Neural Networks for sequential data

29 January, 2020

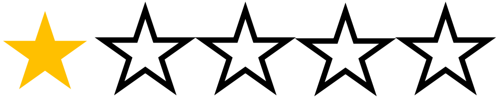
Alexey Zaytsev, head of Laboratory LARSS, PhD

ISP course on sequential data processing

Skoltech
Skolkovo Institute of Science and Technology

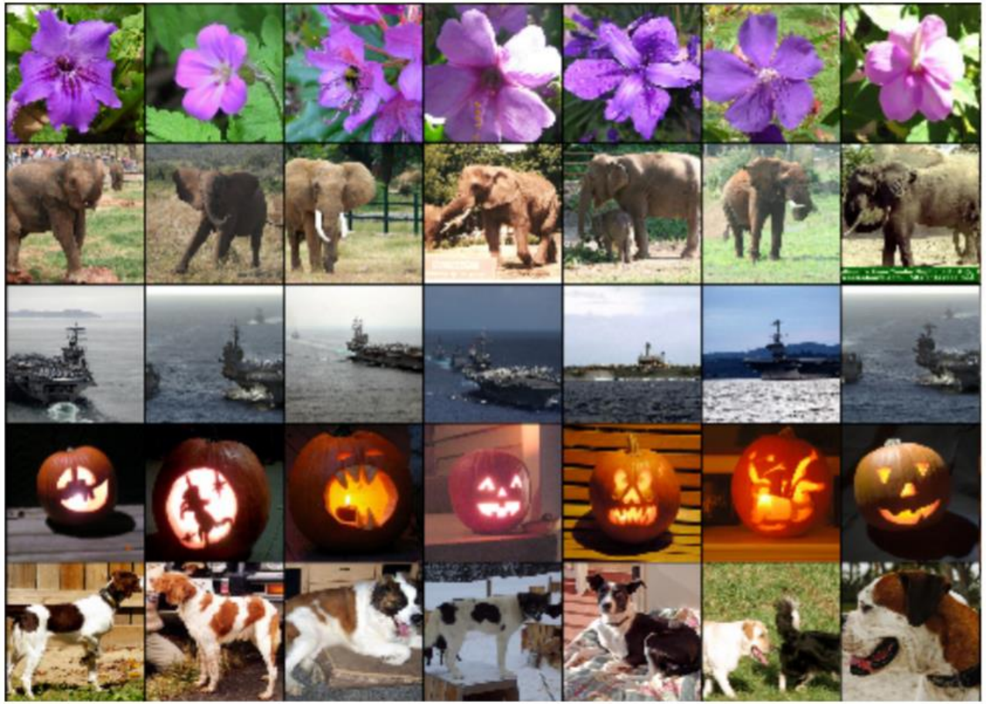
- → **Why bother about Neural Networks for sequential data?**
- → How to do Neural Networks for sequential data?

Complex sequence data examples

Speech recognition		→	“The quick brown fox jumped over the lazy dog.”
Sentiment classification	“There is nothing to like in this movie.”	→	
DNA sequence analysis	AGCCCCTGTGAGGAACTAG	→	AGCCCCTGTGAGGAACTAG
Machine translation	Voulez-vous chanter avec moi?	→	Do you want to sing with me?
Video activity recognition		→	Running

Adopted from Andrew Ng

Why bother with deep learning: Deep learning is better, then a human



Human error percentage is

5.1%

DL error percentage is

3.57%

https://arxiv.org/pdf/1705.02498.pdf
https://qz.com/1034972/the-data-that-changed-the-direction-of-ai-research-and-possibly-the-world/
http://www.image-net.org/papers/imagenet_cvpr09.pdf

Classic approach to Machine Learning

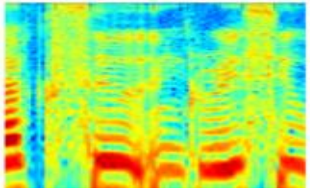
Data → Features → Learning

Image

Features

Recognition

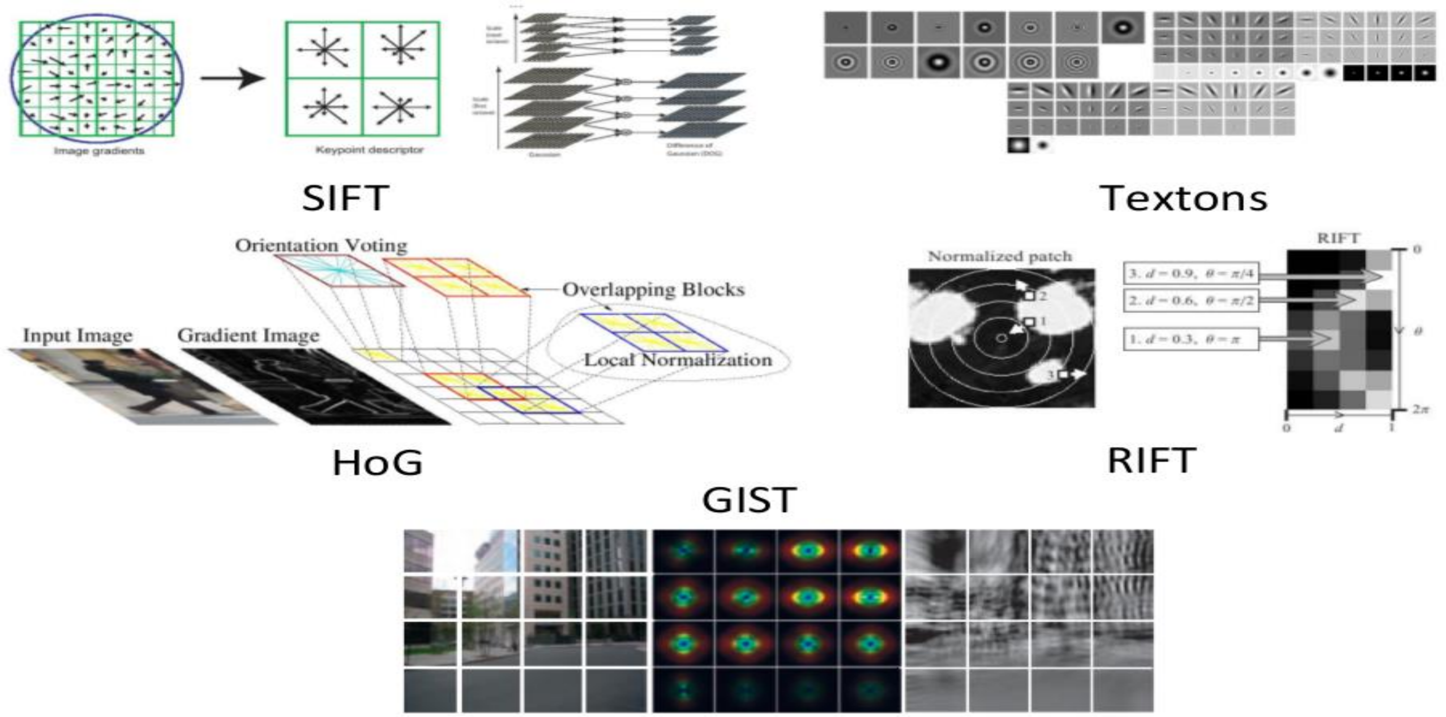
Speech



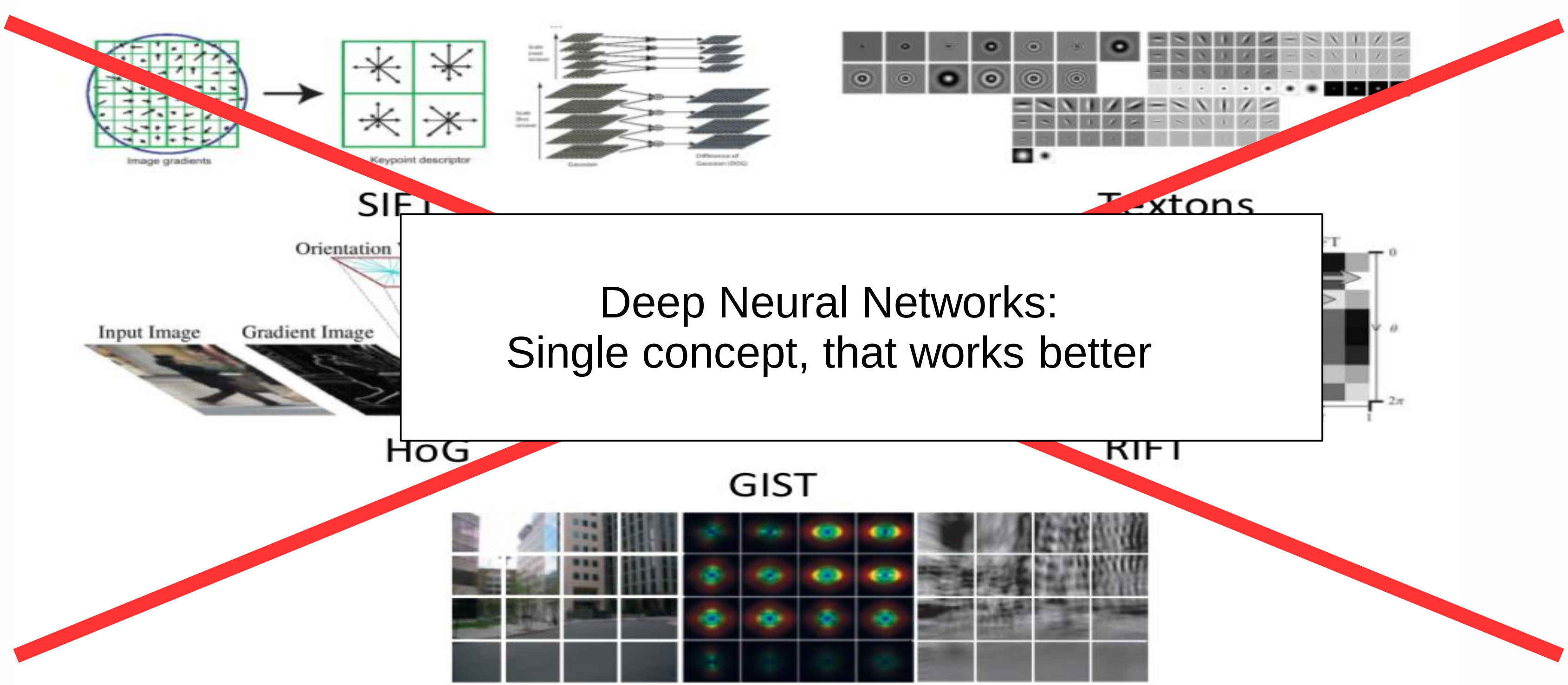
Features

Recognition

An art of feature construction



No art now, just engineering



Skoltech
Skolkovo Institute of Science and Technology

Why and how Deep learning works

- Availability of Graphical processing units (GPU)

