Neural Networks for sequential data

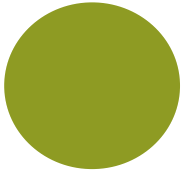
28 April, 2020

Alexey Zaytsev, head of Laboratory LARSS, PhD

Foundations of Data Science

Skoltech
Skolkovo Institute of Science and Technology

How to predict the position of a ball at time (T + 1)?



Position at time T

Adopted from Ava Soleimany, MIT

How to predict the position of a ball at time (T + 1)?

Sequential data is essential for some prediction problems

Adopted from Ava Soleimany, MIT

Skoltech
Skolkovo Institute of Science and Technology

ML problems with sequential data

	Input		Output
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	→	AG**CCCCTGTGAGGAACT**AG
Machine translation	Voulez-vous chanter avec moi?	→	Do you want to sing with me?
Video activity recognition		→	Running

Adopted from Andrew Ng

Skoltech
Skolkovo Institute of Science and Technology

Textbook example: next word prediction

The most complicated and difficult part of it was only just beginning.

Adopted from Ava Soleimany, MIT

Textbook example: next word prediction

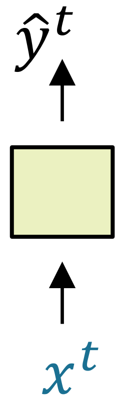
Idea 1: use previous word(s)

Problem 1: long-term dependencies

"**France** is where I grew up, but I now live in Boston. I speak fluent ____."

The most complicated and difficult part of it was only just beginning.

Feature representation: [0, 0, 0, 1, 0, 0]



Adopted from Ava Soleimany, MIT

Skoltech
Skolkovo Institute of Science and Technology

Textbook example: next word prediction

Idea 1: use previous word(s)
Idea 2: use bag of words model

Problem 1: long-term dependencies

The most complicated and difficult part of it was only just beginning.

Feature representation: [0, 3, 0, 2, 0, 0]

Bag of words: number of occurrences of each word

Adopted from Ava Soleimany, MIT

Textbook example: next word prediction

Idea 1: use previous word(s)
Idea 2: use bag of words model

Problem 1: long-term dependencies
Problem 2: order preservation

The food was good, not bad at all.

vs.

The food was bad, not good at all.

The most complicated and difficult part of it was only just beginning.

Feature representation: [0, 3, 0, 2, 0, 0]

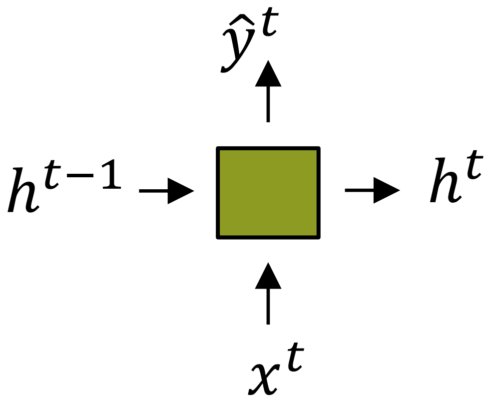
Bag of words: number of occurrences of each word

Adopted from Ava Soleimany, MIT

Skoltech
Skolkovo Institute of Science and Technology

Model Design Criteria

- Long-term memory
- Maintain order information
- Natural preprocessing
- Variable-length sequences processing

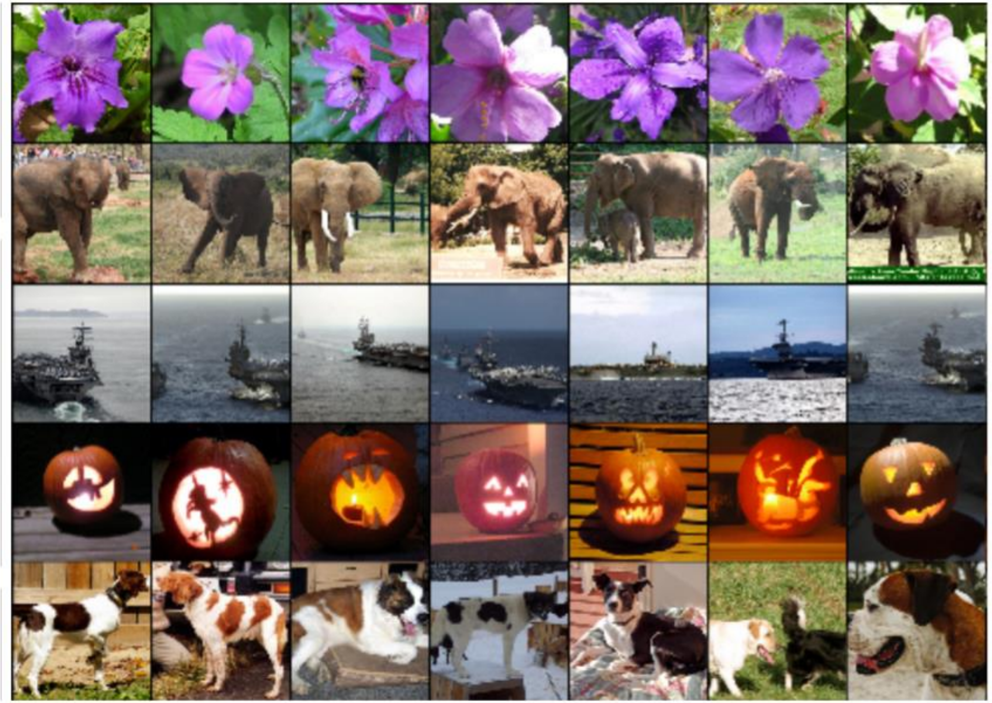


Recurrent Neural Networks are the solution!
…more on this later

Adopted from Ava Soleimany, MIT

Why bother with Neural Networks for sequential data?

Deep learning is better than a human or another machine learning model



Human error percentage is

5.1%

DL error percentage is

3.57%

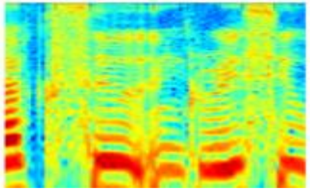
https://arxiv.org/pdf/1705.02498.pdf
https://qz.com/1034972/the-data-that-changed-the-direction-of-ai-research-and-possibly-the-world/
http://www.image-net.org/papers/imagenet_cvpr09.pdf

Classic approach to Machine Learning

Data → Features → Learning

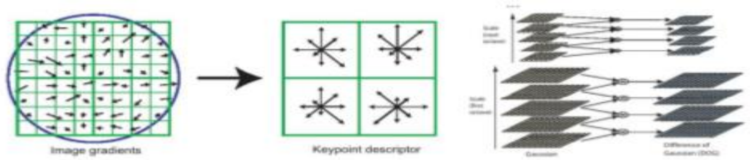


Image

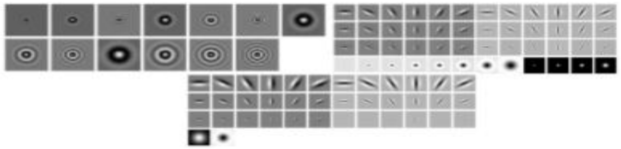




Speech

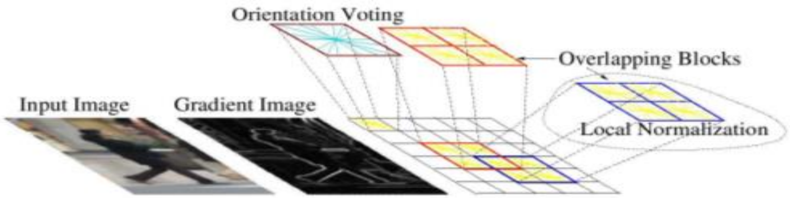
An art of feature construction



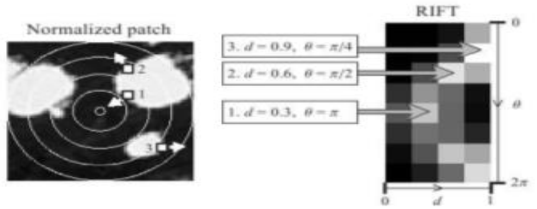
SIFT



Textons

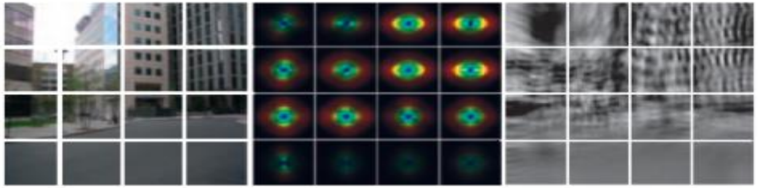


HoG

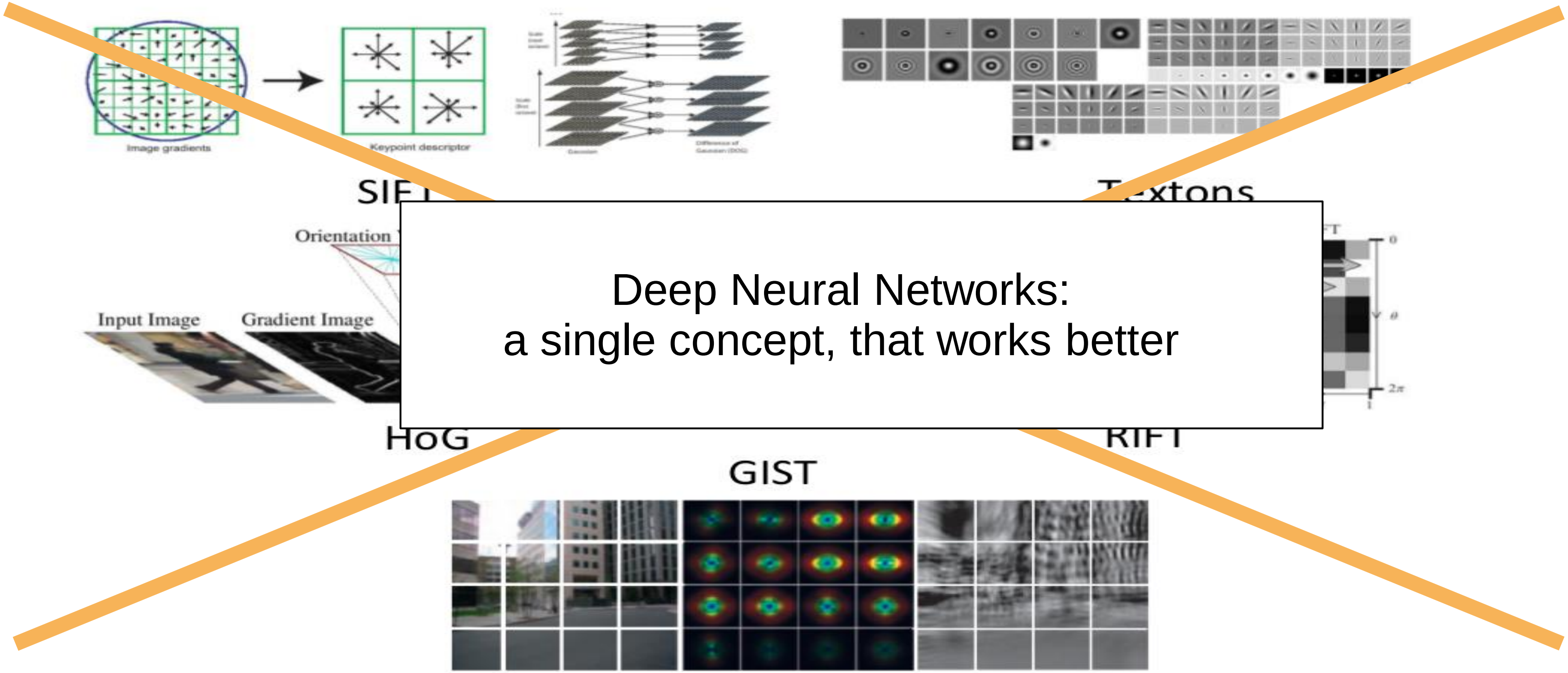


RIFT

GIST



No art now, just engineering



Deep Neural Networks:
a single concept, that works better

Why and how Deep learning works

- Availability of Graphical processing units (GPU)

