

# Lecture 9. Recurrent Neural Networks

Alex Avdyushenko

Kazakh-British Technical University

November 5, 2022



# Five-minutes block

## Five-minutes block

- Write down several names of neural network optimization methods
- Describe a couple of regularization methods for training neural networks
- Draw (or write down) how the skip-connection block works

# Disadvantages of Convolutional Neural Networks

Or why we need recurrent ones :)

# Disadvantages of Convolutional Neural Networks

Or why we need recurrent ones :)

- the input is only fixed-dimensional vectors (e.g.  $28 \times 28$  images)

# Disadvantages of Convolutional Neural Networks

Or why we need recurrent ones :)

- the input is only fixed-dimensional vectors (e.g.  $28 \times 28$  images)
- the output is also a fixed dimension (for example, probabilities of 1000 classes)

# Disadvantages of Convolutional Neural Networks

Or why we need recurrent ones :)

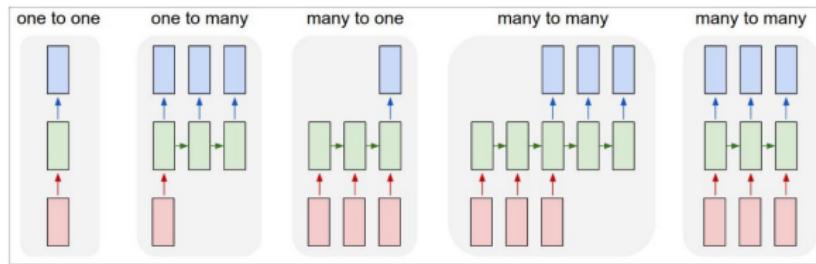
- the input is only fixed-dimensional vectors (e.g.  $28 \times 28$  images)
- the output is also a fixed dimension (for example, probabilities of 1000 classes)
- fixed number of computational steps (i.e. network architecture)

# Disadvantages of Convolutional Neural Networks

Or why we need recurrent ones :)

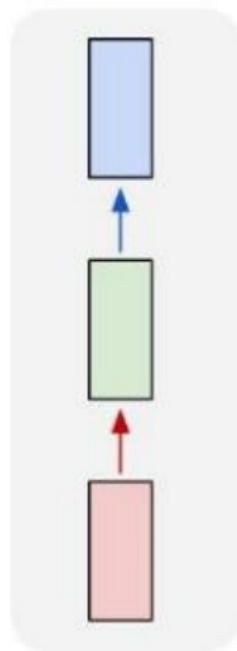
- the input is only fixed-dimensional vectors (e.g.  $28 \times 28$  images)
- the output is also a fixed dimension (for example, probabilities of 1000 classes)
- fixed number of computational steps (i.e. network architecture)