- Large scale “big” data

- Open-source libraries

- Complex structured data

Complex sequence data examples

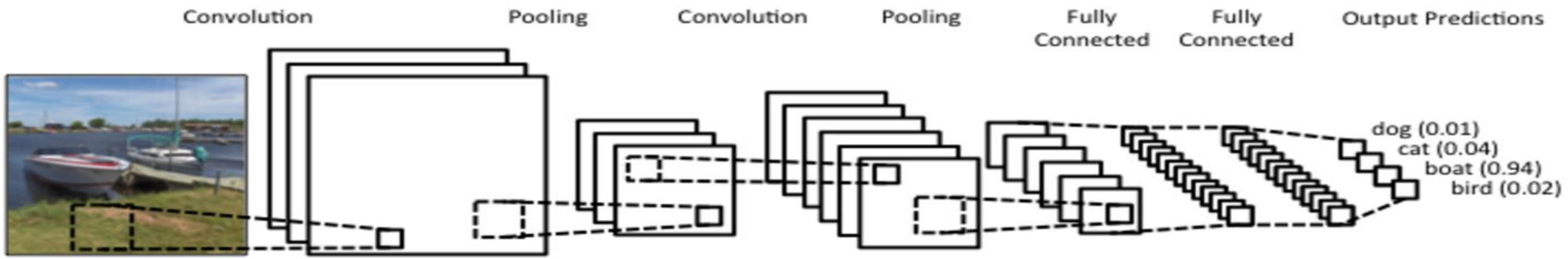
		Large data set	Structured data
Speech recognition		+	+
Sentiment classification	“There is nothing to like in this movie.”	+	+
DNA sequence analysis	AGCCCCTGTGAGGAACTAG	+	+
Machine translation	Voulez-vous chanter avec moi?	+	+
Video activity recognition		+	+

- → Why bother about Neural Networks for sequential data?
- → **How to do Neural Networks for sequential data?**

Skoltech
Skolkovo Institute of Science and Technology

Convolutional Neural Networks

- Mathematical definition: combination of simple transformations
- There can be a lot of layers
- Convolutional layers help

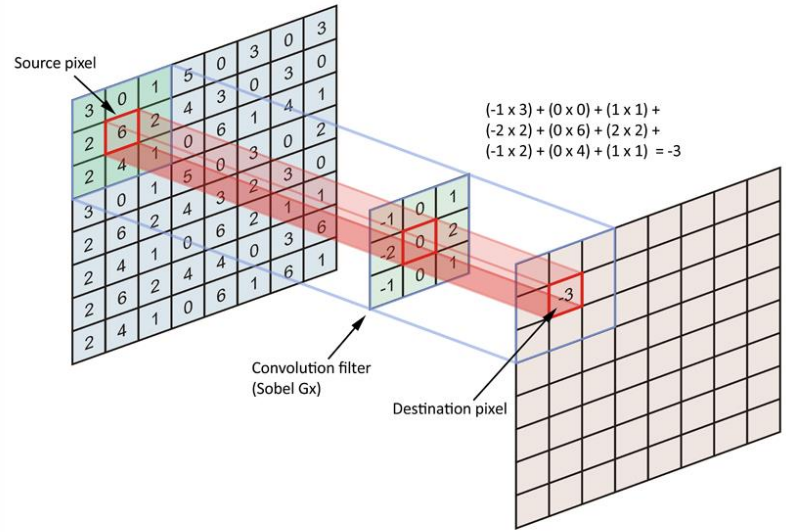


https://playground.tensorflow.org/

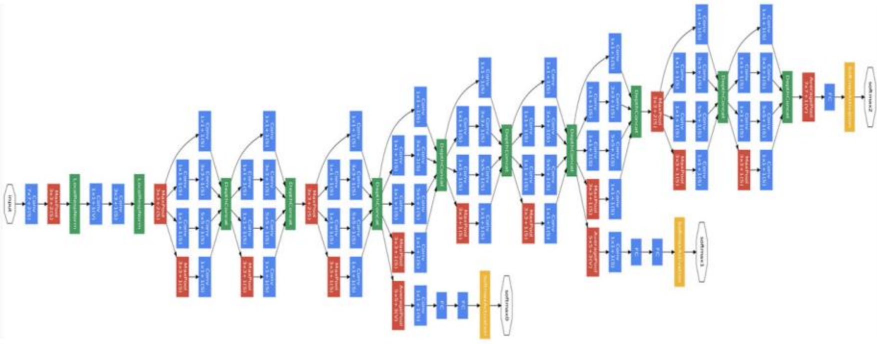
Skoltech
Skolkovo Institute of Science and Technology

Convolutional Neural Networks

- Mathematical definition: combination of simple transformations
- There can be a lot of layers
- Convolutional layers help

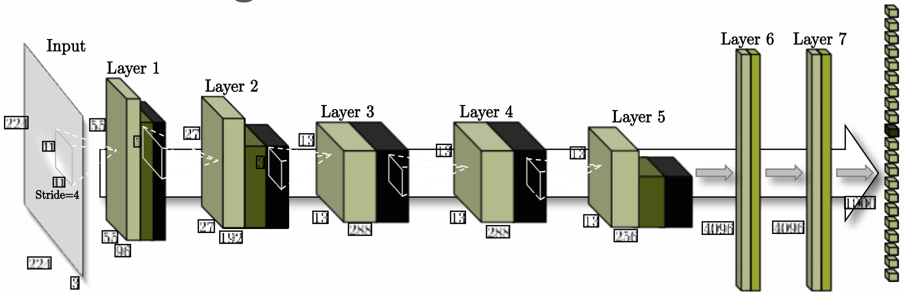


Single convolutional layer

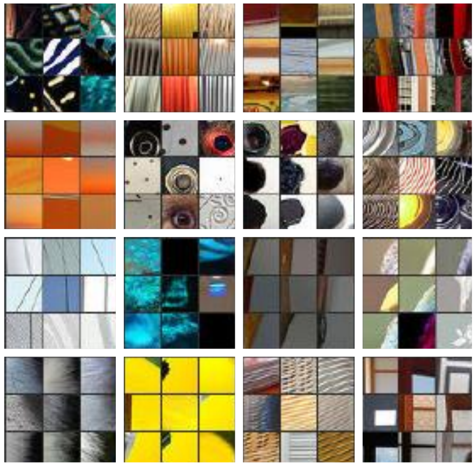


GoogLeNet architecture

Flow of data through Convolutional Neural Network



Layer 1

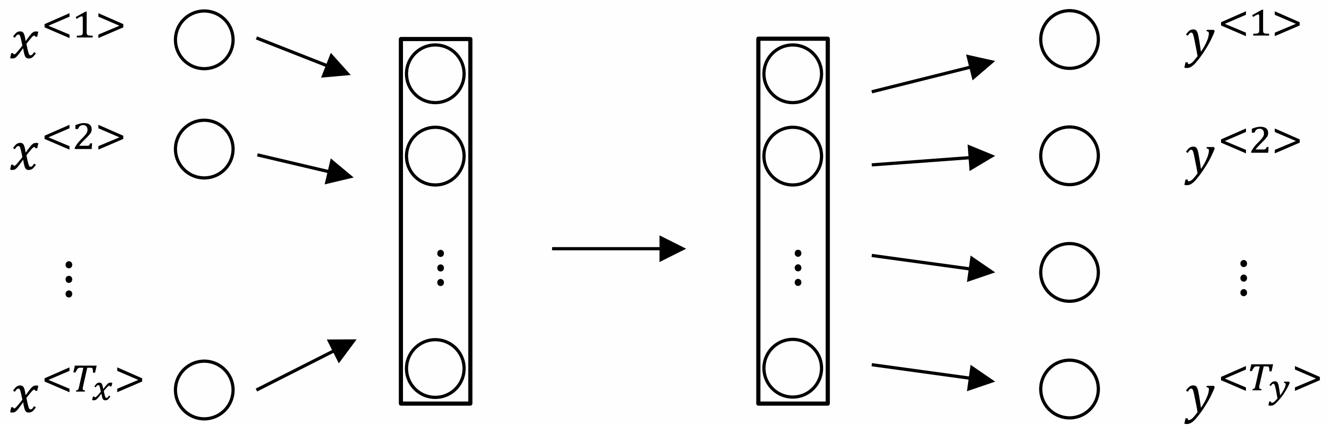


Layer 2

Layer 5

[Zeiler Fergus 14]

Why not a standard fully-connected or convolutional network?