- Large scale "big" data

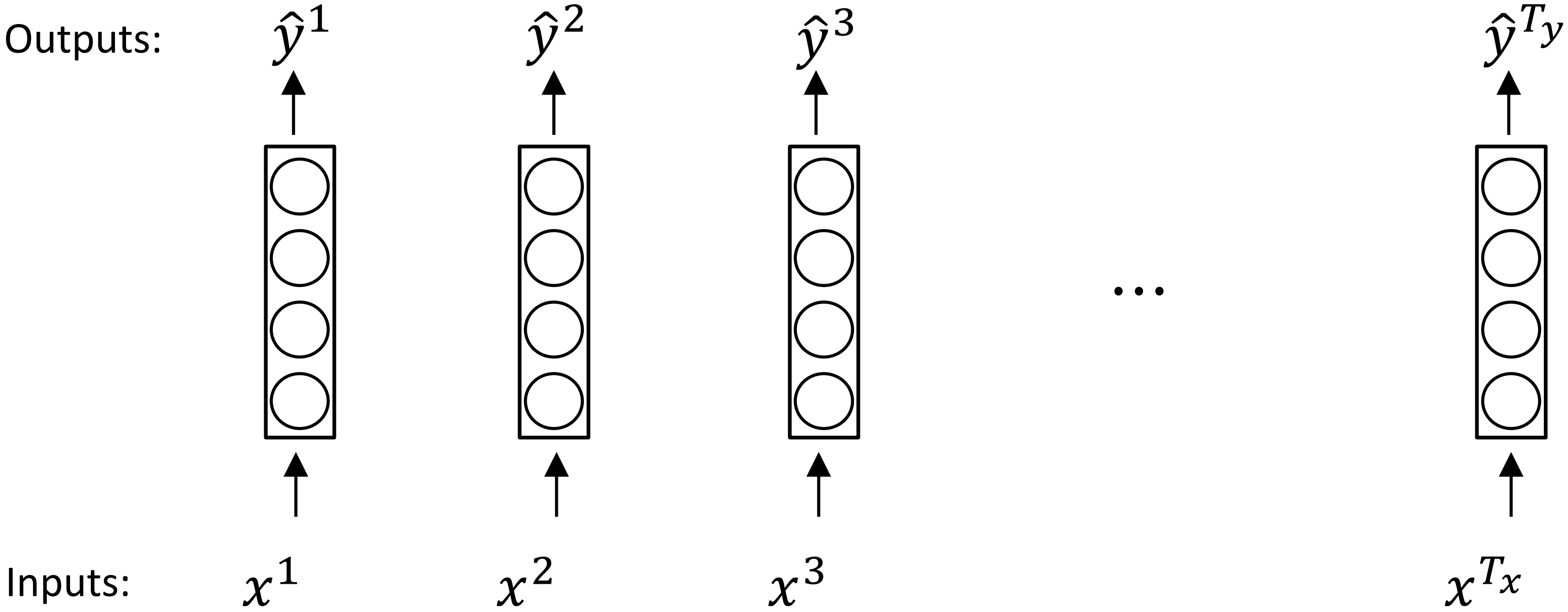
- Open-source libraries

- Complex structured data

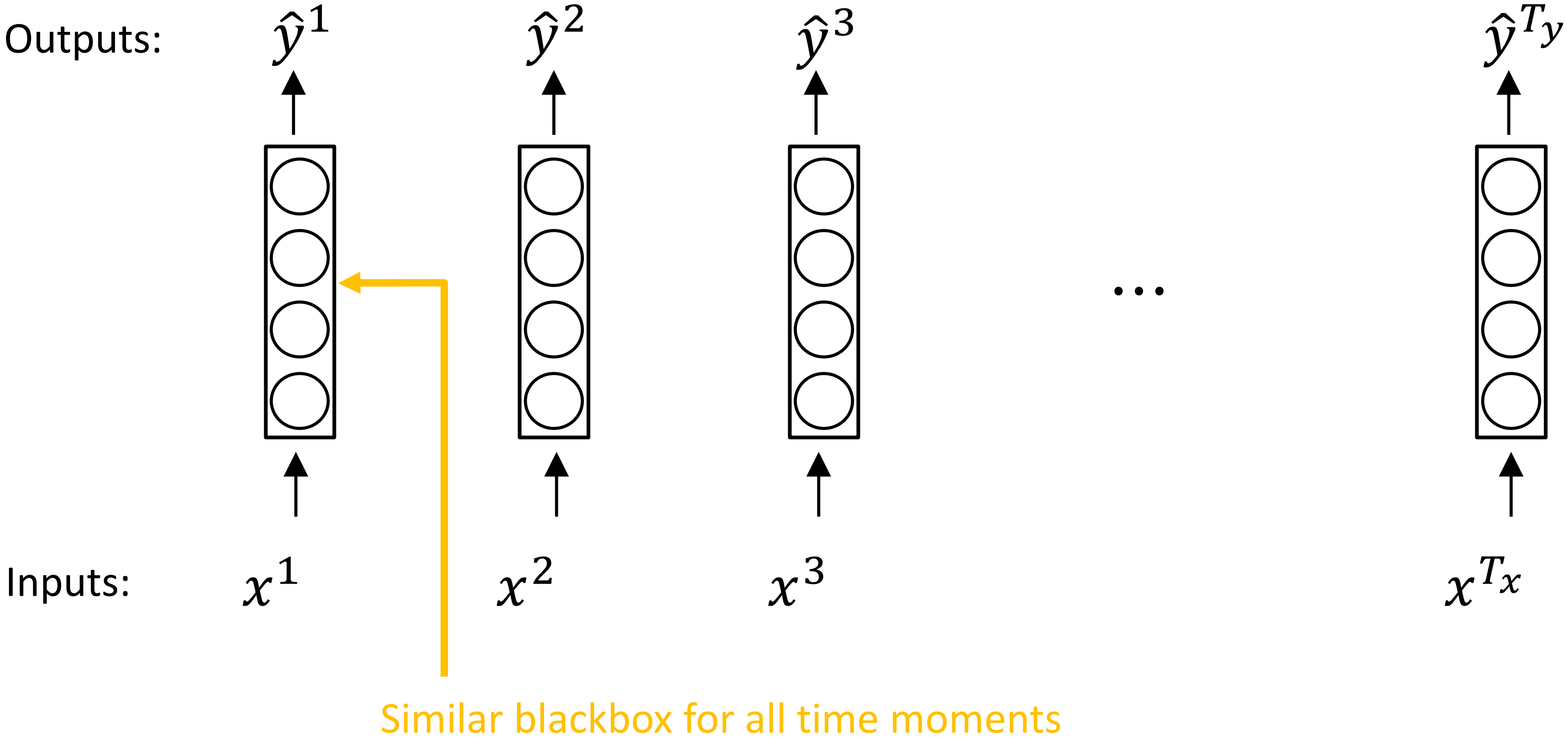
Sequence data examples

		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		+	+

Separate Fully-Connected Neural Networks or other separate models



Separate Fully-Connected Neural Networks or other separate models



Skoltech
Skolkovo Institute of Science and Technology

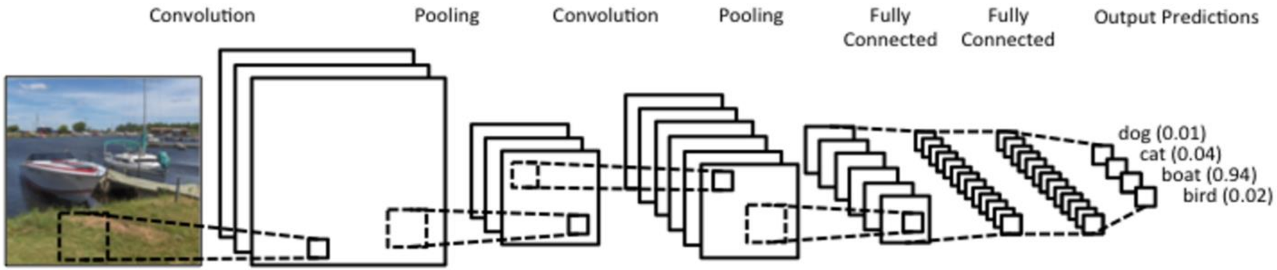
Sequence processing with separate models

• Long-term memory	**NO**
• Maintain order information	**NO**
• Natural preprocessing	**a kind of**
• Variable-length sequences processing	**YES (if one to one)**

Skoltech
Skolkovo Institute of Science and Technology

Convolutional Neural Networks

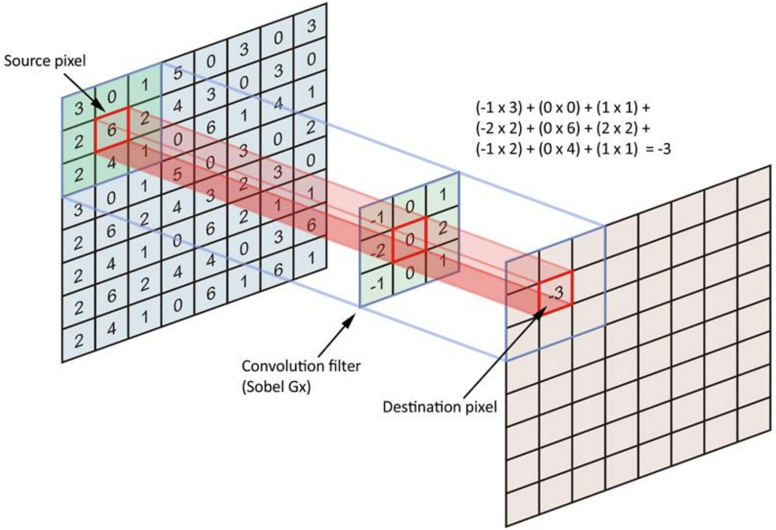
- Mathematical definition: combination of simple transformations
- There can be a lot of layers
- Convolutional layers help