## A. Karpathy. The Unreasonable Effectiveness of Recurrent Neural Networks

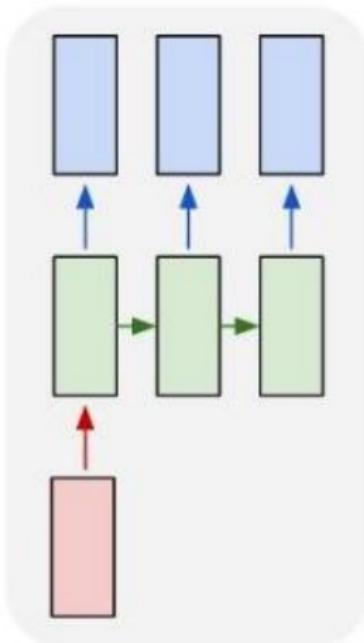


# Architectures of Recurrent Networks

one to one



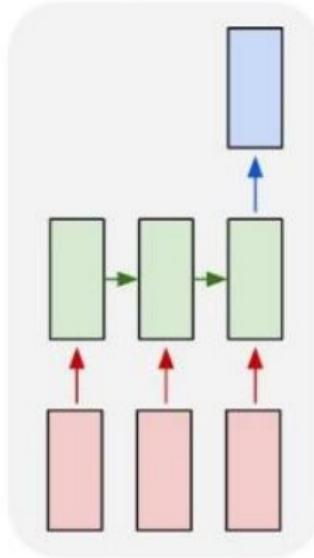
one to many



Vanilla Neural Networks

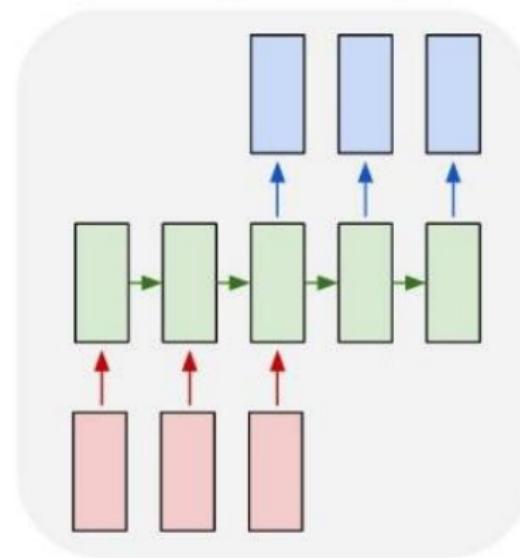
Image Captioning  
image → (sequence of words)

many to one



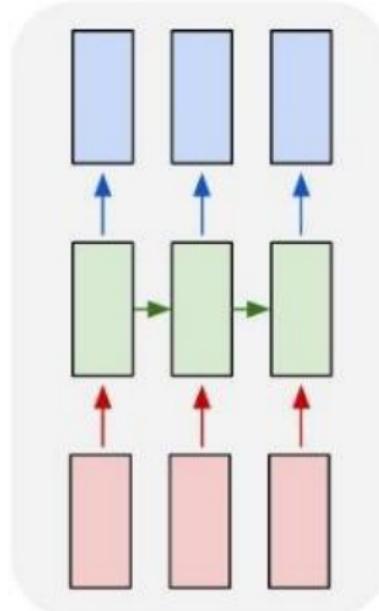
Sentiment Classification  
(sequence of words) → sentiment

many to many



Machine Translation  
(seq of words) → (seq of words)

many to many



Video Classification  
(on frame level)

# Sequential Processing of Fixed Input

---

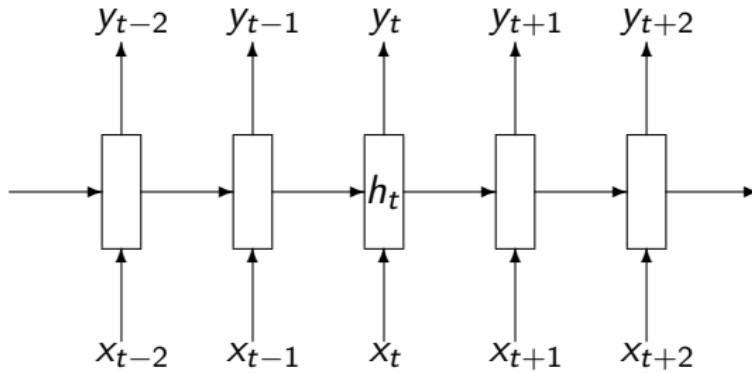
J. Ba, V. Mnih, K. Kavukcuoglu. Multiple Object Recognition with Visual Attention

# Sequential Generation of Fixed Output

---

K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, D. Wierstra. DRAW: A Recurrent Neural Network For Image Generation

# Recurrent Neural Network scheme



# Recurrent Neural Network

We process the sequence of vectors  $x$  with **one and the same** function with parameters:

$$h_t = f_W(h_{t-1}, x_t)$$

$f_W$  is a function parameterized by  $W$

$x_t$  — next input vector

$h_t$  — hidden state

# Recurrent Neural Network

We process the sequence of vectors  $x$  with **one and the same** function with parameters:

$$h_t = f_W(h_{t-1}, x_t)$$

$f_W$  is a function parameterized by  $W$

$x_t$  — next input vector

$h_t$  — hidden state

## Question

What function can we take as  $f_W$ ?