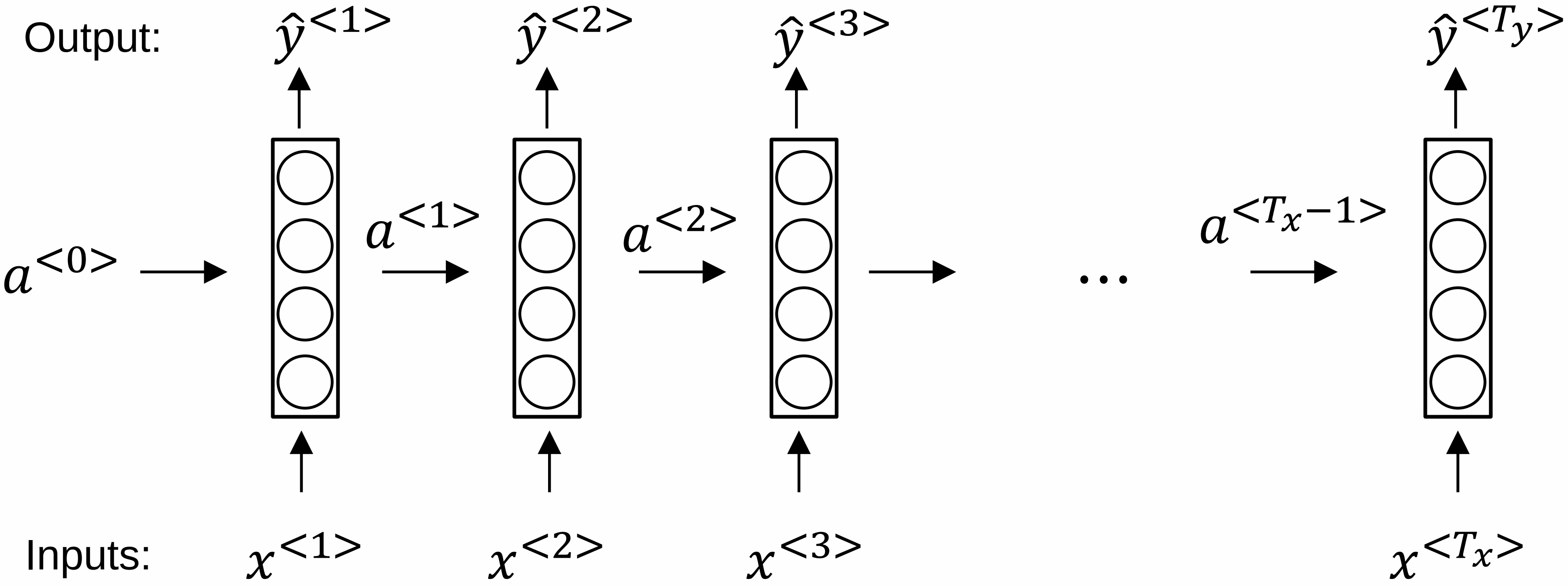
Problems:

- Inputs, outputs can be different lengths in different examples
- Doesn’t share features learned across different positions of text

Adopted from Andrew Ng

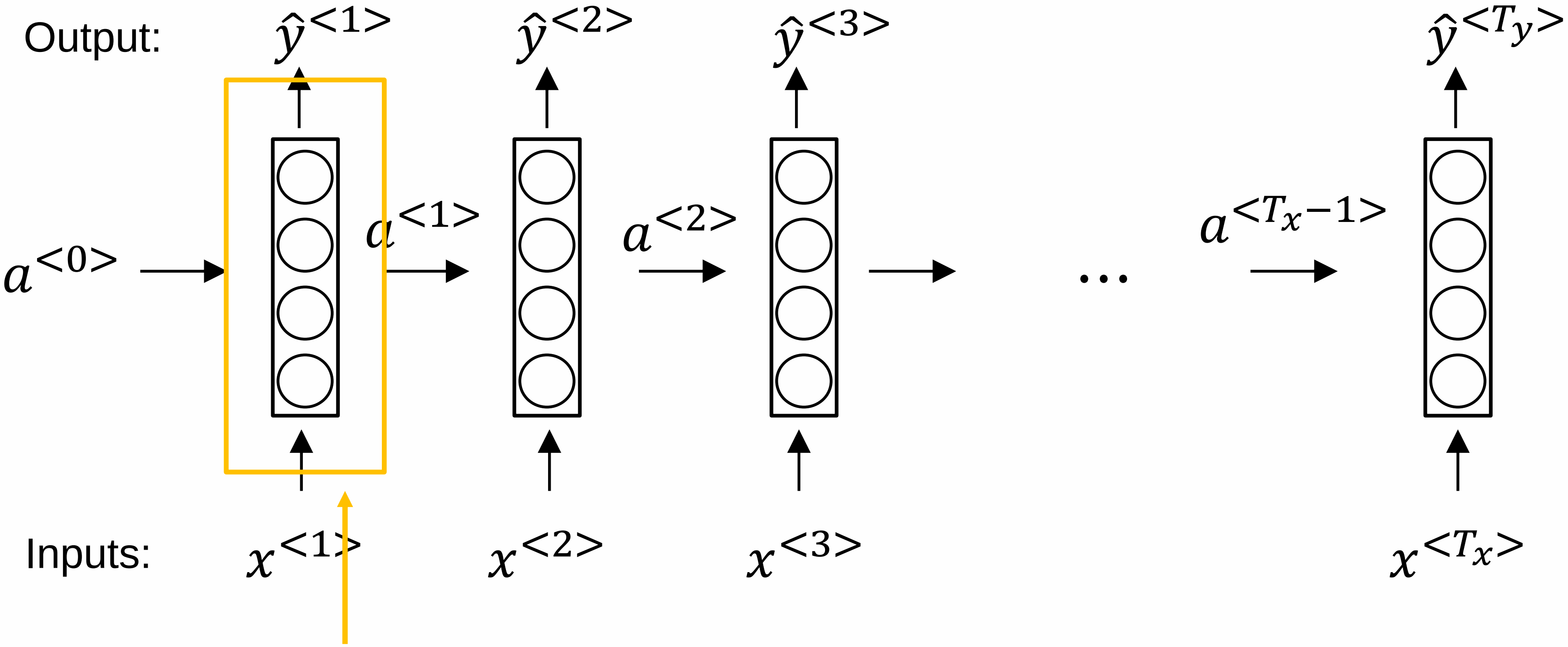
Skoltech
Skolkovo Institute of Science and Technology

Forward propagation through Recurrent Neural Network



Adopted from Andrew Ng

Forward propagation through Recurrent Neural Network

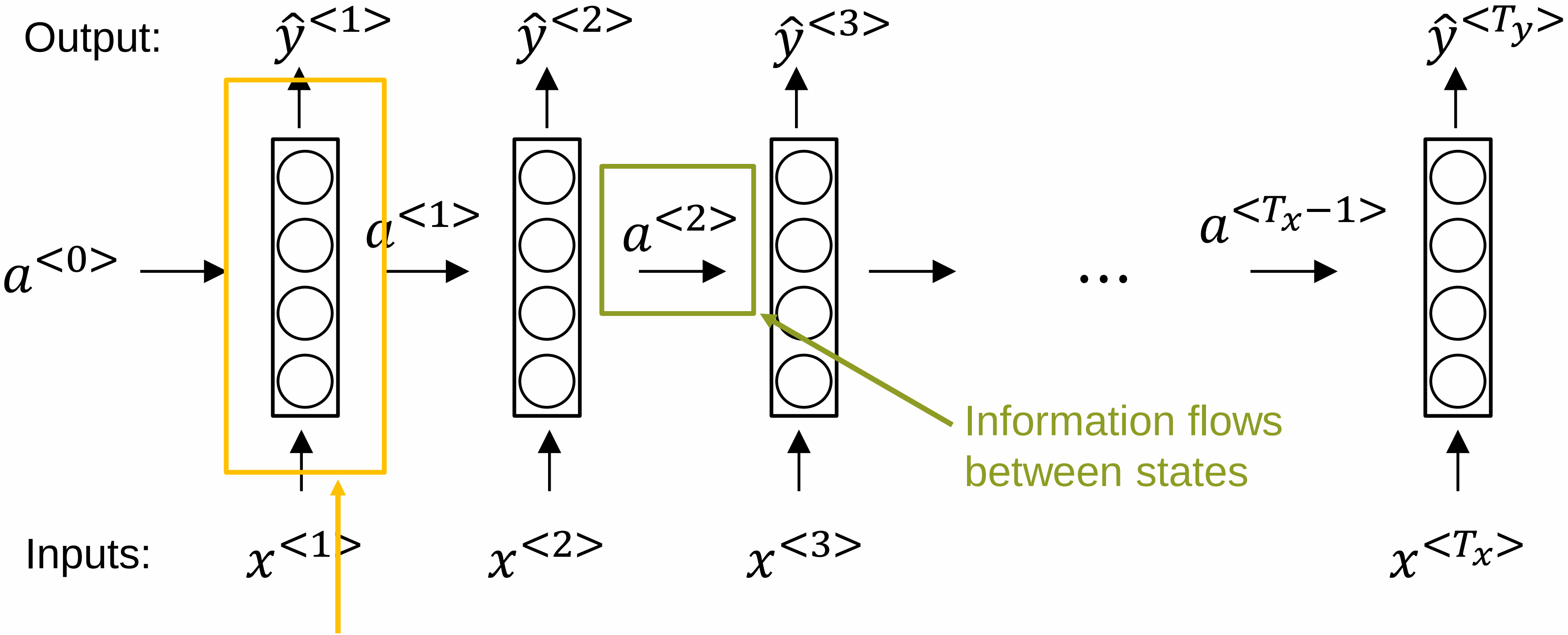


Processing unit is the same for all time moments.
Units has parameters we want to learn!