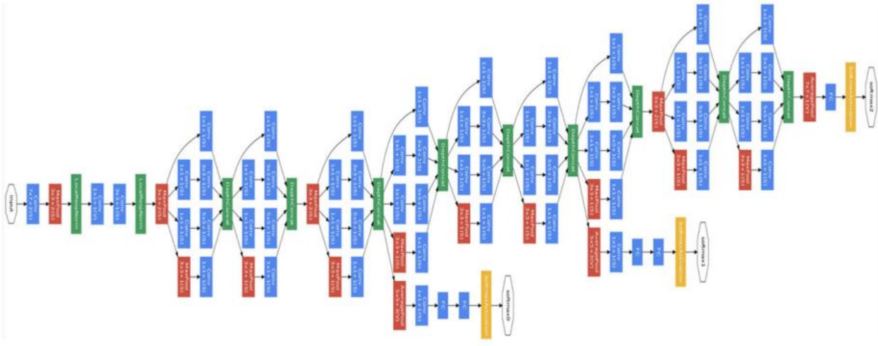
https://playground.tensorflow.org/

Convolutional Neural Networks

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



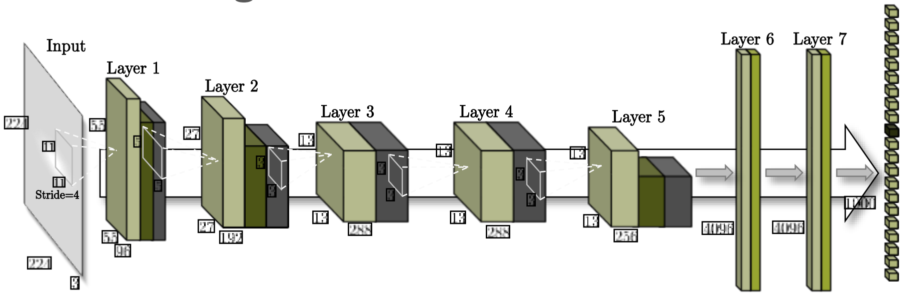
Single convolutional layer



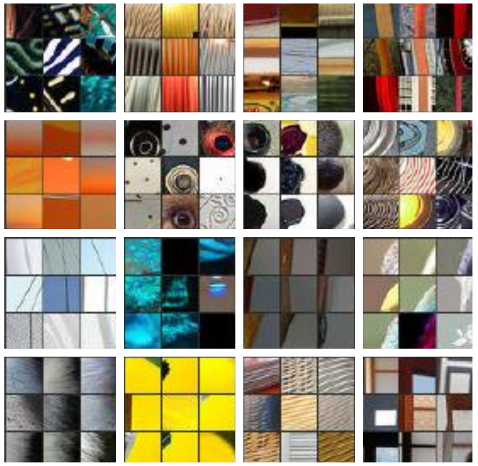
GoogLeNet architecture

Skoltech
Skolkovo Institute of Science and Technology

Flow of data through Convolutional Neural Network



Layer 1



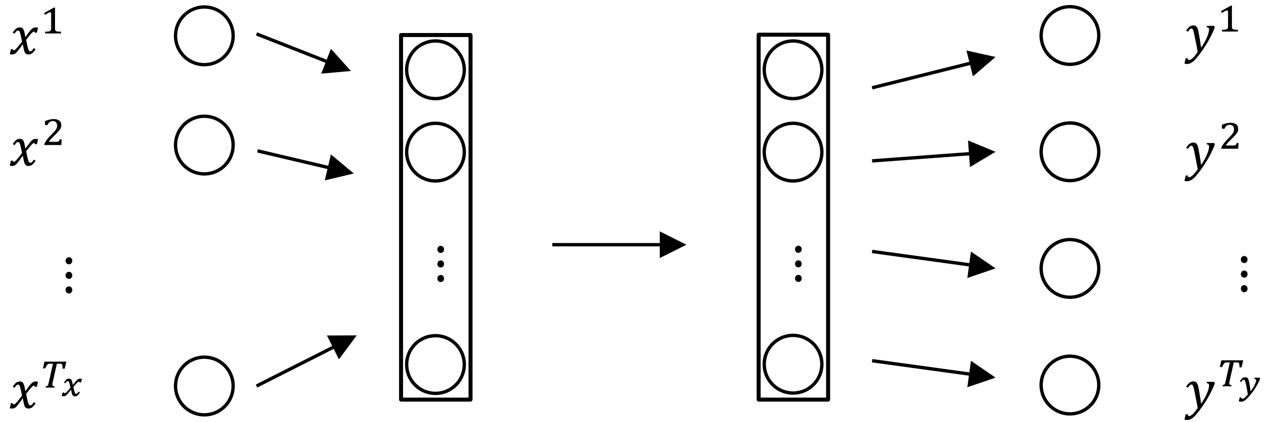
Layer 2

Layer 5

Zeiler, Matthew D., and Rob Fergus. "Visualizing and understanding convolutional networks." *European conference on computer vision*. Springer, Cham, 2014.

Skoltech
Skolkovo Institute of Science and Technology

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

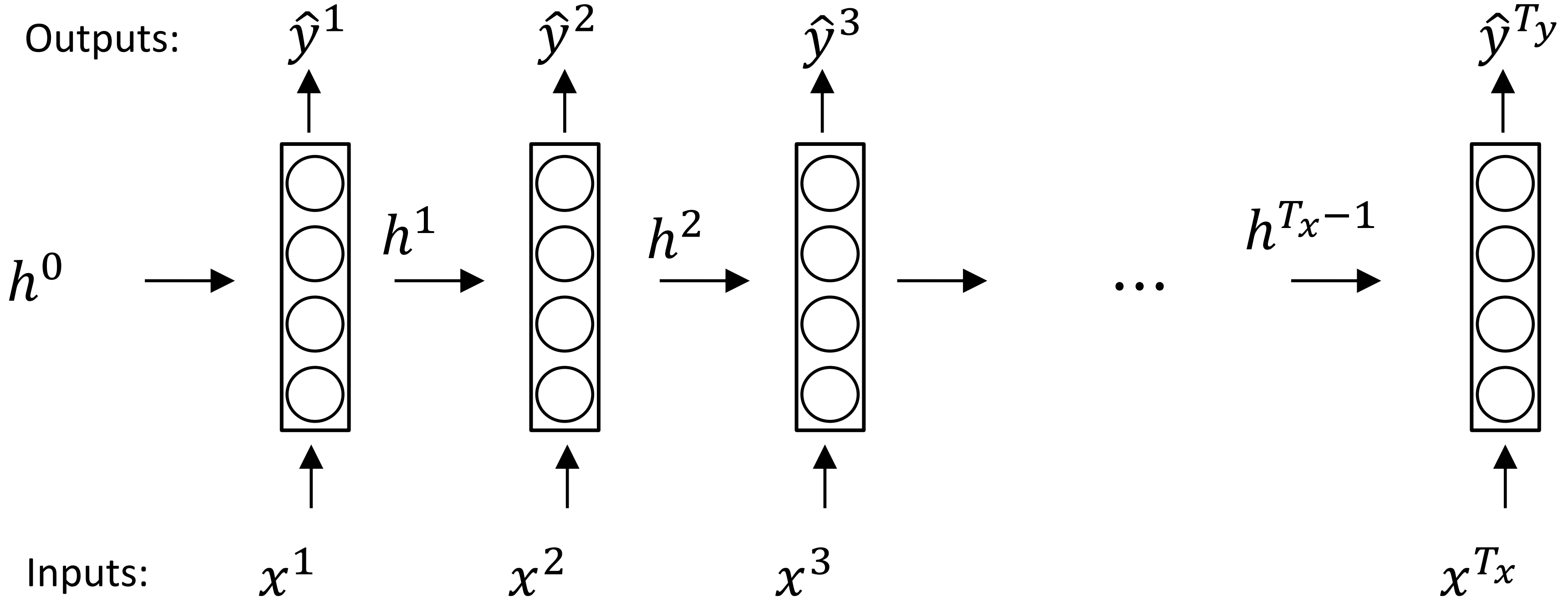
Adopted from Andrew Ng

Sequence processing with classic Fully connected neural networks

• Long-term memory	YES
• Maintain order information	YES
• Natural preprocessing	NO
• Variable-length sequences processing	NO

Skoltech
Skolkovo Institute of Science and Technology

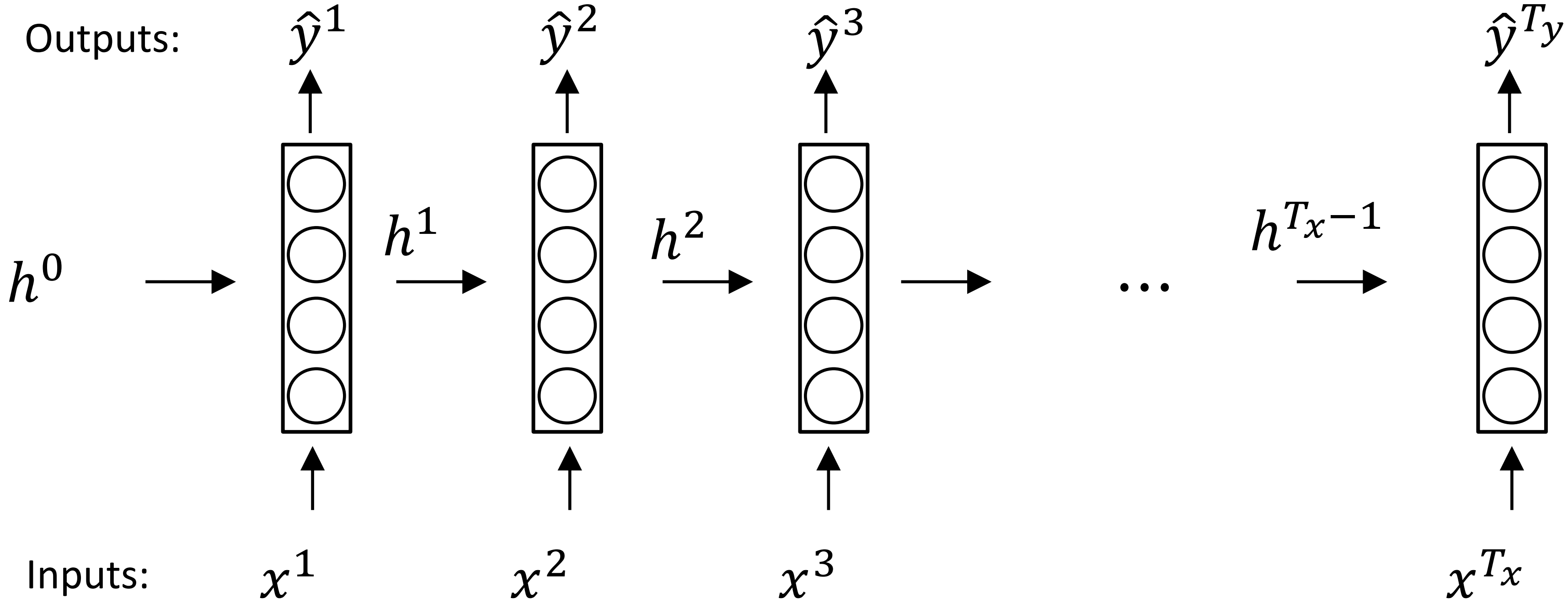
Forward propagation through Recurrent Neural Network



