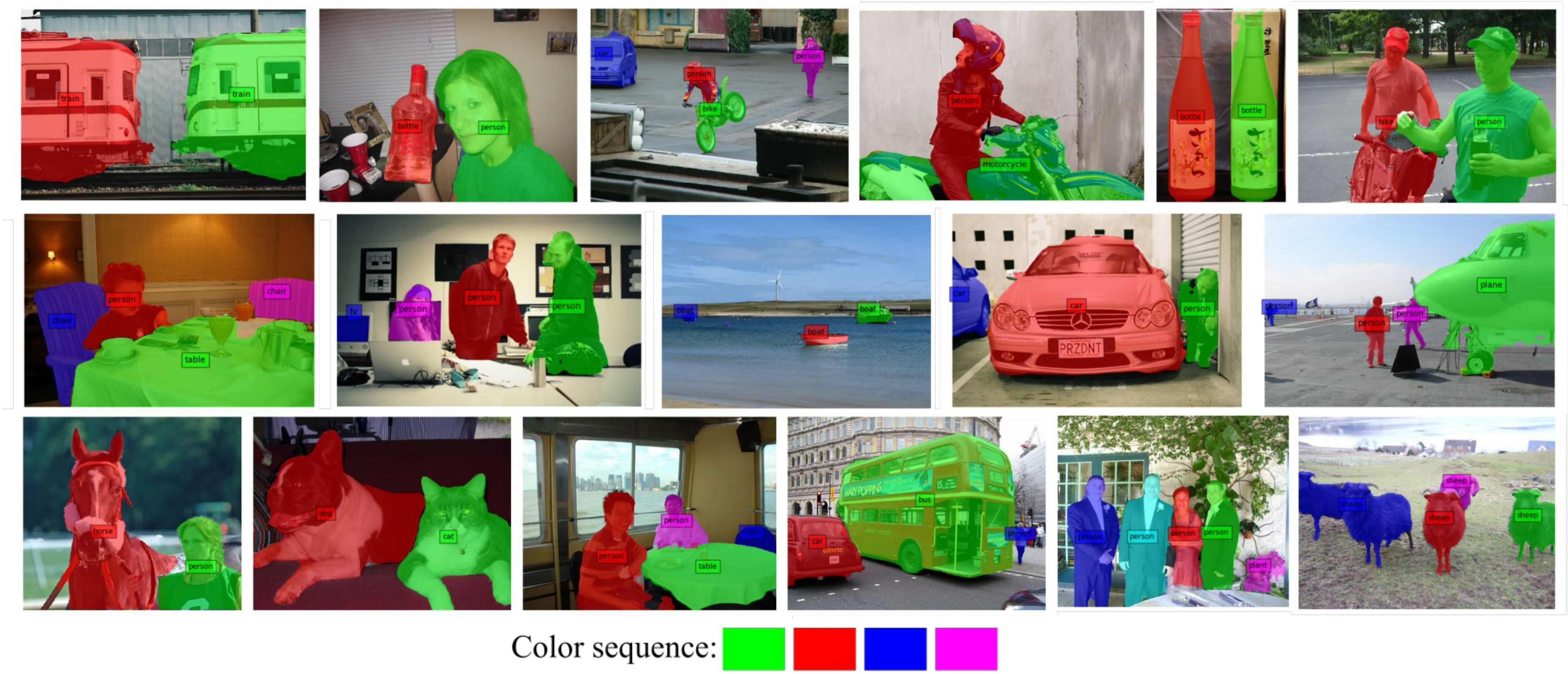
#RSIS Salvador, Amaia, Miriam Bellver, Victor Campos, Manel Baradad, Ferran Marques, Jordi Torres, Xavier Giro-i-Nieto. ["Recurrent neural networks for semantic instance segmentation."]() arXiv preprint arXiv:1712.00617 (2017).

Recurrent Instance Segmentation



#RSIS Salvador, Amaia, Miriam Bellver, Victor Campos, Manel Baradad, Ferran Marques, Jordi Torres, Xavier Giro-i-Nieto. ["Recurrent neural networks for semantic instance segmentation."]() arXiv preprint arXiv:1712.00617 (2017).