Adopted from Andrew Ng

Skoltech
Skolkovo Institute of Science and Technology

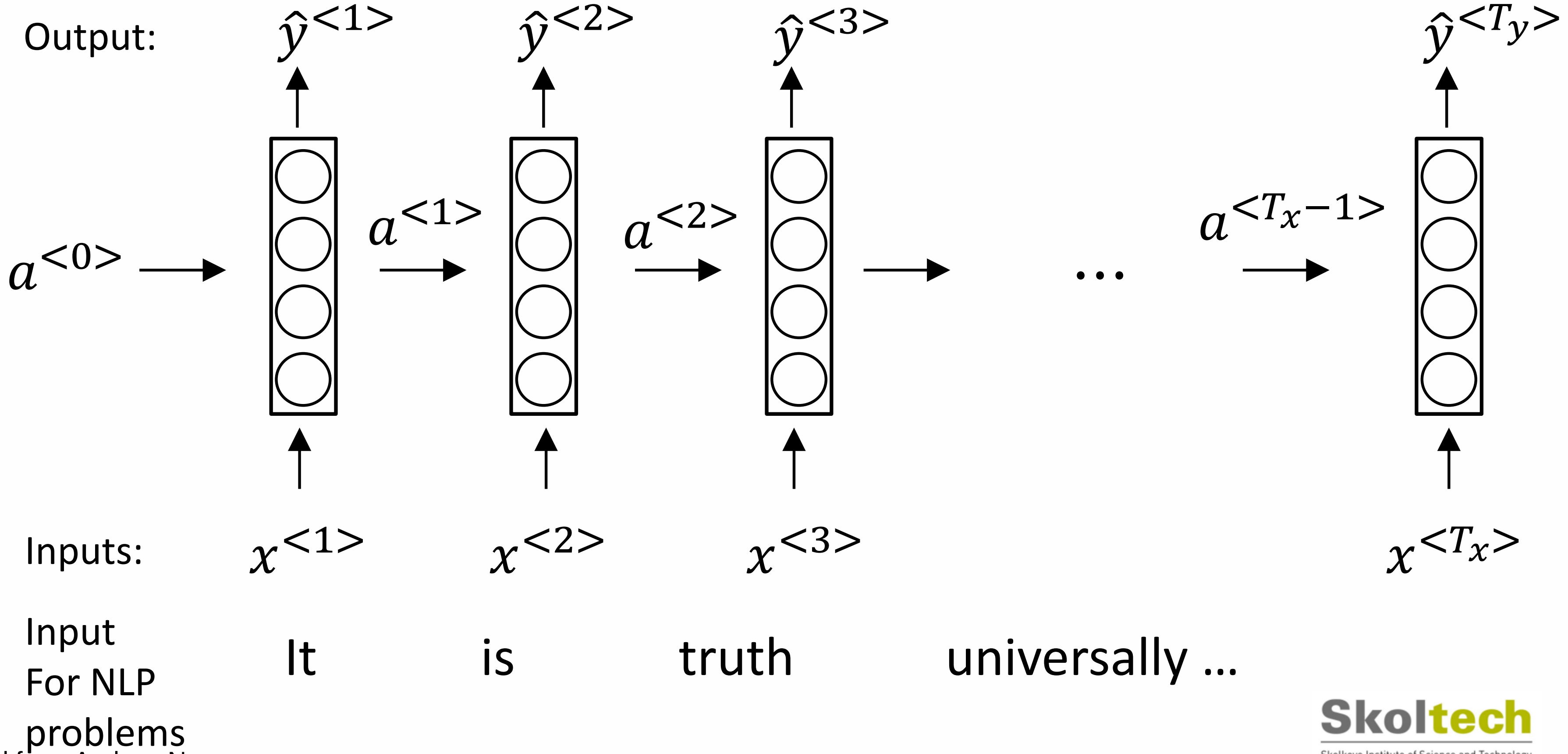
Forward propagation through Recurrent Neural Network



Processing unit is the same for all time moments.
Units has parameters we want to learn!

Adopted from Andrew Ng

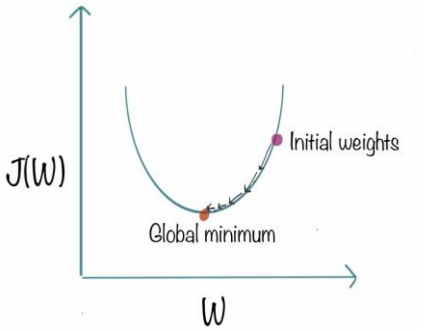
Forward propagation through Recurrent Neural Network

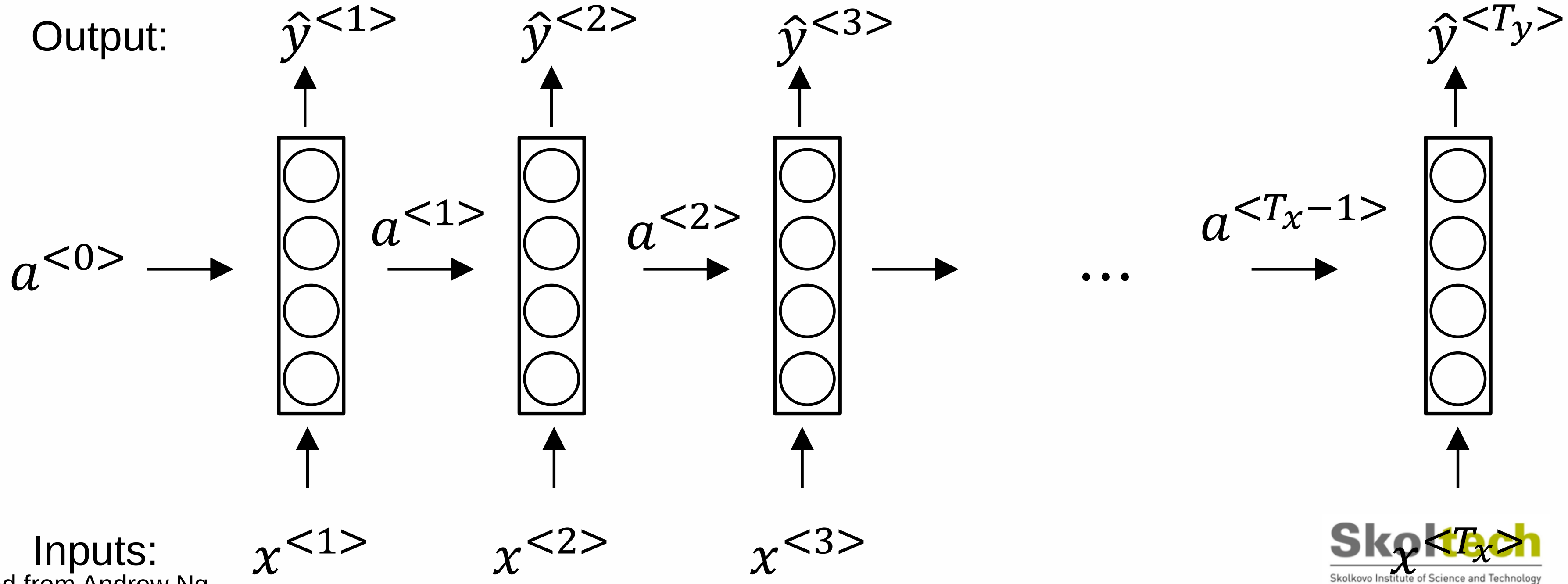


Adopted from Andrew Ng

Backward propagation through Recurrent Neural Network

- Our loss function is $\mathcal{L}^{<t>}(\hat{y}^{<t>}, y^{<t>})$
- We want to estimate parameters, so we need derivatives!