Skoltech
Skolkovo Institute of Science and Technology

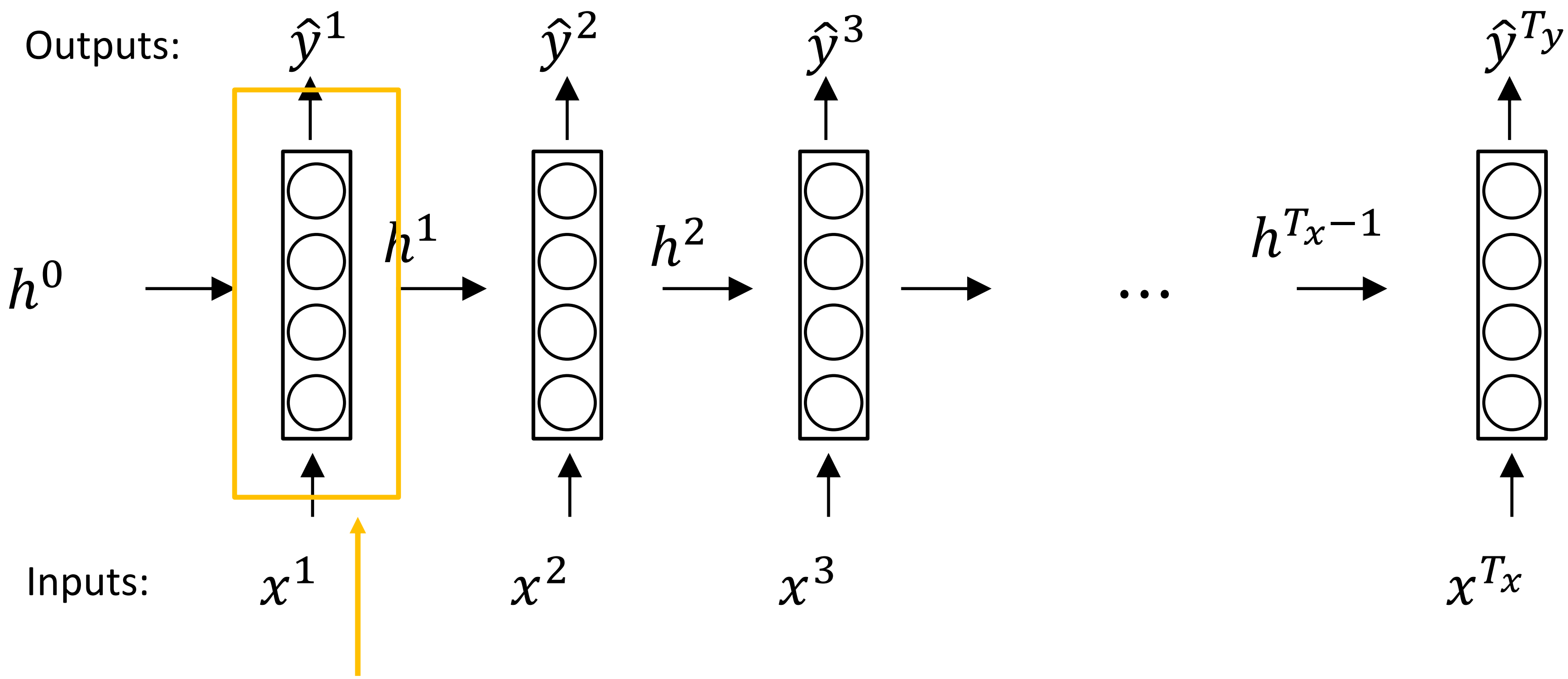
Adopted from Andrew Ng

Forward propagation through Recurrent Neural Network



Adopted from Andrew Ng

Forward propagation through Recurrent Neural Network

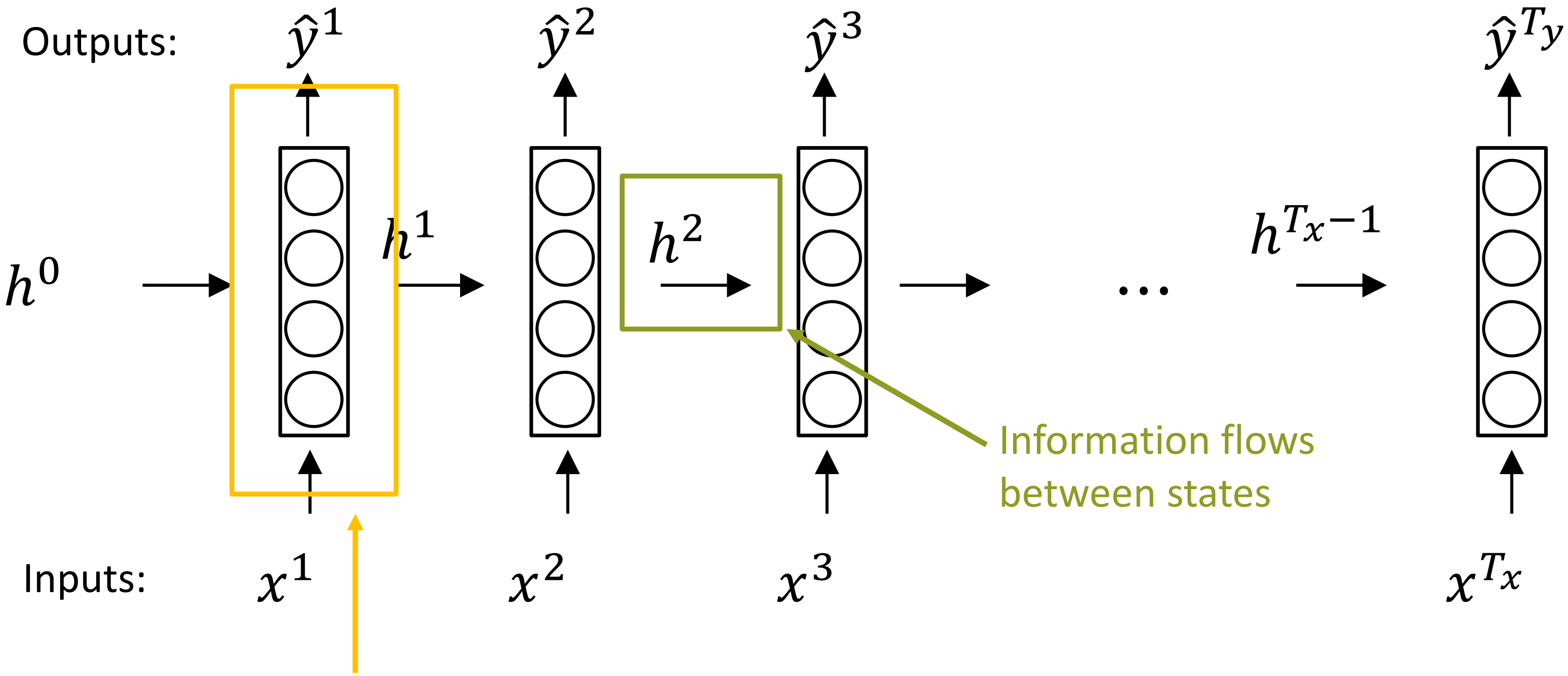


Processing unit is the same for all time moments.
Units has parameters we want to learn!

Adopted from Andrew Ng

Skoltech
Skolkovo Institute of Science and Technology

Forward propagation through Recurrent Neural Network

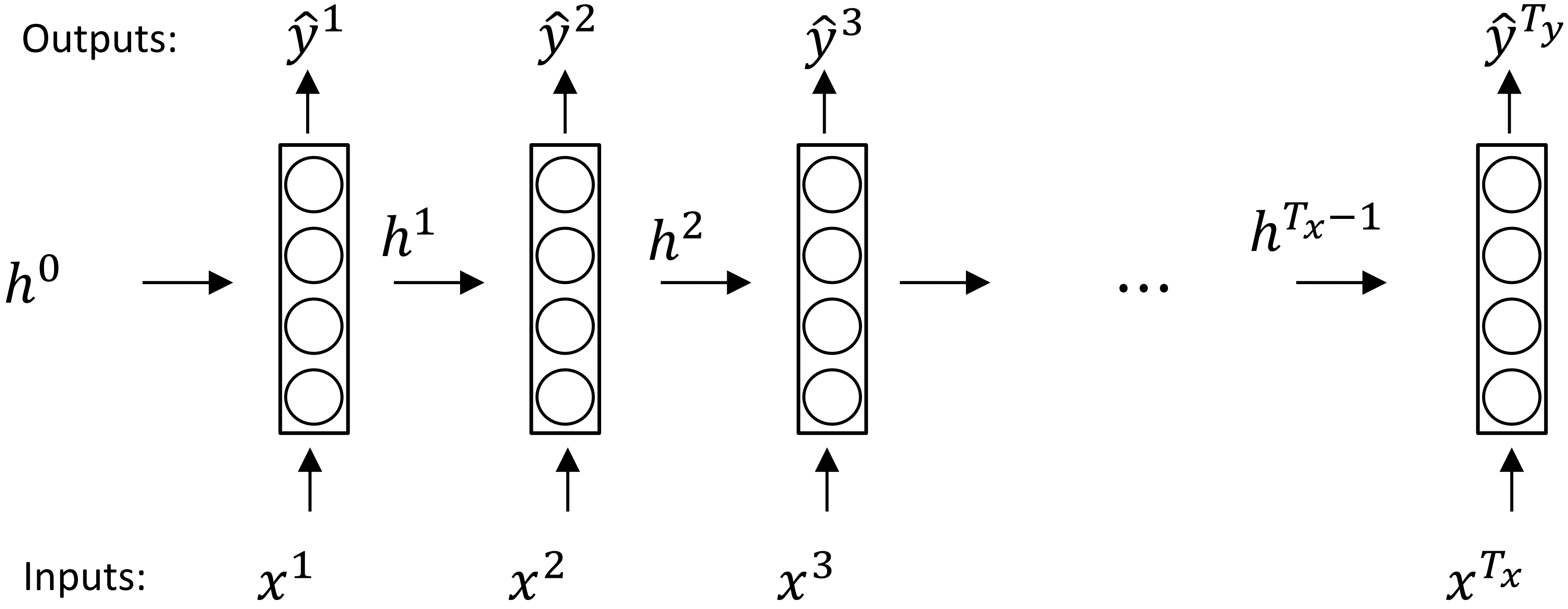


Processing unit is the same for all time moments.
Units has parameters we want to learn!