Recurrent Instance Segmentation



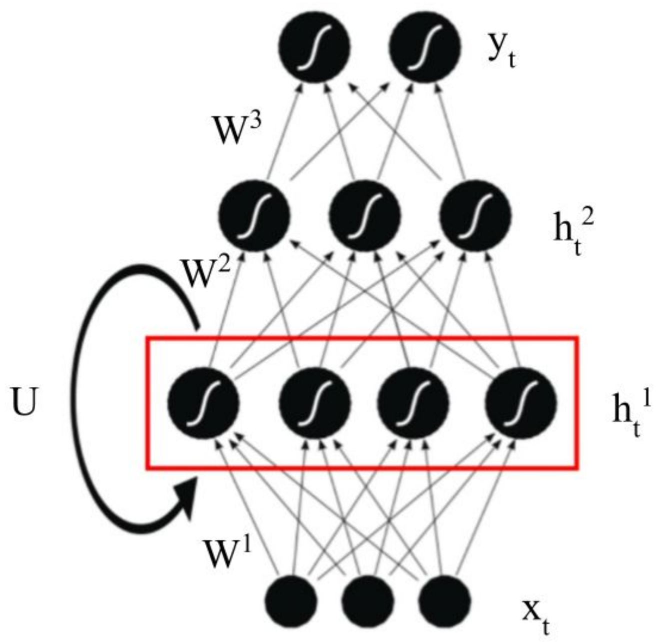
#RSIS Salvador, Amaia, Miriam Bellver, Victor Campos, Manel Baradad, Ferran Marques, Jordi Torres, Xavier Giro-i-Nieto. ["Recurrent neural networks for semantic instance segmentation."]() arXiv preprint arXiv:1712.00617 (2017).

Outline

1. Motivation
2. Recurrent Neural Layer
3. Forward & Backward Passes Through Time
4. Gated Recurrent Neurons
    a. LSTM
    b. GRU
5. Advanced Architectures
6. Encoder-Decoder Arquitectures

Self-study (homework)

Consider a the neural network depicted in the figure, that represents the processing of a data sequence at timestep t. The layer in the box is a recurrent one, with a set of weights U.



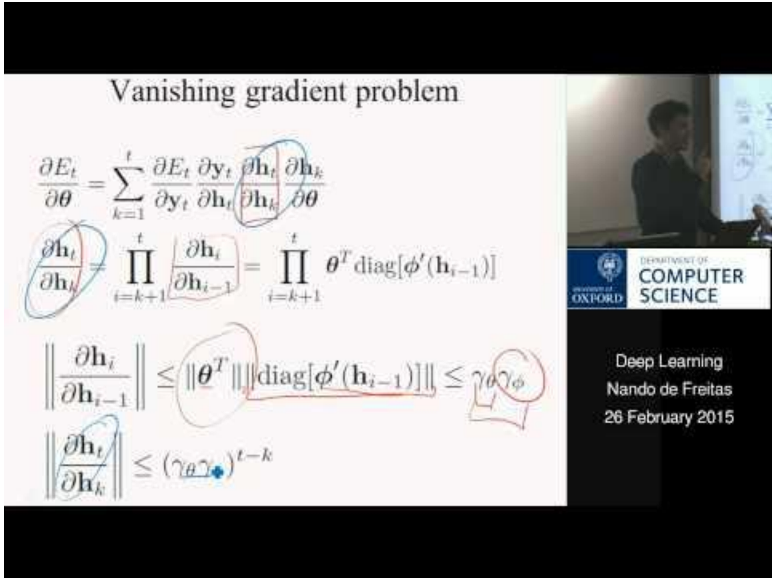
a) Formulate how is the output of the recurrent layer, h_t^1, computed considering a tanh(·) activation.

b) How many parameters does this architecture contain ? Together with the depicted weight matrices W^i, also consider the biases b^i. Develop the calculations you made to reach the provided results.