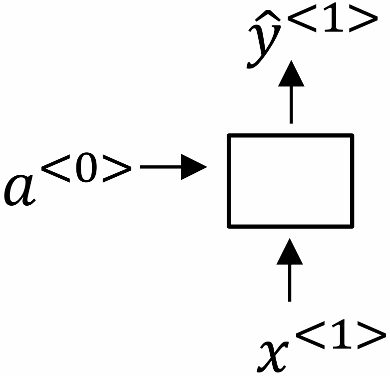




Adopted from Andrew Ng

Zoo of RNN (Recurrent Neural Network) models

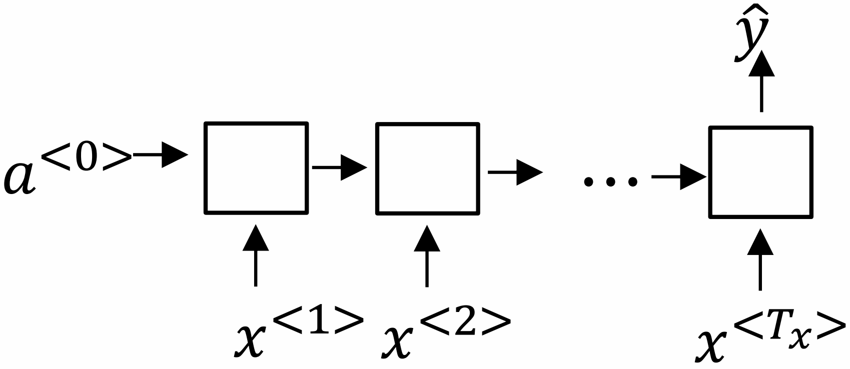
One to one



We have a separate input & output each time

- Image classification from cameras

Many to one



We have a single output for a sequence

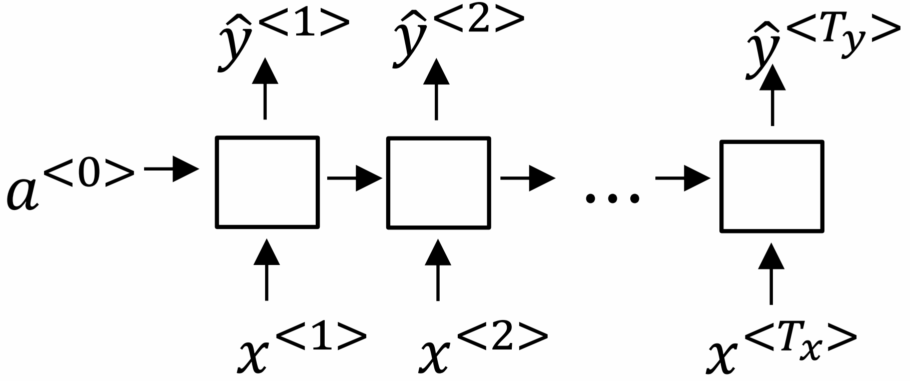
- Sentiment of a sentence: good or bad review?

Skoltech
Skolkovo Institute of Science and Technology

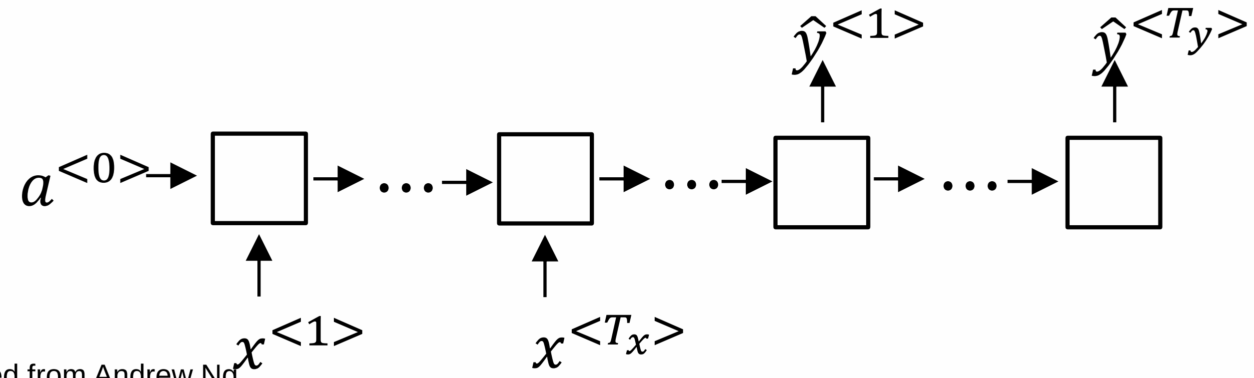
Adopted from Andrew Ng

Zoo of RNN (Recurrent Neural Network) models

Many to many



- Time series prediction
- Anomaly detection



- Translation

Adopted from Andrew Ng

Forward propagation through Recurrent Neural Network

Initial sequence:

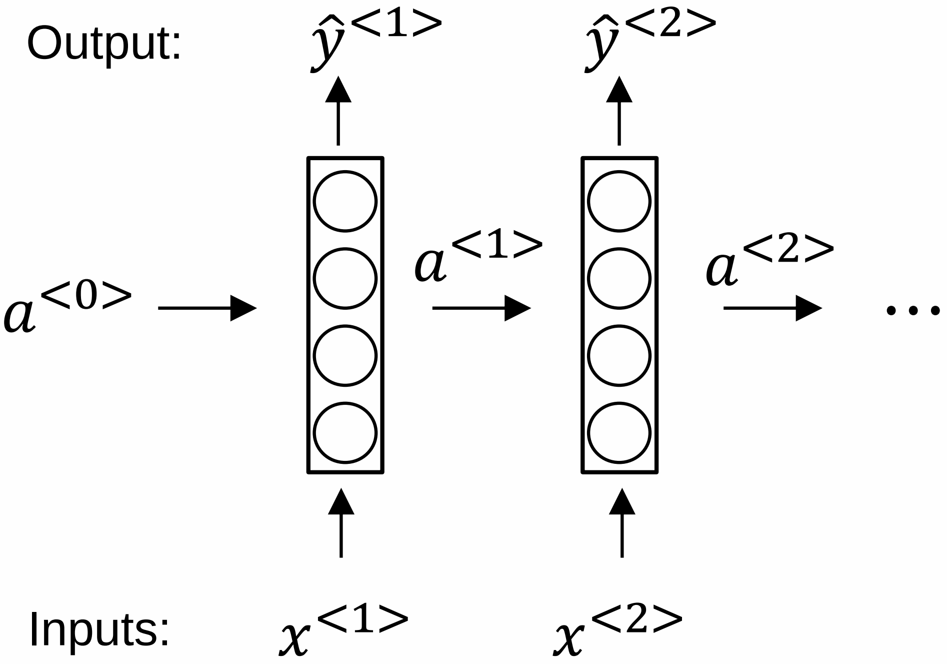
$$x^{<1:n>} = x^{<1>}, x^{<2>}, \dots, x^{<T>}, x_i \in \mathbb{R}^{d_{in}}$$

For each input $x_{1:i}$ we get an output y_i:

$$y^{<i>} = RNN(x^{<1:i>}), y^{<i>} \in \mathbb{R}^{d_{out}}$$

For the whole sequence $x^{<1:n>}$:

$$y^{<1:n>} = RNN^*(x^{<1:n>}), y^{<i>} \in \mathbb{R}^{d_{out}}$$



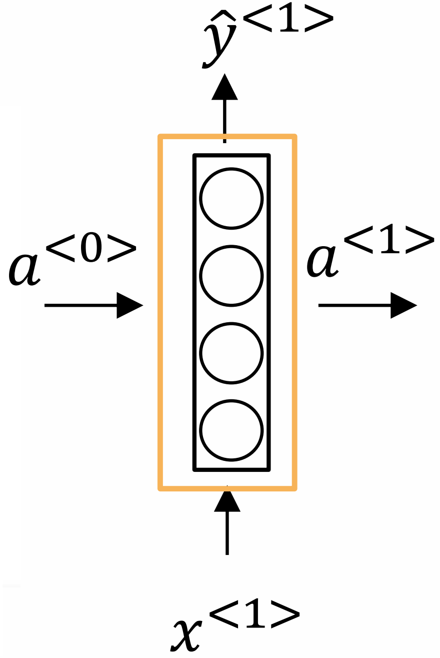
Adopted from Andrew Ng

Skoltech
Skolkovo Institute of Science and Technology

Selection of RNN architecture

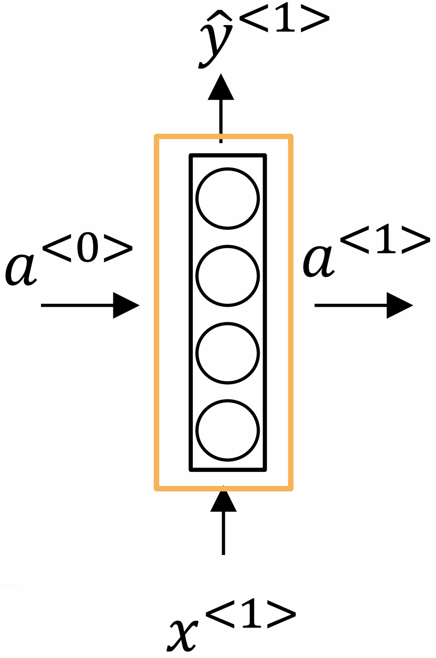
Simplest RNN unit: what is going on inside yellow rectangle?

- $RNN^*(x^{<1:n>}, a^{<0>}) = y^{<1:n>}$
- $y^{<i>} = g(W^{out}[a^{<i>}, x^{<i>}] + b)$



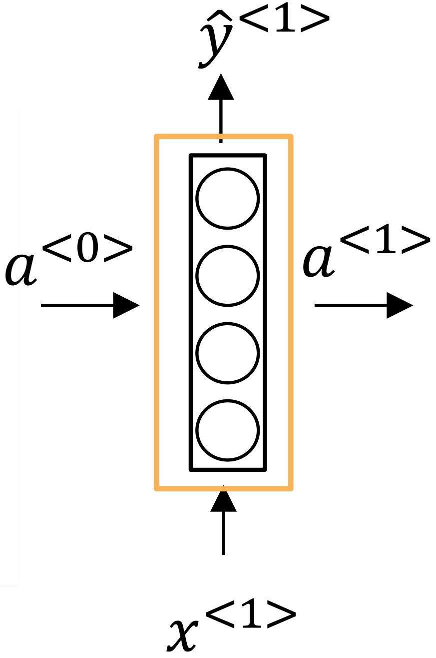
Simplest RNN unit: what is going on inside yellow rectangle?