Adopted from Andrew Ng

Skoltech
Skolkovo Institute of Science and Technology

Forward propagation through Recurrent Neural Network

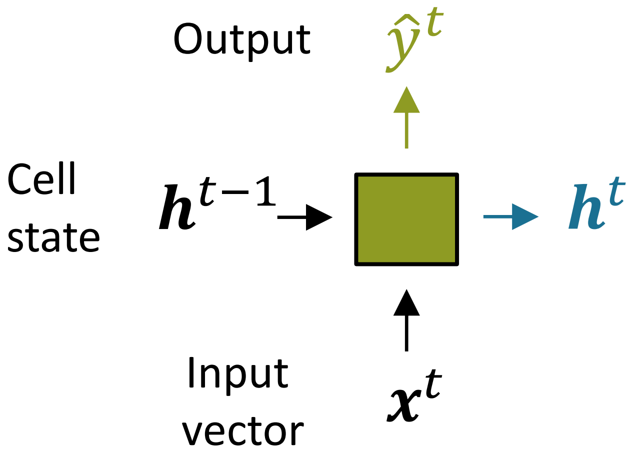


Input
For NLP
problems

It is truth universally …

Adopted from Andrew Ng

Simple RNN model



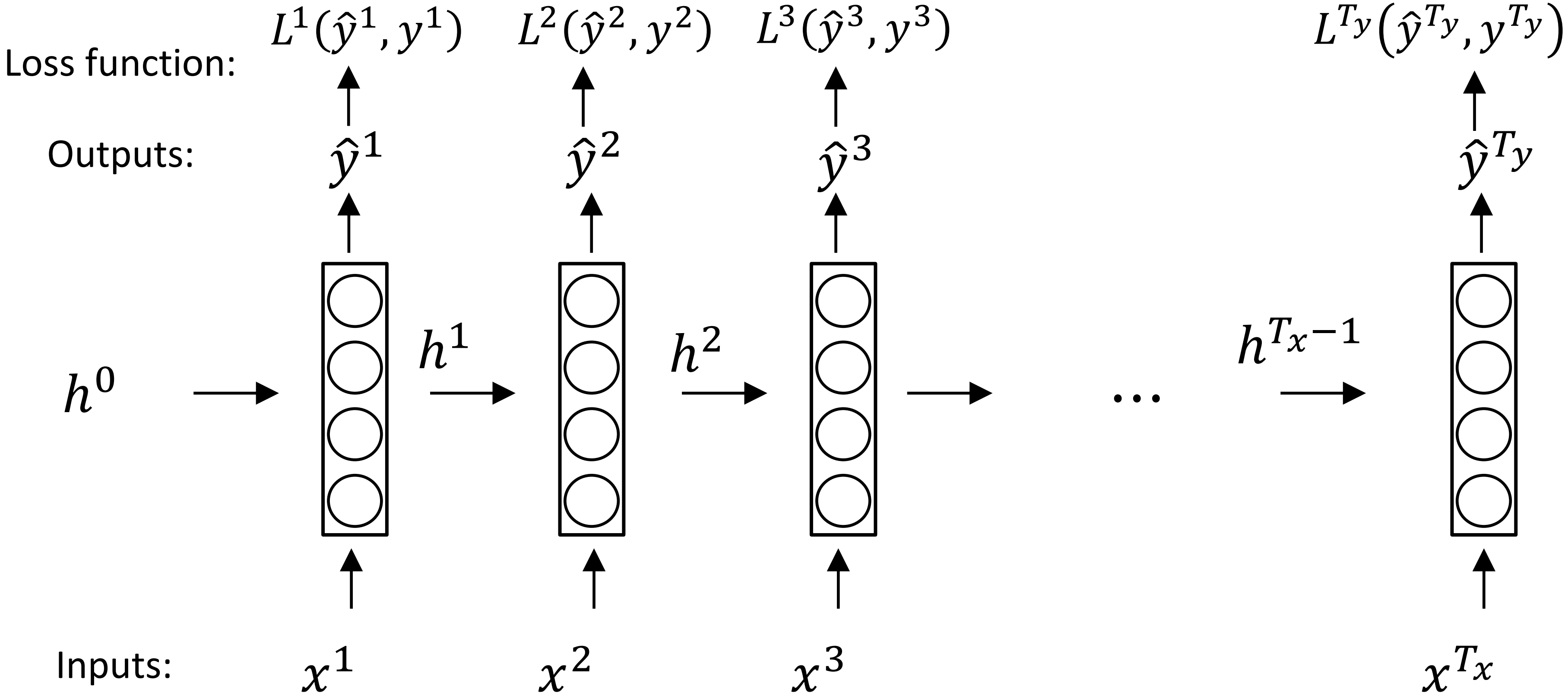
$$\boldsymbol{h}^t = f_h(\boldsymbol{x}^t, \boldsymbol{h}^{t-1})$$

$$\boldsymbol{h}^t = \tanh(V\boldsymbol{x}^t + W\boldsymbol{h}^{t-1} + b_h)$$

$$\hat{y}^t = f_y(\boldsymbol{h}^t)$$

$$\hat{y}^t = \text{softmax}(U\boldsymbol{h}^t + b_y)$$

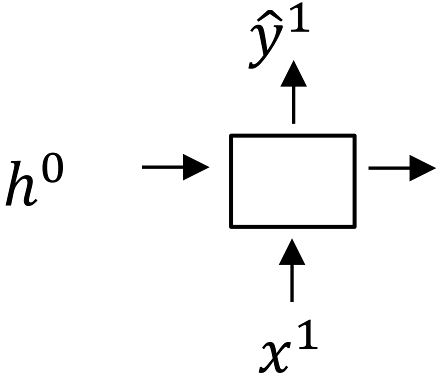
Backward propagation through Recurrent Neural Network



Adopted from Andrew Ng

Zoo of RNN (Recurrent Neural Network) models

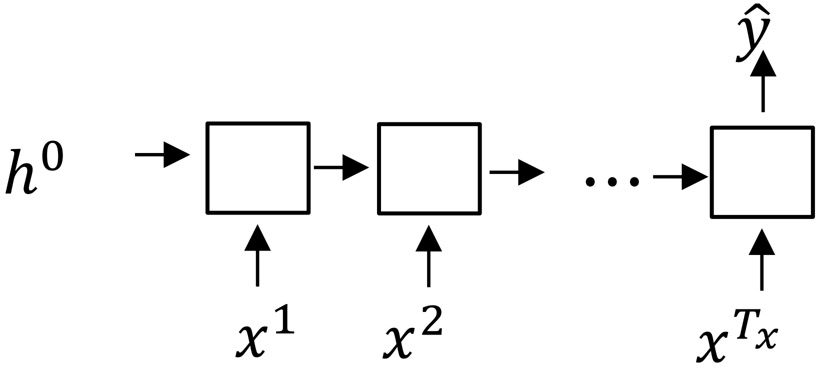
One to one



Separate input & output
each time

- Image classification from cameras

Many to one



Single output
for a sequence

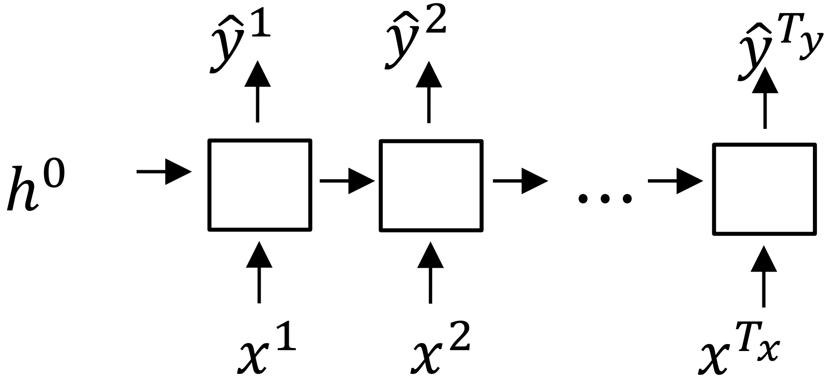
- Sentiment of a sentence: good or bad review?

Adopted from Andrew Ng

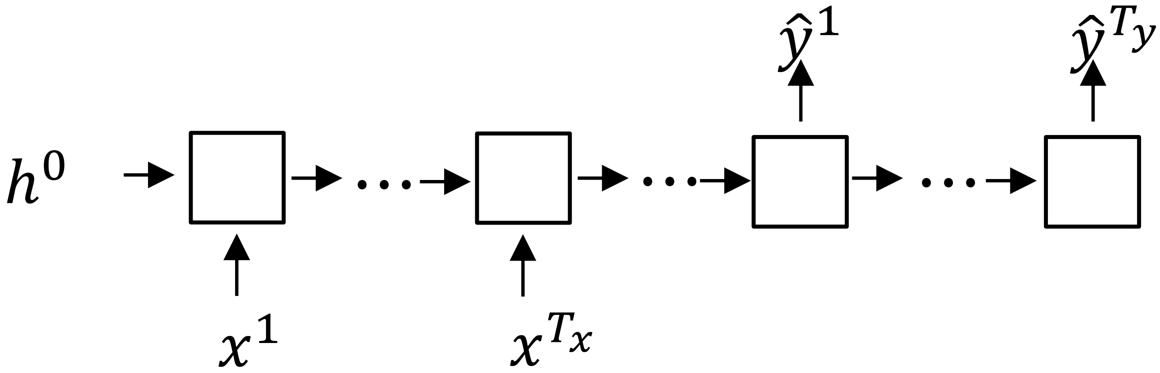
Skoltech
Skolkovo Institute of Science and Technology