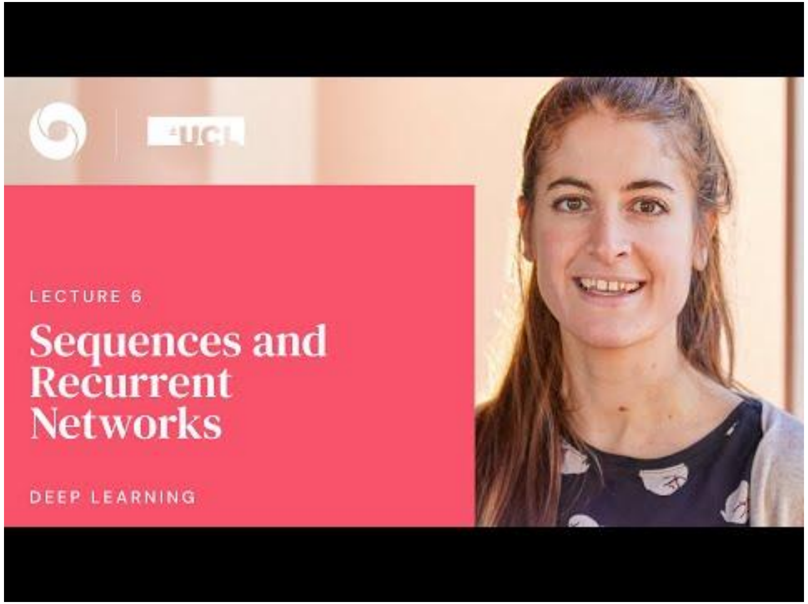
c) Draw the unfolded representation of this architecture for timestep t-1, t and t+1, labeling the edges with the appropriate weight matrices.

Learn more

- Chris Olah, Shan Carter, [“Attention and Augmented Recurrent Neural Networks”]. distill.pub 2016.
- Jordi Pons (UPF): [Slides on Recurrent Neural Networks] [[tweet]]
- Ian Goodfellow, Yoshua Bengio, Aaron Courville, [“Deep Learning (Chapter 10)”]. MIT Press 2016.
- Alex Graves, [“Supervised Sequence Labelling with Recurrent Neural Networks”]

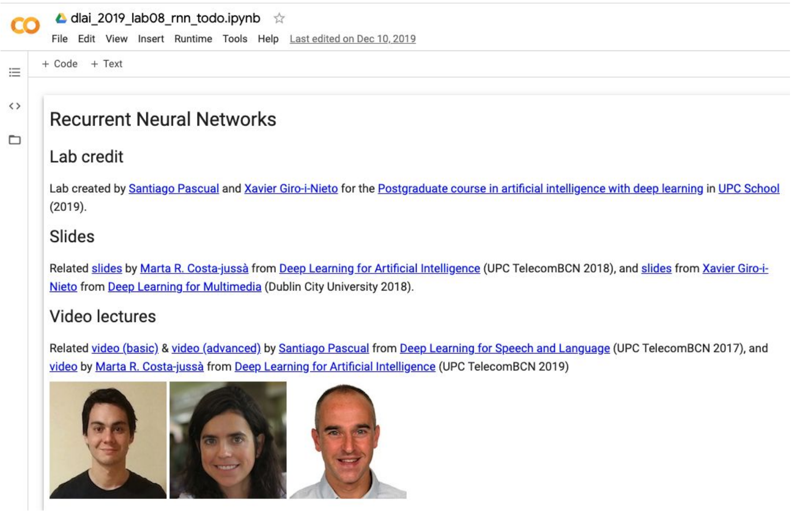


Nando de Freitas, [“Recurrent Neural Nets and LSTMs”] (University of Oxford 2015).



Marta Garnelo, [Deepmind x UCL 2020] [[slides]]