# Vanilla Recurrent Neural Network

$$h_t = f_W(h_{t-1}, x_t)$$

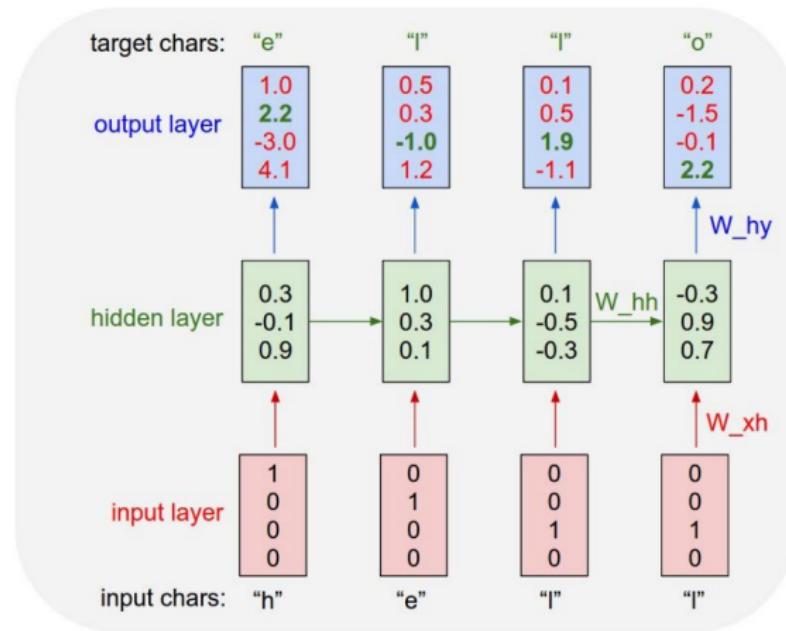
As a function  $f_W$  we set a linear transformation with a non-linear component-wise "sigmoid":

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

$$y_t = W_{hy}h_t$$

# Character level model example

The entire four-letter dictionary:  $[h, e, l, o]$  and word “hello” as train:



**Softmax** is also applied to the values of the output layer to obtain the loss function

# Demo

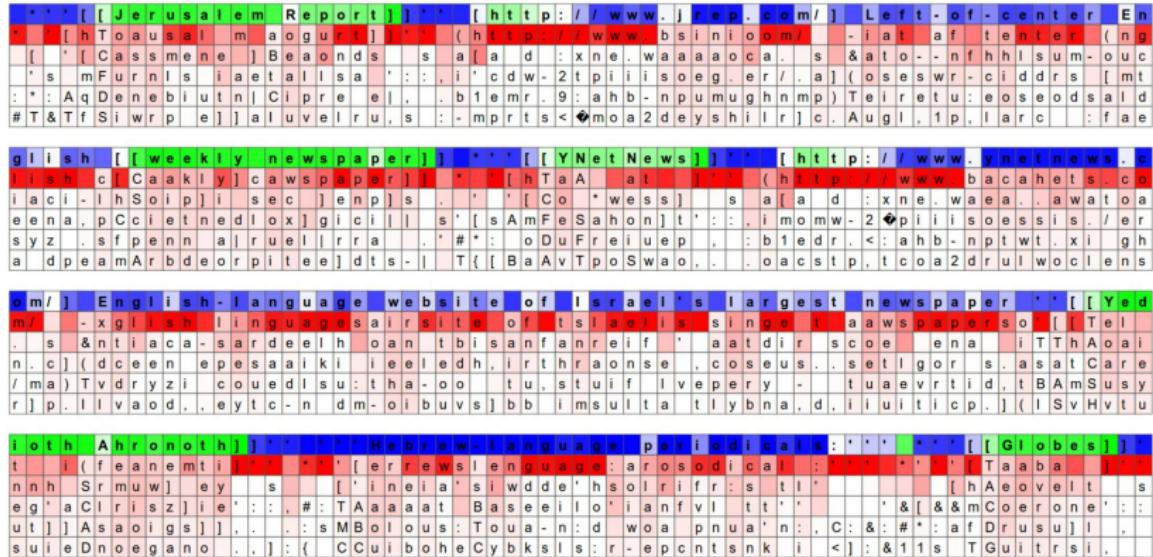
**numpy** implementation by Karpathy

Let's get to grips with the code in jupyter notebook!