- $RNN^*(x^{<1:n>}, a^{<0>}) = y^{<1:n>}$
- $y^{<i>} = g(W^{out}[a^{<i>}, x^{<i>}] + b)$
- R is a recursive activation function. It depends on inputs $x^{<t>}$ and output of the previous state $a_{<t-1>}$ (vector of the previous state)
- $a^{<i>} = R(a^{<i-1>}, x^{<i>})$
- $R(a^{<i-1>}, x^{<i>}) = g(W^{hid}[a^{<i-1>}, x^{<i>}] + b)$, $[a^{<i>}, x^{<i>}]$ is the concatenation of $a^{<i>}$ and $x^{<i>}$
- $x^{<i>} \in \mathbb{R}^{d_{in}}, y^{<i>} \in \mathbb{R}^{d_{out}}, a^{<i>} \in \mathbb{R}^{d_{hid}}$

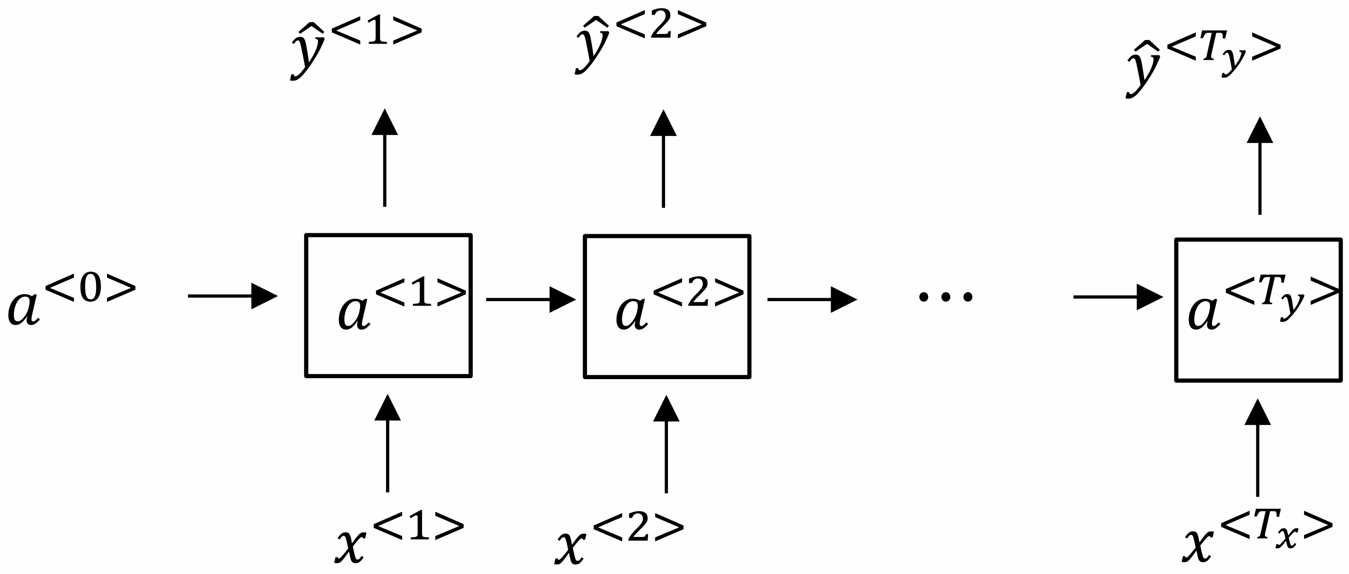


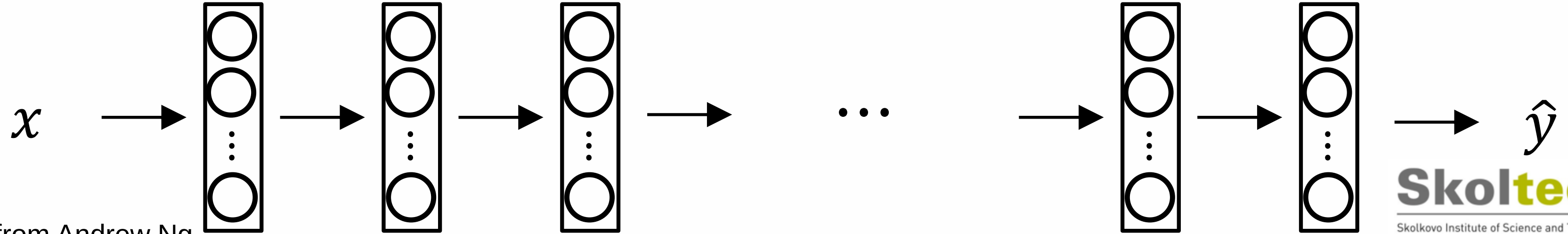
Simplest RNN unit: what is going on inside yellow rectangle?

- $RNN^*(x^{<1:n>}, a^{<0>}) = y^{<1:n>}$
- $y^{<i>} = g(W^{out}[a^{<i>}, x^{<i>}] + b)$
- R is a recursive activation function. It depends on inputs $x^{<t>}$ and output of the previous state $a_{<t-1>}$ (vector of the previous state)
- $a^{<i>} = R(a^{<i-1>}, x^{<i>})$
- $R(a^{<i-1>}, x^{<i>}) = g(W^{hid}[a^{<i-1>}, x^{<i>}] + b)$, $[a^{<i>}, x^{<i>}]$ is the concatenation of $a^{<i>}$ and $x^{<i>}$
- $x^{<i>} \in \mathbb{R}^{d_{in}}, y^{<i>} \in \mathbb{R}^{d_{out}}, a^{<i>} \in \mathbb{R}^{d_{hid}}$
- Parameters of Neural Network are $W^{hid} \in \mathbb{R}^{(d_{in}+d_{out})\times d_{hid}}, W^{out} \in \mathbb{R}^{d_{hid}\times d_{out}}$