(extra) PyTorch Lab on Google Colab



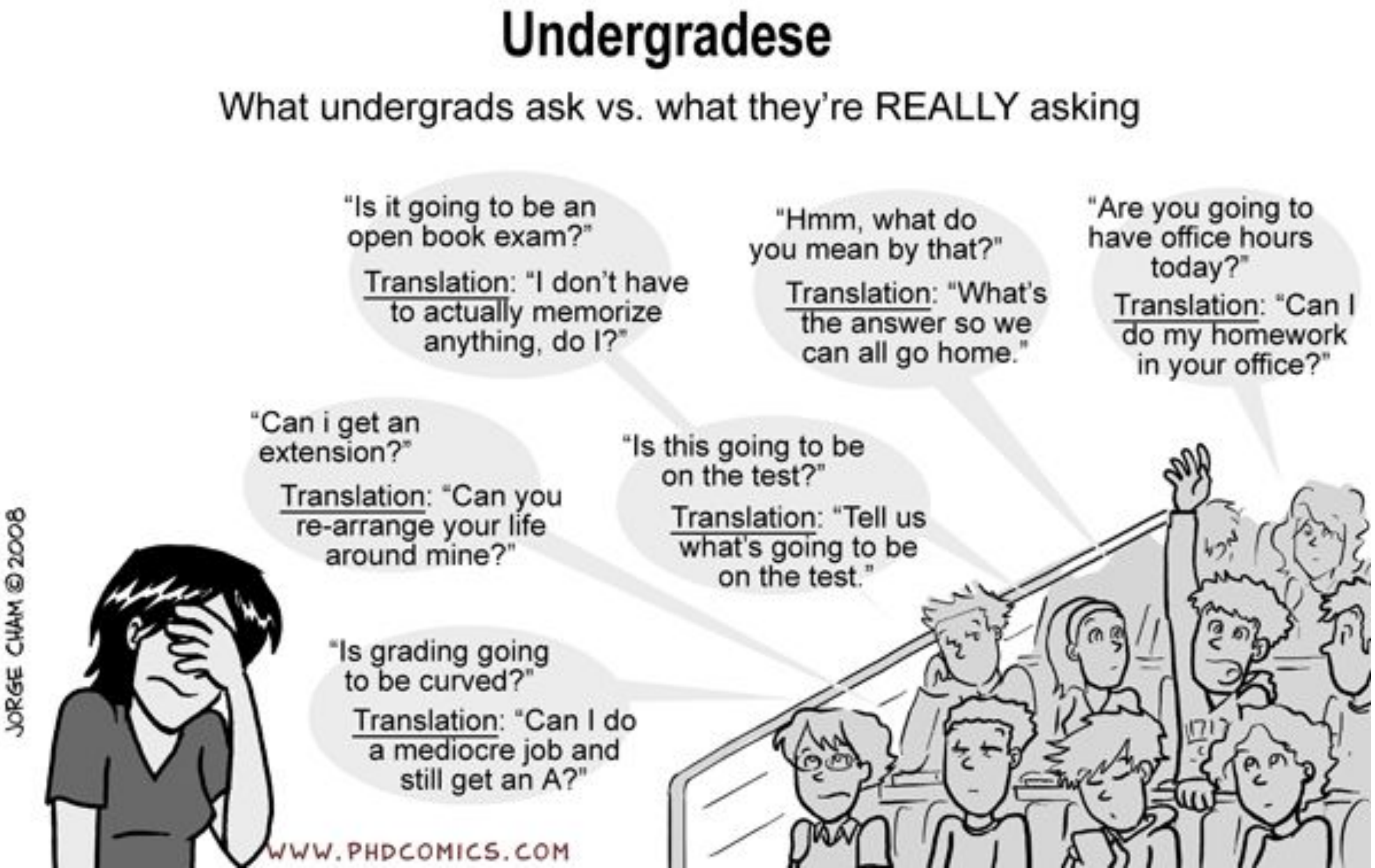
Santiago Pascual de la Puente
santi.pascual@upc.edu

PhD 2019
Universitat Politecnica de Catalunya
Technical University of Catalonia

DL resources from UPC Telecos:

- Lectures (with Slides & Videos)
- Labs

Questions ?


Undergradese
What undergrads ask vs. what they're REALLY asking
"Is it going to be an open book exam?"
Translation: "I don't have to actually memorize anything, do I?"
"Hmm, what do you mean by that?"
Translation: "What's the answer so we can all go home."
"Are you going to have office hours today?"
Translation: "Can I do my homework in your office?"
"Can i get an extension?"
Translation: "Can you re-arrange your life around mine?"
"Is this going to be on the test?"
Translation: "Tell us what's going to be on the test."
"Is grading going to be curved?"
Translation: "Can I do a mediocre job and still get an A?"
JORGE CHAM © 2008
WWW.PHDCOMICS.COM