## How does it work?

http://www.ynetnews.com/] English-language website of Israel's largest news website [http://www.bacharach.com/] - xglish-languagesairsite of Israel's singed news website [xne.waea.awatoa.s &ntiacasardeelh oanbtisanfanreif'aad mw-2phiisessis./ern.c] (dcean epesaaikiieeedh,irthraonse, cose dr.<ahb-nptwt.xi ghma) Tvdryzi couedslus:tha-oo tu,stuif lveperystp.tcoa2drulwoc tensr] p.lvaod.,eytc-n dm-oibuvsv] bbb imsulta tlybngest newspaper '[[Yedidot Ahronoth]]' '[[Hebrew-language periodical]]' awspaperson[[Tel Aviv feanemtii]]' '[[errewslanguage:arosodi irscoe ena iTThAoainnh Srmuw] ey s [[ineia'siwddehsolrifr: us..setlgors.asatCareeg'AClrisz] ie':.:.#:TAAAAAT Baseeilo'ianfvi -tuaevrtid,tBAmSusyut]] Asaoigs]],..:sMBolous:Touan-d woapnu a.d.iiuiticp.][ISvHvtusuiDnoegano .]:{CCuibohCybkls:r-epcnts icals:'[[Globes]]' '[http://www.globes.co.il/] business daily :[[Taaba]]' '[[http://www.buobal.com/SA-ytinessaet stl':[[hAeoelt sahad xge.woirr.tao.el.iT&ai eg eoy tt'&[&McOerone':.i'odw.:niiisaue.eni/omlcC.(eftgir iu a'n:.C:&:#:afDrusu]] .omel p.<.dha;deuoot/ihncsifS, urhost, tun nk i <]:&11s TGuitrsi, :bacmr-xtpob-gresislerlnafad] losptad, ifrm illy \*'[[Haaretz|Ha'aretz]]' '[http://www.haaretz.co.il/] Relatively \*'[[Terrdn Ferantah]]' '[[http://www.bonmdst.comun/s -esated! re.'hAilnntteHalsrcnol'saha d:xne.waamrtdeoh.ol.c &opinive kii: \*'CO Sanit hiTim'lie' .imcdw-2phiiserdit.ina/cmfi.(aficana ds-![[tBTCommgd]] Won aae,:baerr.<taib-dulcnncc/arne si]] liceysto nds#: GI DuvccsaoSucitel] zl, :o'omt],:eo a2nivfsrooeiunala) uvvro

The neuron highlighted in this image seems to get very excited about URLs and turns off outside of the URLs. The LSTM is likely using this neuron to remember if it is inside a URL or not.



Cell sensitive to position in line:

The sole importance of the crossing of the Berezina lies in the fact that it plainly and indubitably proved the fallacy of all the plans for cutting off the enemy's retreat and the soundness of the only possible line of action--the one Kutuzov and the general mass of the army demanded--namely, simply to follow the enemy up. The French crowd fled at a continually increasing speed and all its energy was directed to reaching its goal. It fled like a wounded animal and it was impossible to block its path. This was shown not so much by the arrangements it made for crossing as by what took place at the bridges. When the bridges broke down, unarmed soldiers, people from Moscow and women with children who were with the French transport, all--carried on by vis inertiae--pressed forward into boats and into the ice-covered water and did not, surrender.

Cell that turns on inside quotes:

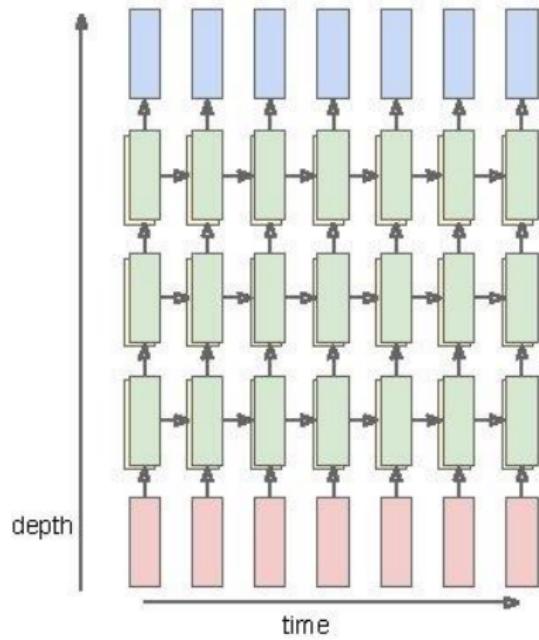
"You mean to imply that I have nothing to eat out of.... On the contrary, I can supply you with everything even if you want to give dinner parties," warmly replied Chichagov, who tried by every word he spoke to prove his own rectitude and therefore imagined Kutuzov to be animated by the same desire.

Kutuzov, shrugging his shoulders, replied with his subtle penetrating smile: "I meant merely to say what I said."