Zoo of RNN (Recurrent Neural Network) models

Many to many



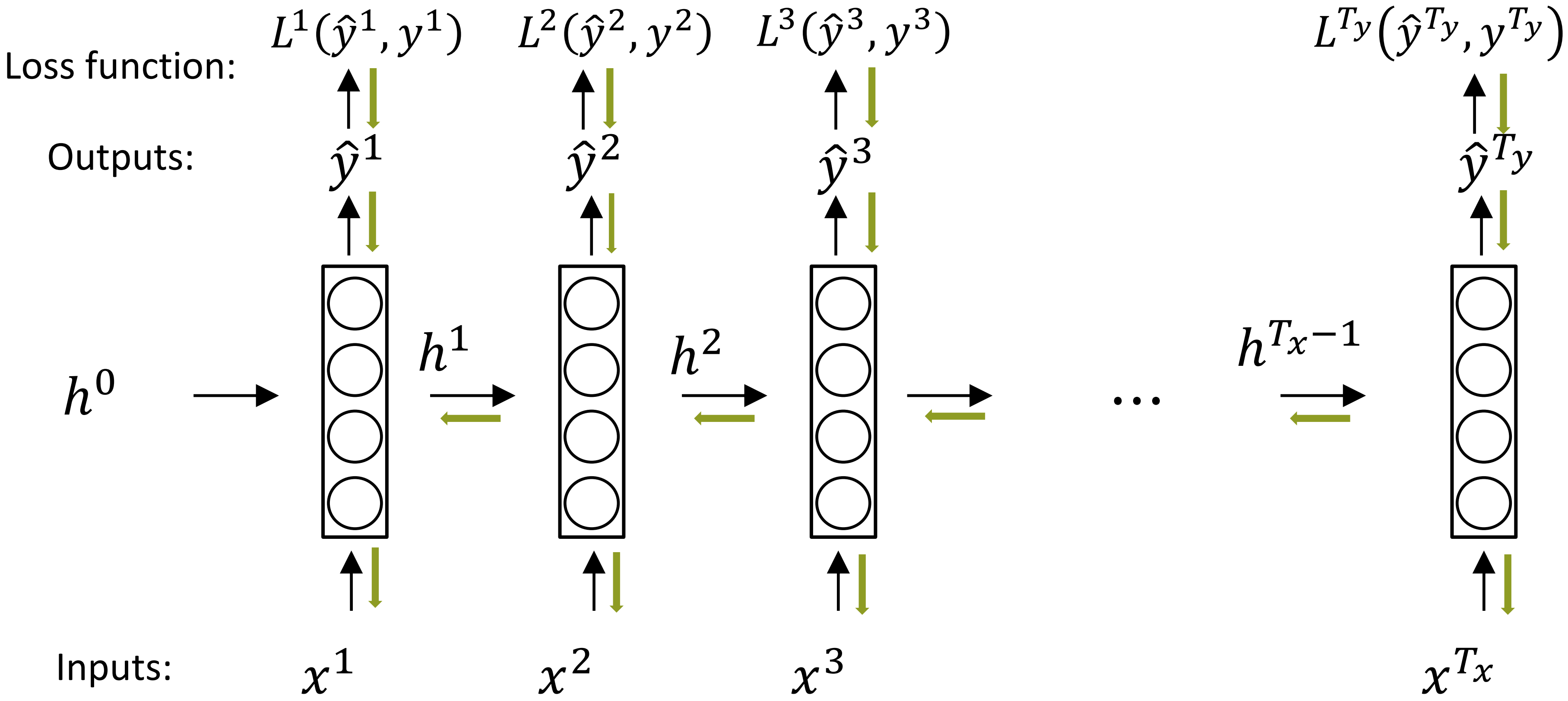
- Time series prediction
- Anomaly detection



- Translation

Adopted from Andrew Ng

Backward propagation through Recurrent Neural Network



Adopted from Andrew Ng

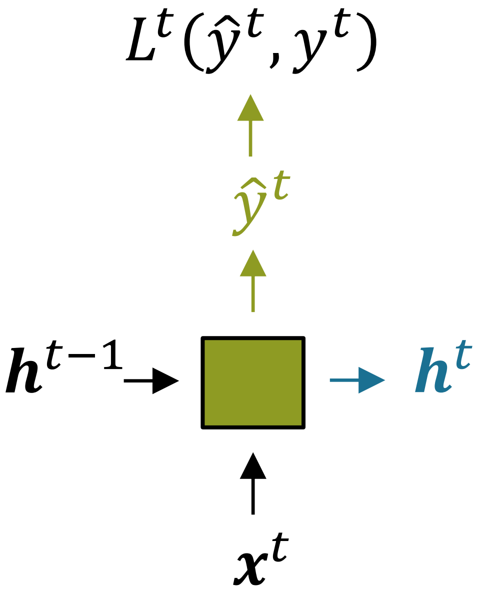
Skoltech
Skolkovo Institute of Science and Technology

Backpropagation w.r.t. U

$$L = \sum_{i=1}^{T_y} L^i(\hat{y}^i, y^i)$$

$$\frac{\partial L}{\partial U} = \sum_{i=1}^{T_y} \frac{\partial L_i}{\partial U} = \sum_{i=1}^{T_y} \frac{\partial L_i}{\partial \hat{y}^i} \frac{\partial \hat{y}^i}{\partial U}$$

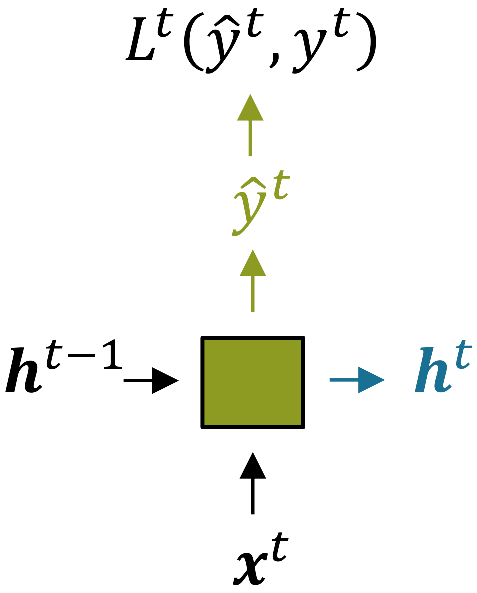
$$\hat{y}^t = \mathrm{softmax}(U\boldsymbol{h}^t + b_y)$$



Skoltech
Skolkovo Institute of Science and Technology

Backpropagation w.r.t. W

$$\hat{y}^t = \mathrm{softmax}(U\boldsymbol{h}^t + b_y)$$



$$L = \sum_{i=1}^{T_y} L^i(\hat{y}^i, y^i)$$

$$\frac{\partial L}{\partial W} = \sum_{i=1}^{T_y} \frac{\partial L_i}{\partial W} = \cancel{\sum_{i=1}^{T_y} \frac{\partial L_i}{\partial \hat{y}^i} \frac{\partial \hat{y}^i}{\partial W}}$$

$$\boldsymbol{h}^t = \tanh(V\boldsymbol{x}^t + \boxed{W\boldsymbol{h}^{t-1}} + b_h)$$

$$\frac{\partial L_i}{\partial \hat{y}^i}\frac{\partial \hat{y}^i}{\partial W} = \frac{\partial L_i}{\partial \hat{y}^i}\frac{\partial \hat{y}^i}{\partial h_t}\left(\frac{\partial h_t}{\partial W} + \frac{\partial h_t}{\partial h_{t-1}}\frac{\partial h_{t-1}}{\partial W} + \right)$$

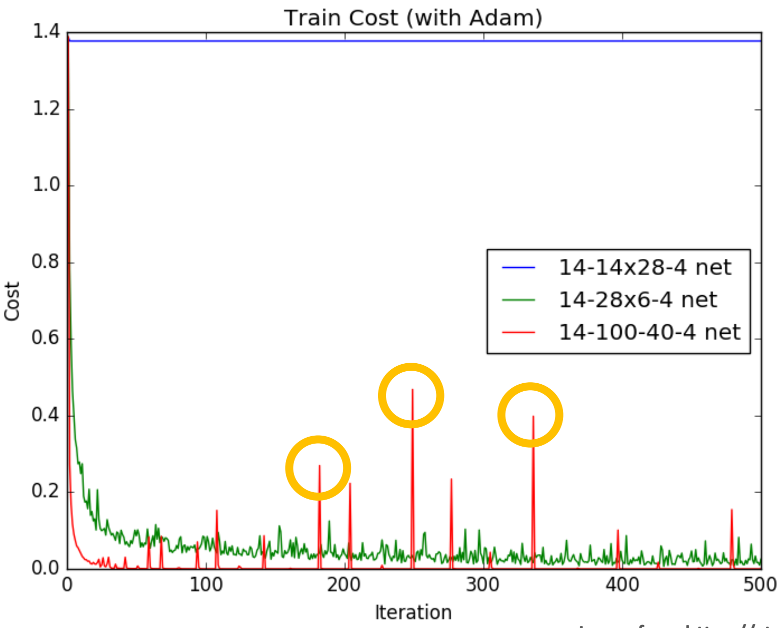
$$\frac{\partial L_i}{\partial \hat{y}^i}\frac{\partial \hat{y}^i}{\partial W} = \frac{\partial L_i}{\partial \hat{y}^i}\frac{\partial \hat{y}^i}{\partial h_t}\Sigma_{i=0}^{T_y}\left(\prod_{j=i+1}^{T_y}\frac{\partial h_j}{\partial h_{j-1}}\right)\frac{\partial h_i}{\partial W}$$

Skoltech
Skolkovo Institute of Science and Technology

Problems of classic RNN

Gradients explode

Many values > 1



Solution:

- **Gradient clipping to scale big gradients**

1. $g = \frac{\partial L}{\partial W}$
2. If $g > t$ for some threshold t:

$$g = \frac{t}{\|g\|} g$$

Threshold t is selected given the dynamic of loss function over iterations

Image from https://stats.stackexchange.com/questions/303857/explanation-of-spikes-in-training-loss-vs-iterations-with-adam-optimizer

Skoltech
Skolkovo Institute of Science and Technology

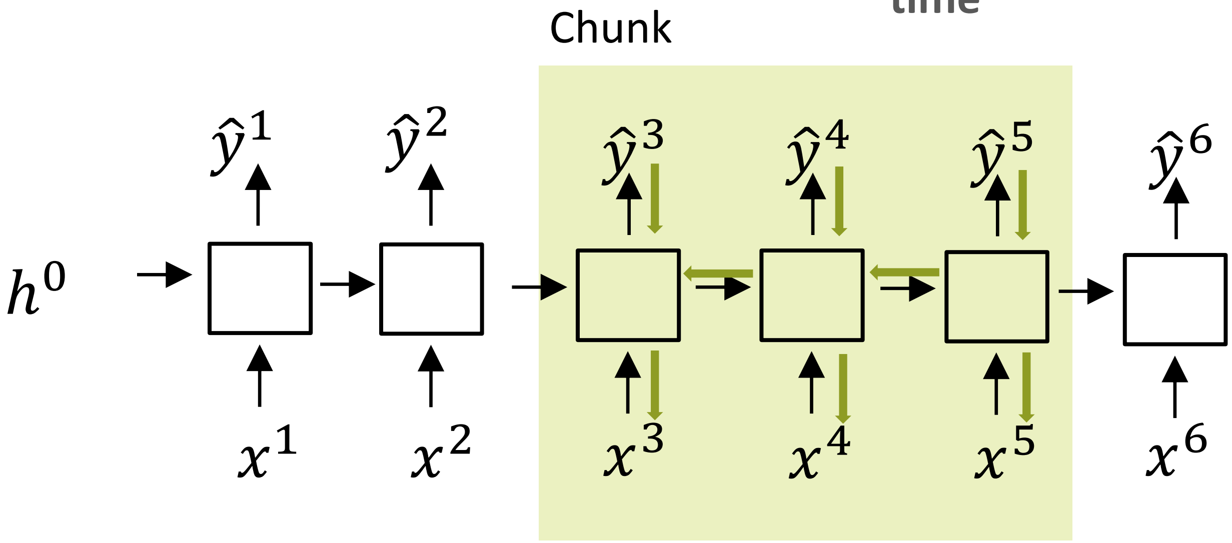
Problems of classic RNN

Gradients explode

Many values > 1

Solution:

- **Gradient clipping to scale big gradients**
- **Truncated backpropagation through time**



Skoltech
Skolkovo Institute of Science and Technology

Problems of classic RNN

Gradients vanish is a more

Many values < 1

Bias parameters to capture long-term dependencies

Hard to detect!