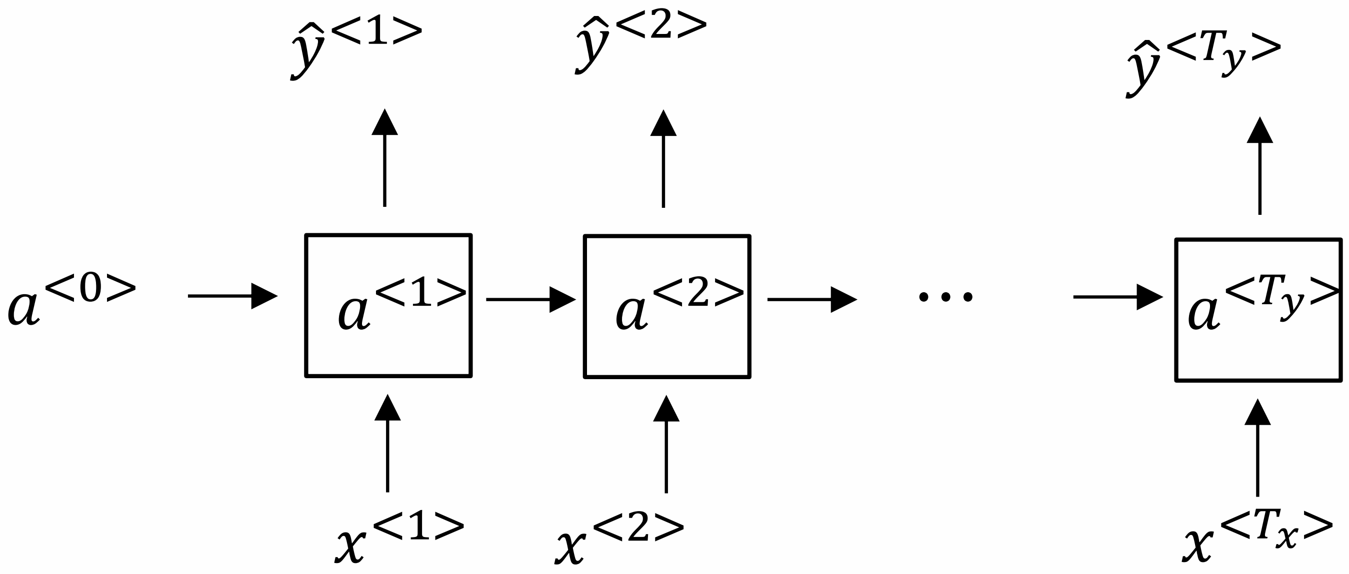
Gradient vanish and/or explode





Adopted from Andrew Ng

Gradient vanish and/or explode

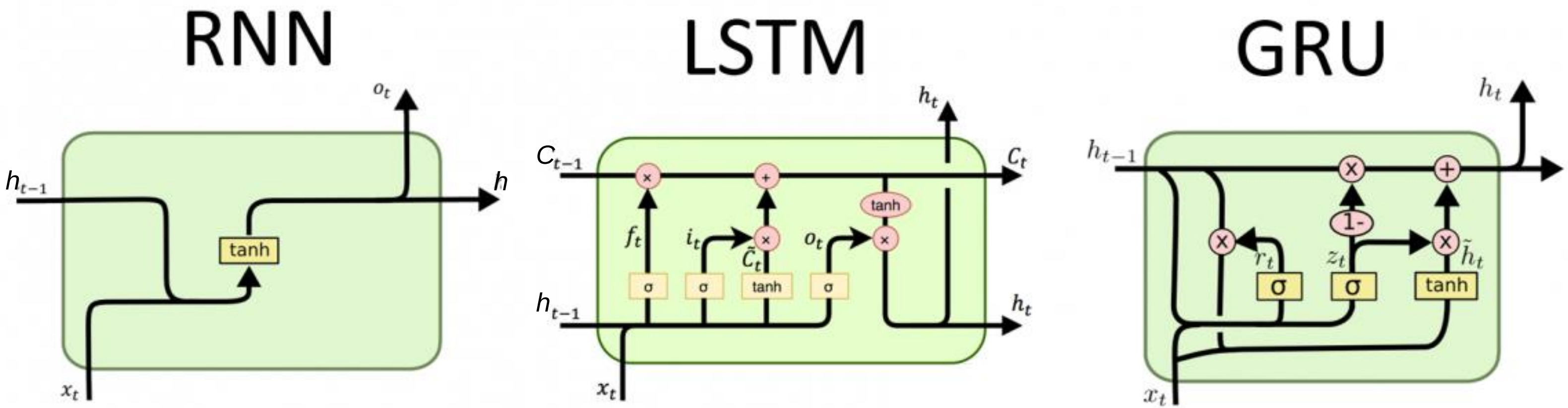


Neural networks forget fast, and it is hard to learn long-term dependencies

Adopted from Andrew Ng

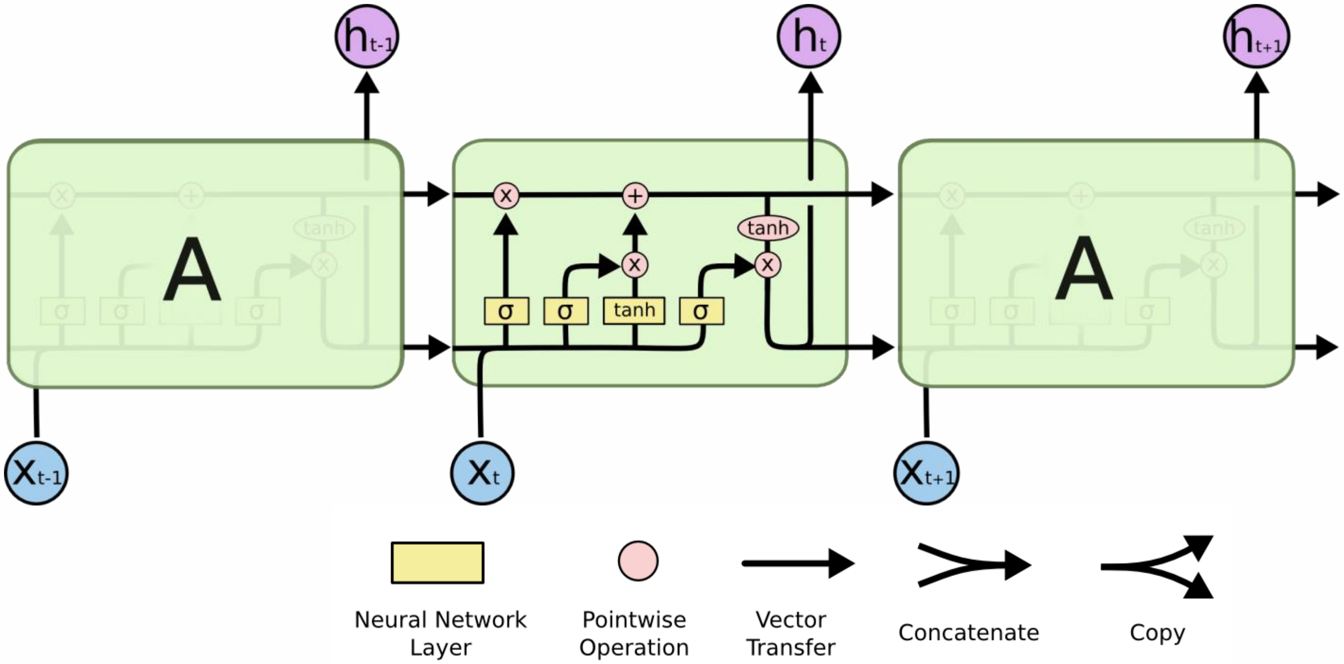
Better RNN units: LSTM and GRU

- LSTM: long short term memory
- GRU: Gated recurrent unit



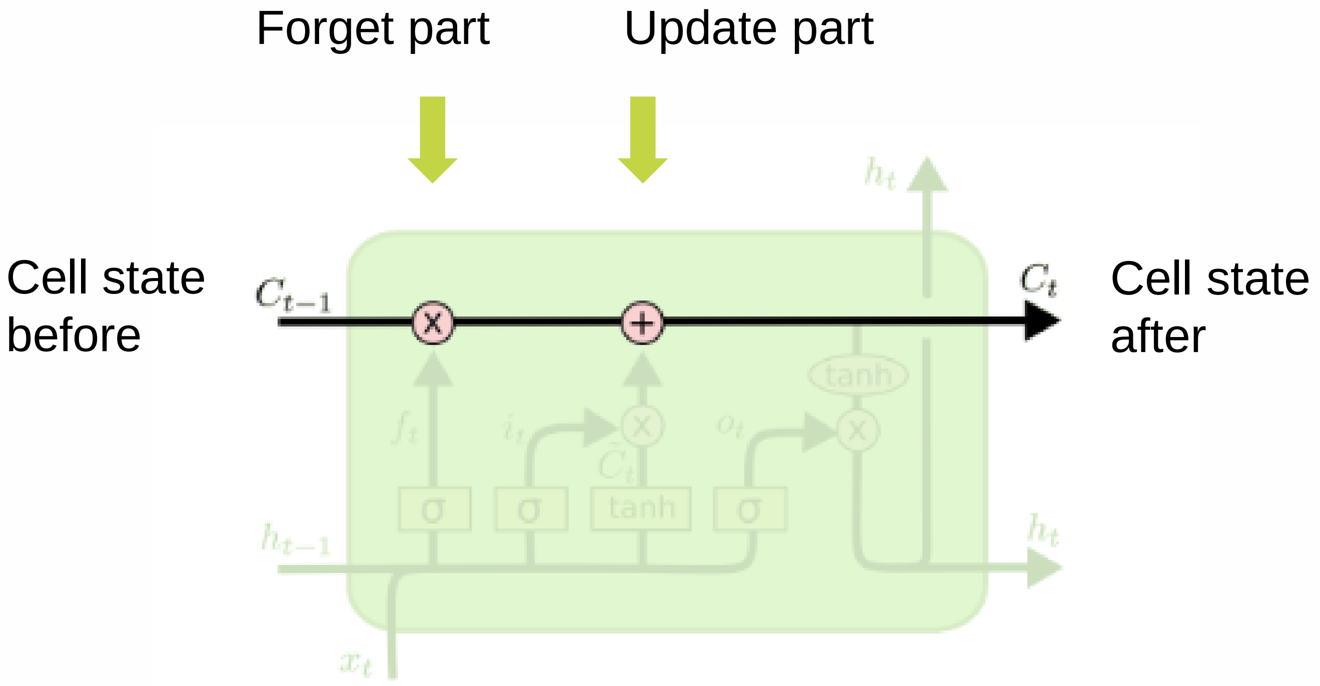
Details on how LSTM works

Remembering information for long periods of time is practically the default behavior of LSTM



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

Long term memory part – Cell state

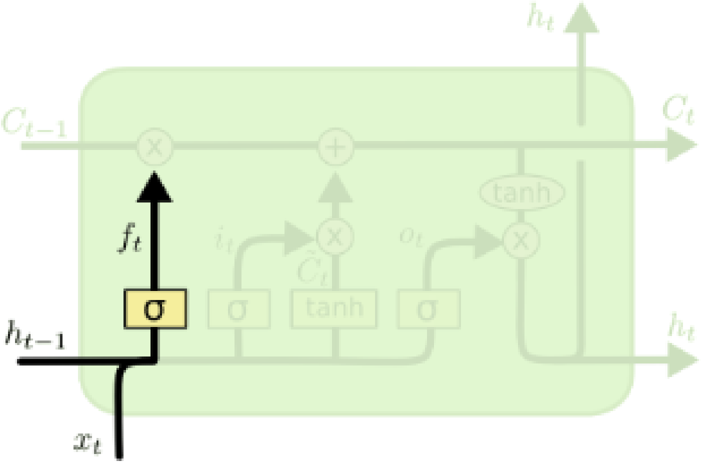


http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Forget part

Identify how much should we forget:
sigmoid returns value between 0 and 1

hidden
state (a)



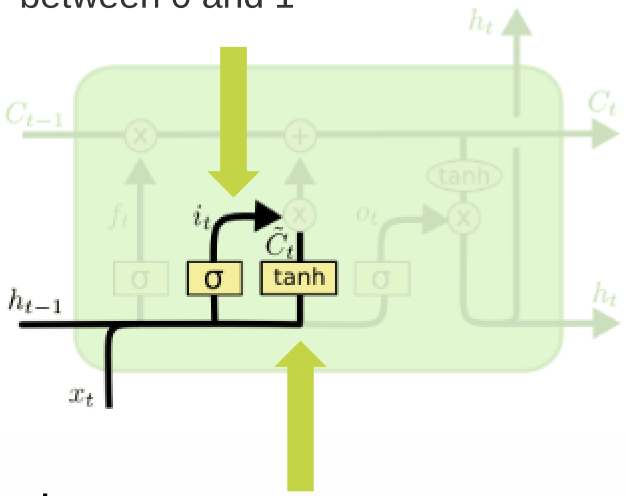
inputs

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

Update part

Identify how much should we update: sigmoid returns value between 0 and 1

hidden state (a)



inputs

What update we do

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

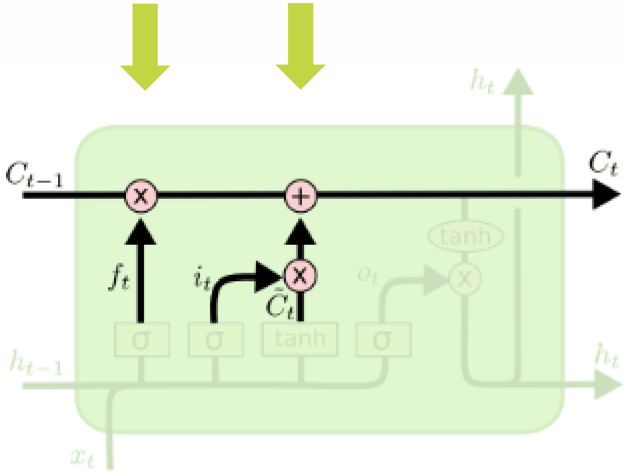
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Long term memory part – Cell state