# Deep recurrent networks

$$h_t^\ell = \tanh W^\ell \begin{pmatrix} h_t^{\ell-1} \\ h_{t-1}^\ell \end{pmatrix}$$

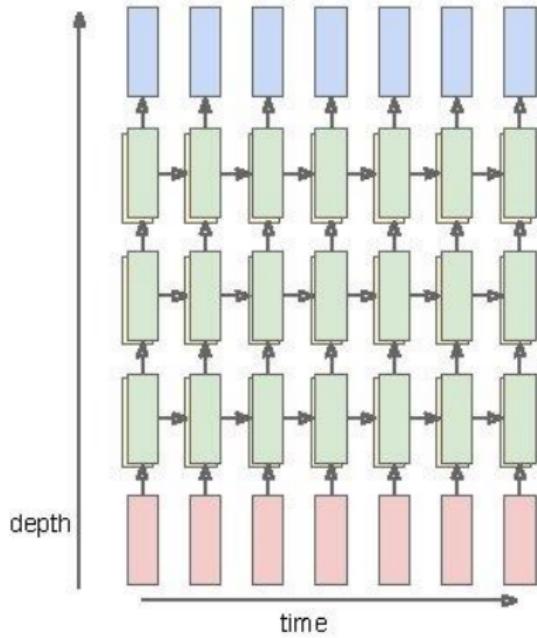
$h \in \mathbb{R}^n, \quad W^\ell[n \times 2n]$



# Deep recurrent networks

$$h_t^\ell = \tanh W^\ell \begin{pmatrix} h_t^{\ell-1} \\ h_{t-1}^\ell \end{pmatrix}$$

$h \in \mathbb{R}^n, \quad W^\ell[n \times 2n]$



## Question

What is the main problem with vanilla RNN?

# Long short-term memory (LSTM)

$$\begin{aligned} W^\ell[4n \times 2n] \\ \begin{pmatrix} i \\ f \\ o \\ c'_t \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \tanh \end{pmatrix} W^\ell \begin{pmatrix} h_t^{\ell-1} \\ h_{t-1}^\ell \end{pmatrix} \\ c_t^\ell = f \odot c_{t-1}^\ell + i \odot c'_t \\ h_t^\ell = o \odot \tanh(c_t^\ell) \end{aligned}$$

$\odot$  — component-wise product

# LSTM: Motivation and Schema

The network should remember the context for a long time. Which context? Network learns itself. To do this, the vector  $c_t$  is introduced, which is the state vector of the network at the moment  $t$ .

$$c'_t = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_{c'}) \quad \text{candidate cell state}$$

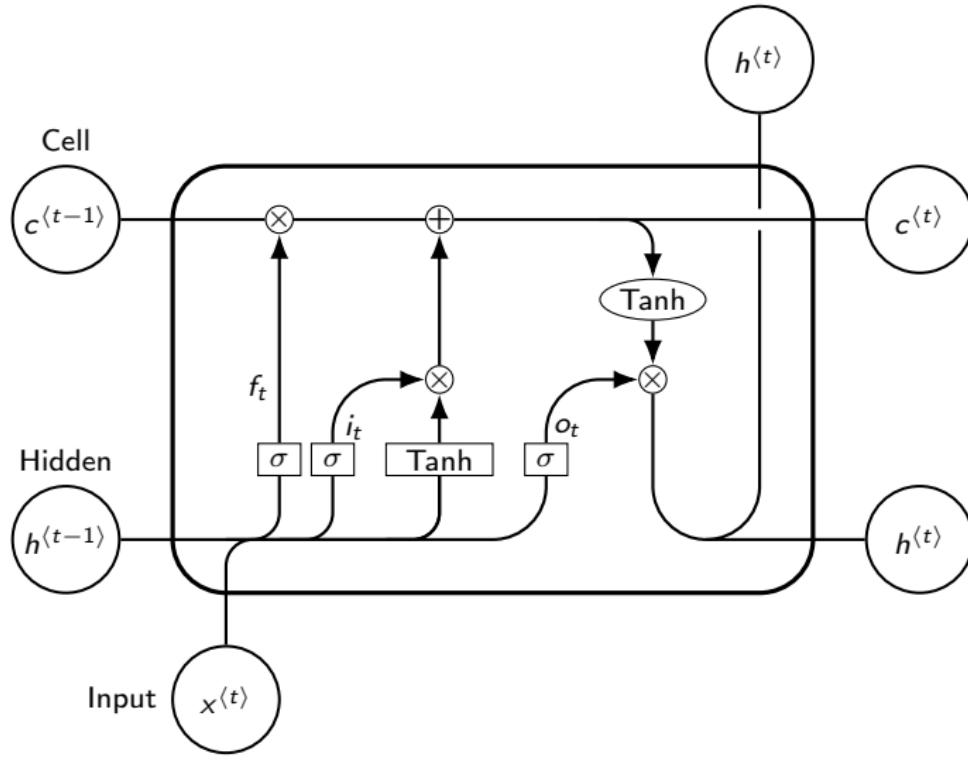
$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \quad \text{input gate}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \quad \text{forget gate}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \quad \text{output gate}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot c'_t \quad \text{cell state}$$

$$h_t = o_t \odot \tanh(c_t) \quad \text{block output}$$



## LSTM: forget gate

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

Example: Removing the Gender of an Actor When Generating Text

# LSTM: input/candidate gates

$$c'_t = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_{c'}) \quad \text{candidate cell state}$$
$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \quad \text{input gate}$$