Tricks:

Activation functions

- Use ReLU

Parameter initialization

- Initialize weights to identity matrix
- Initialize biases to zero

Another big problem of classic RNN

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

Solution: Gated architectures

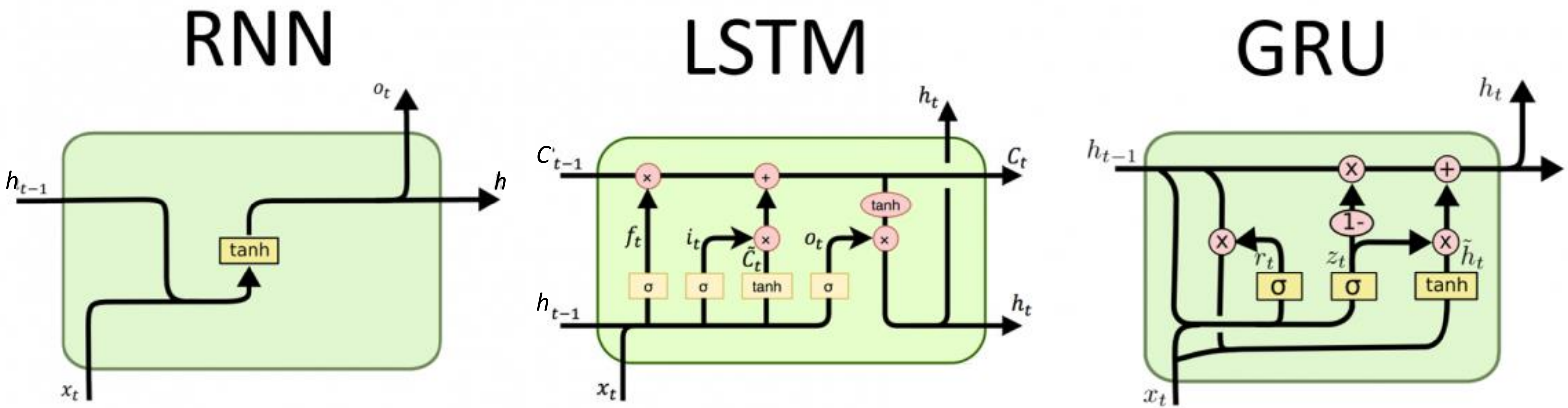
More complex recurrent units with gates to control what information is passed through

- GRU (Gated Recurrent Unit)
- LSTM (Long-Short Term Memory)

Selection of RNN architecture

Better RNN units: LSTM and GRU

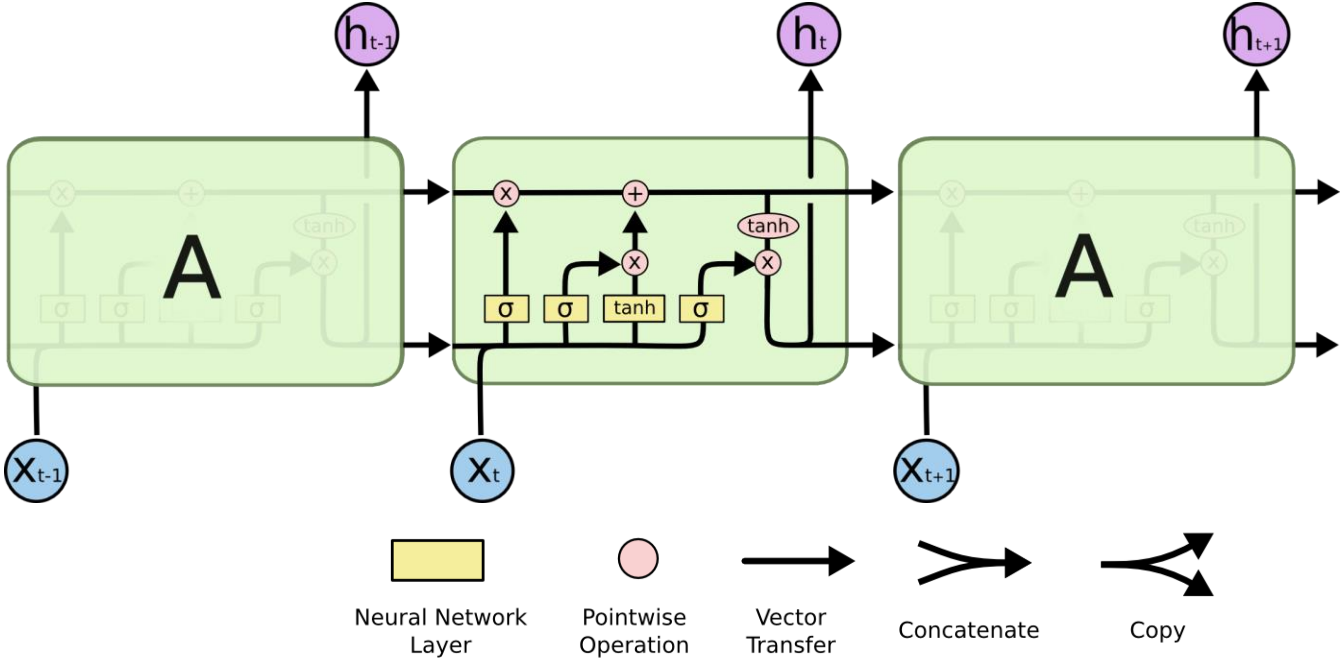
- LSTM: long short term memory
- GRU: Gated recurrent unit



Details on how LSTM works

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

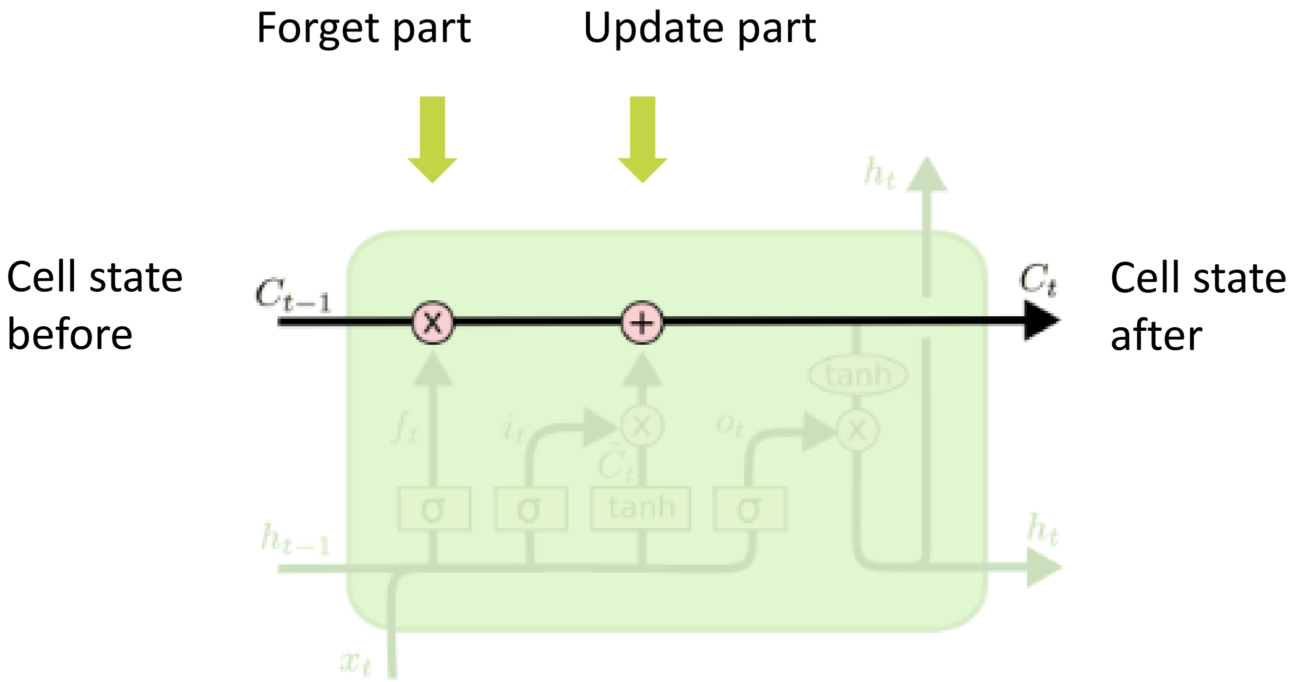
LSTM was proposed by J. Schmidhuber group in 1991



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

Skoltech
Skolkovo Institute of Science and Technology

Long term memory part – Cell state



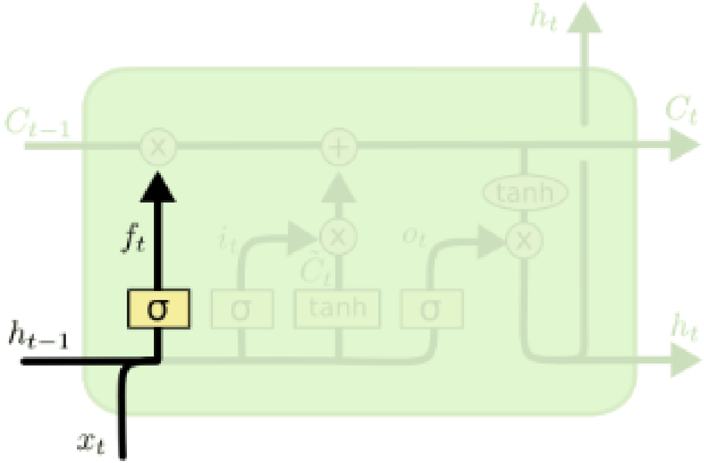
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Skoltech
Skolkovo Institute of Science and Technology

Forget part

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

hidden state (a)



inputs

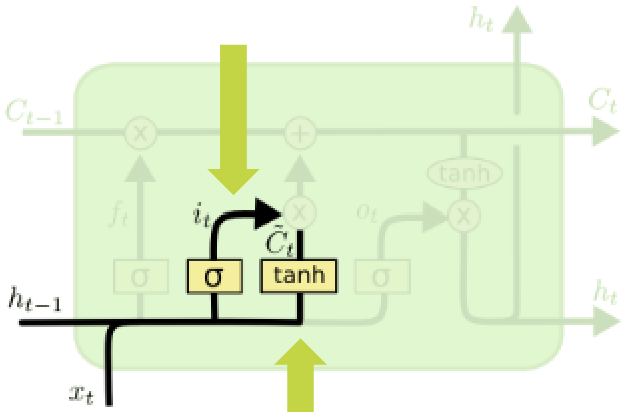
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right)$$