Forget part

Update part

Cell state before

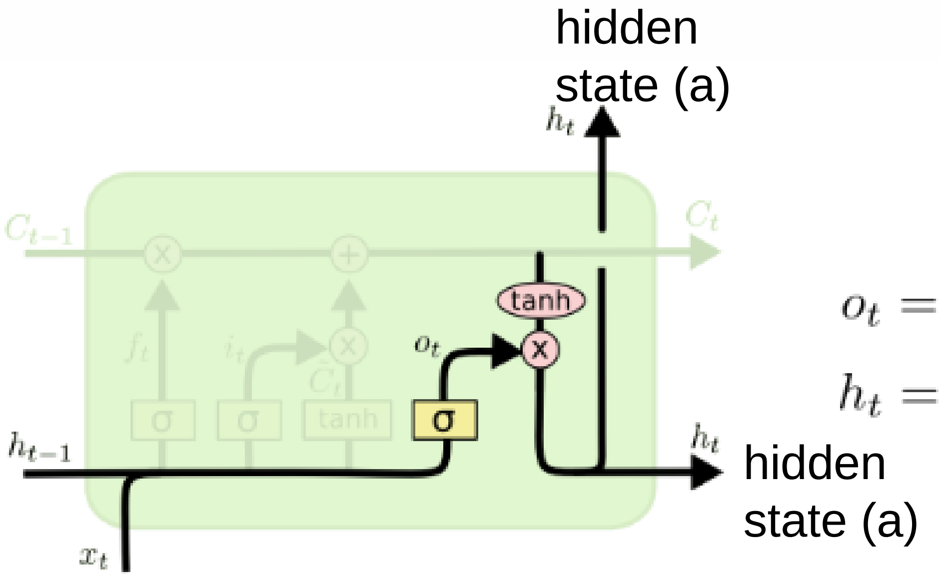


Cell state after

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

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

Update everything else



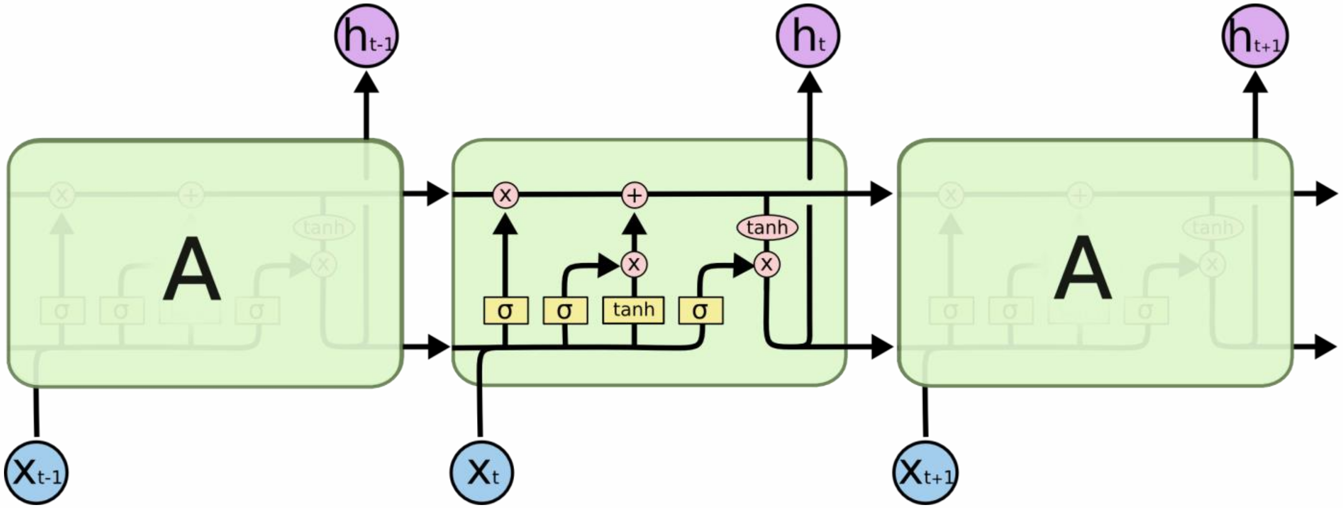
$$o_t = \sigma\left(W_o \left[h_{t-1}, x_t\right] + b_o\right)$$

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

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

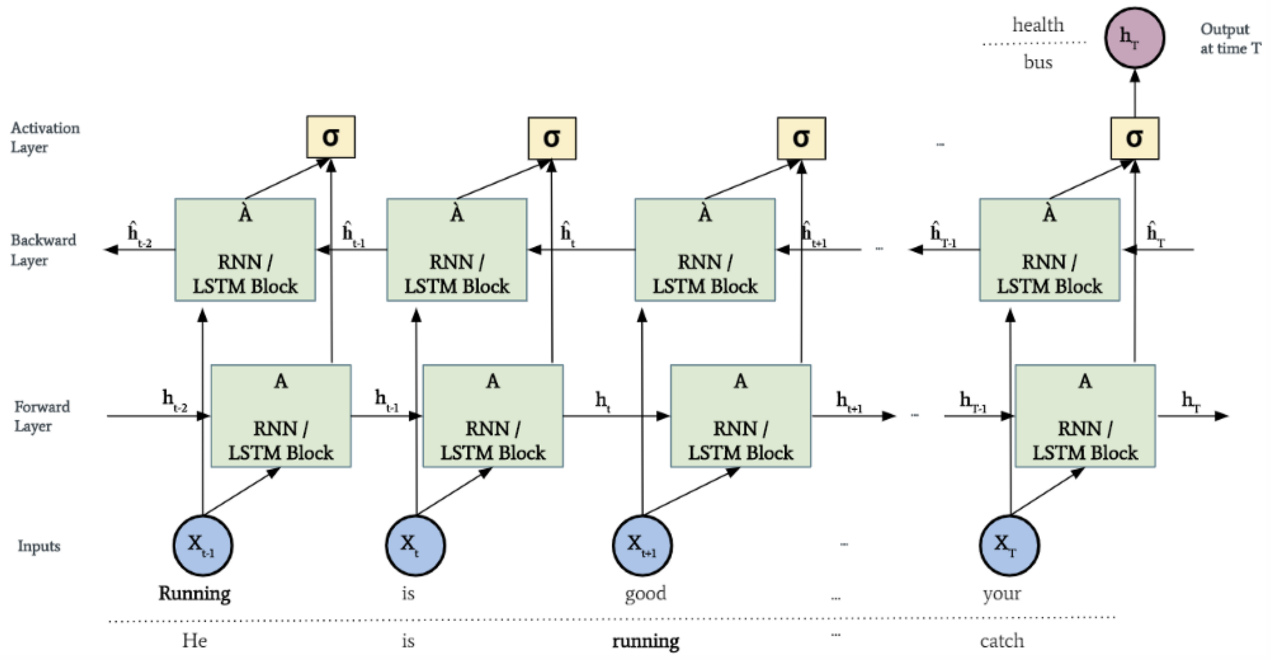
Details on how LSTM works

- There are cell and hidden (activation) states
- LSTM block forgets and updates cell state during processing at one block



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

Other architectures: bidirectional LSTM

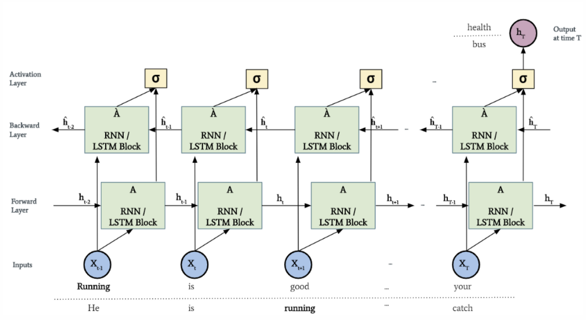


https://mc.ai/review-generation-using-bidirectional-long-short-term-memorylstm/

Other architectures: bidirectional LSTM

Bidirectional LSTM are useful when we benefit from the future data:

- handwriting recognition
- speech recognition
- protein Structure Prediction (bioinformatics)



https://mc.ai/review-generation-using-bidirectional-long-short-term-memorylstm/

Skoltech
Skolkovo Institute of Science and Technology

Take-home messages

- For some types of data classic methods fail:
  we need to learn a representation i.e. extract features automatically

- Neural Networks provide enough flexibility for this problem for various data types

- The basic architecture is Recurrent Neural Network RNN

- But we can do better in terms of keeping the necessary information with LSTM and GRU blocks/architectures

Skoltech
Skolkovo Institute of Science and Technology