Example: Adding the Gender of an Actor When Generating Text

## LSTM: updating the state of a memory cell

$$c_t = f_t \odot c_{t-1} + i_t \odot c'_t \quad (\text{cell state})$$

The new state  $c_t$  is obtained by the sum of the previous state  $c_{t-1}$  with the filter  $f_t$  and the vector of candidate values  $c'_t$  with the filter  $i_t$

# LSTM: output generation

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \quad \text{output gate}$$
$$h_t = o_t \odot \tanh(c_t) \quad \text{block output}$$

# GRU: Gated Recurrent Unit

$$u_t = \sigma(W_{xu}x_t + W_{hu}h_{t-1} + b_u)$$

$$r_t = \sigma(W_{xr}x_t + W_{hr}h_{t-1} + b_r)$$

$$h'_t = \tanh(W_{xh'}x_t + W_{hh'}(r_t \odot h_{t-1}))$$

$$h_t = (1 - u_t) \odot h'_t + u_t \odot h_{t-1}$$

Only  $h_t$  is used, vector  $c_t$  is not introduced.

Update-gate instead of input and forget.

The reset gate determines how much memory to move forward from the previous step.

# Disadvantages of Vanilla RNN

$$h_t = f_W(h_{t-1}, x_t)$$

As a function  $f_W$  we set a linear transformation with a non-linear component-wise “sigmoid”:

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

$$y_t = W_{hy}h_t$$

# Disadvantages of Vanilla RNN

$$h_t = f_W(h_{t-1}, x_t)$$

As a function  $f_W$  we set a linear transformation with a non-linear component-wise “sigmoid”:

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$
$$y_t = W_{hy}h_t$$

## Disadvantages

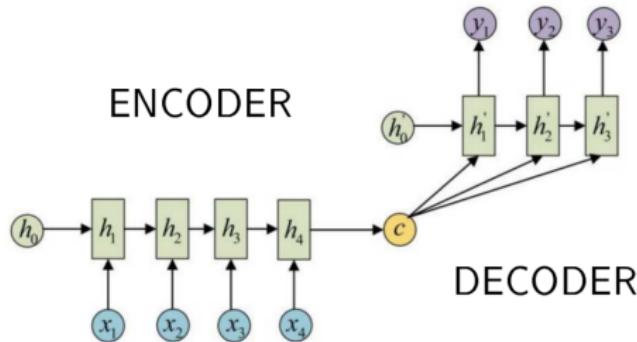
- ① input and output signal lengths must match
- ② “reads” the input only from left to right, does not look ahead
- ③ therefore is not suitable for machine translation, question answering tasks and others

# RNN for sequence synthesis (seq2seq)

$X = (x_1, \dots, x_n)$  — input sequence

$Y = (y_1, \dots, y_m)$  — output sequence

$c \equiv h_n$  encodes all information about  $X$  to synthesize  $Y$



$$h_i = f_{in}(x_i, h_{i-1})$$

$$h'_t = f_{out}(h'_{t-1}, y_{t-1}, c)$$

$$y_t = f_y(h'_t, y_{t-1})$$

# Summary

- Recurrent neural networks — a simple, powerful and flexible approach to solving various machine learning problems
- Vanilla RNNs are simple, but still not good enough
- So you need to use LSTM or GRU, and seq2seq architecture
- LSTM prevents zeroing gradients
- Clipping helps with «explosion of gradients»
- We need deeper understanding, both theoretical and practical — there are many open questions

# Summary

- Recurrent neural networks — a simple, powerful and flexible approach to solving various machine learning problems
- Vanilla RNNs are simple, but still not good enough
- So you need to use LSTM or GRU, and seq2seq architecture
- LSTM prevents zeroing gradients
- Clipping helps with «explosion of gradients»
- We need deeper understanding, both theoretical and practical — there are many open questions

What else can you see?

- [Lecture 10](#) of the course «CS231n» by Andrej Karpathy at Stanford
- [How to train neural networks?](#)
- [Similar lecture](#) of the School of Data Analysis course (in Russian)
- [Fresh interview](#) of Andrej Karpathy by Lex Fridman