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

Update part

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

hidden
state (a)



inputs

What update we do

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

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

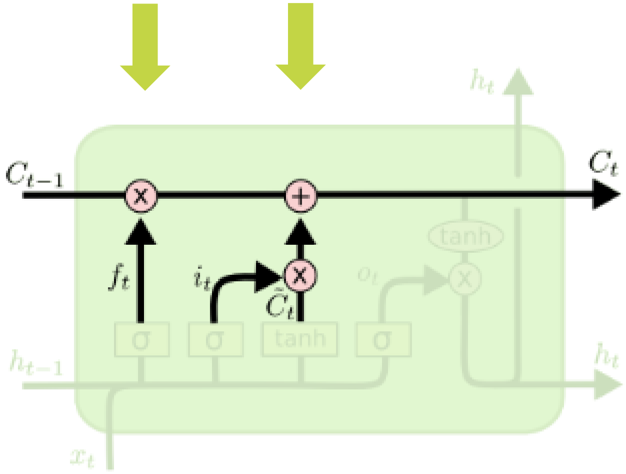
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Skoltech
Skolkovo Institute of Science and Technology

Long term memory part – Cell state

Forget part   Update part

Cell state before



Cell state after

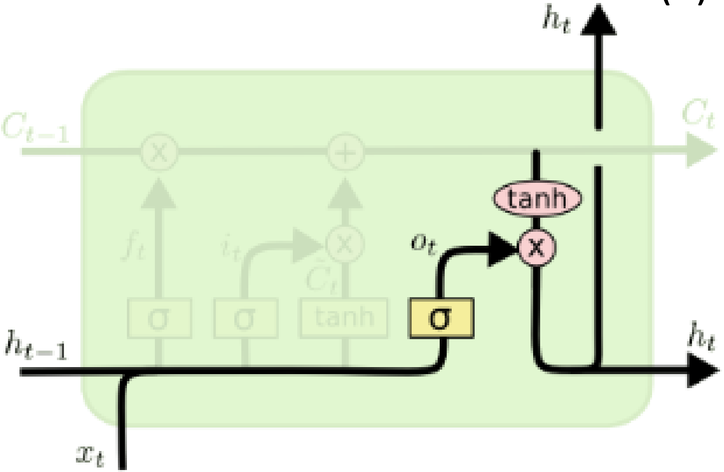
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

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

Skoltech
Skolkovo Institute of Science and Technology

Update everything else

hidden
state (a)



hidden
state (a)

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

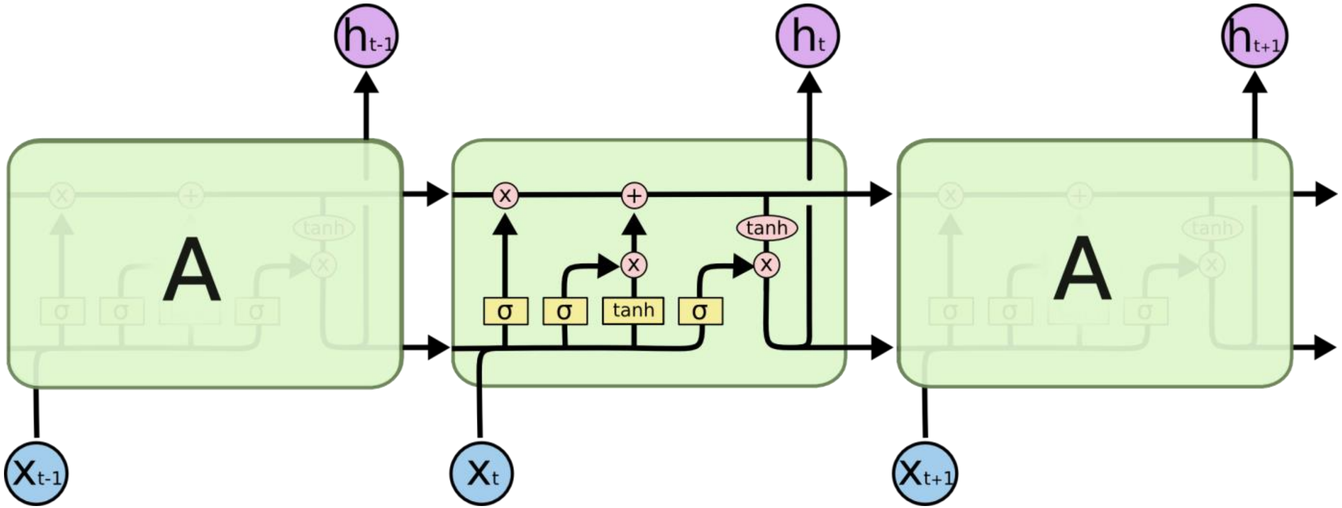
$$h_t = o_t * \tanh\left(C_t\right)$$

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

Skoltech
Skolkovo Institute of Science and Technology

Details on how LSTM works

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

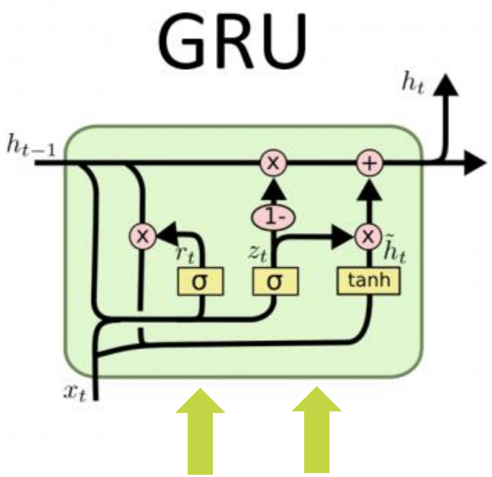


http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Skoltech
Skolkovo Institute of Science and Technology

GRU – Gated Recurrent Unit

- Update gate – what to pay attention to
- Reset gate – what to forget



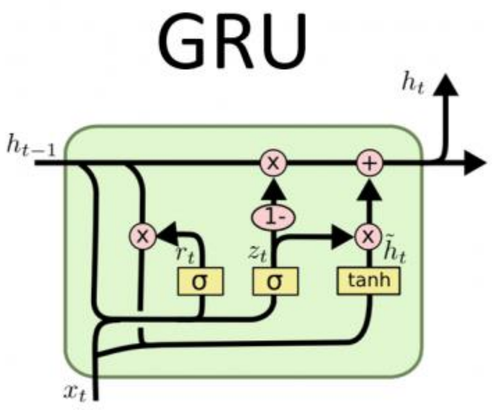
$$\boldsymbol{r}^t = \sigma(W_{xr}\boldsymbol{x}^t + W_{hr}\boldsymbol{h}^{t-1} + b_r)$$
$$\boldsymbol{z}^t = \sigma(W_{xz}\boldsymbol{x}^t + W_{hz}\boldsymbol{h}^{t-1} + b_z)$$

$$\widetilde{\boldsymbol{h}}^t = \tanh(W_{xh}\boldsymbol{x}^t + W_{hr}(\boldsymbol{r}^t \odot \boldsymbol{h}^{t-1}) + b_h)$$
$$\boldsymbol{h}^t = \boldsymbol{z}^t \odot \boldsymbol{h}^{t-1} + (1 - \boldsymbol{z}^t) \odot \widetilde{\boldsymbol{h}}^t$$

Cho, Kyunghyun et al. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation.

GRU – Gated Recurrent Unit

- Update gate – what to pay attention to
- Reset gate – what to forget



- Slightly worse than LSTM
- Simpler and cheaper than LSTM

Cho, Kyunghyun et al. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation.

Towards a better recurrent block

- LSTM architecture is ad-hoc and has a substantial number of components whose purpose is not immediately apparent
- Like the LSTM, it is hard to tell, at a glance, which part of the GRU is essential for its functioning.

- Let’s compare 10 000 different architectures on 3 problems with 1 000 of them pass the initial filtering stage: genetic algorithm
- Each architecture has been evaluated on about 220 hyperparameter settings.
- 230 000 hyperparameter configurations in total!

Jozefowicz, R., Zaremba, W., & Sutskever, I. 2015. An empirical exploration of recurrent network architectures. ICML

Best found architectures MUTx are close to GRU

GRU:

$$
\begin{aligned}
r_t &= \mathrm{sigm}\left(W_{\mathrm{xr}} x_t + W_{\mathrm{hr}} h_{t-1} + b_{\mathrm{r}}\right) \\
z_t &= \mathrm{sigm}(W_{\mathrm{xz}} x_t + W_{\mathrm{hz}} h_{t-1} + b_{\mathrm{z}}) \\
\tilde{h}_t &= \tanh(W_{\mathrm{xh}} x_t + W_{\mathrm{hh}}(r_t \odot h_{t-1}) + b_{\mathrm{h}}) \\
h_t &= z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t
\end{aligned}
$$

Arch.	N	N-dropout	P
Tanh	3.612	3.267	6.809
LSTM	3.492	3.403	6.866
LSTM-f	3.732	3.420	6.813
LSTM-i	3.426	**3.252**	6.856
LSTM-o	3.406	**3.253**	6.870
LSTM-b	3.419	3.345	6.820
GRU	3.410	3.427	6.876
MUT1	**3.254**	3.376	**6.792**
MUT2	3.372	3.429	6.852
MUT3	3.337	3.505	6.840

Table 2. Negative Log Likelihood on the music datasets. N stands for Nottingham, N-dropout stands for Nottingham with nonzero dropout, and P stands for Piano-Midi.

MUT1:

$$
\begin{aligned}
z &= \mathrm{sigm}(W_{\mathrm{xz}} x_t + b_{\mathrm{z}}) \\
r &= \mathrm{sigm}(W_{\mathrm{xr}} x_t + W_{\mathrm{hr}} h_t + b_{\mathrm{r}}) \\
h_{t+1} &= \tanh(W_{\mathrm{hh}}(r \odot h_t) + \tanh(x_t) + b_{\mathrm{h}}) \odot z \\
&+ h_t \odot (1 - z)
\end{aligned}
$$

MUT2:

$$
\begin{aligned}
z &= \mathrm{sigm}(W_{\mathrm{xz}} x_t + W_{\mathrm{hz}} h_t + b_{\mathrm{z}}) \\
r &= \mathrm{sigm}(x_t + W_{\mathrm{hr}} h_t + b_{\mathrm{r}}) \\
h_{t+1} &= \tanh(W_{\mathrm{hh}}(r \odot h_t) + W_{xh} x_t + b_{\mathrm{h}}) \odot z \\
&+ h_t \odot (1 - z)
\end{aligned}
$$

MUT3:

$$
\begin{aligned}
z &= \mathrm{sigm}(W_{\mathrm{xz}} x_t + W_{\mathrm{hz}} \tanh(h_t) + b_{\mathrm{z}}) \\
r &= \mathrm{sigm}(W_{\mathrm{xr}} x_t + W_{\mathrm{hr}} h_t + b_{\mathrm{r}}) \\
h_{t+1} &= \tanh(W_{\mathrm{hh}}(r \odot h_t) + W_{xh} x_t + b_{\mathrm{h}}) \odot z \\
&+ h_t \odot (1 - z)
\end{aligned}
$$

Jozefowicz, R., Zaremba, W., & Sutskever, I. 2015. An empirical exploration of recurrent network architectures. ICML

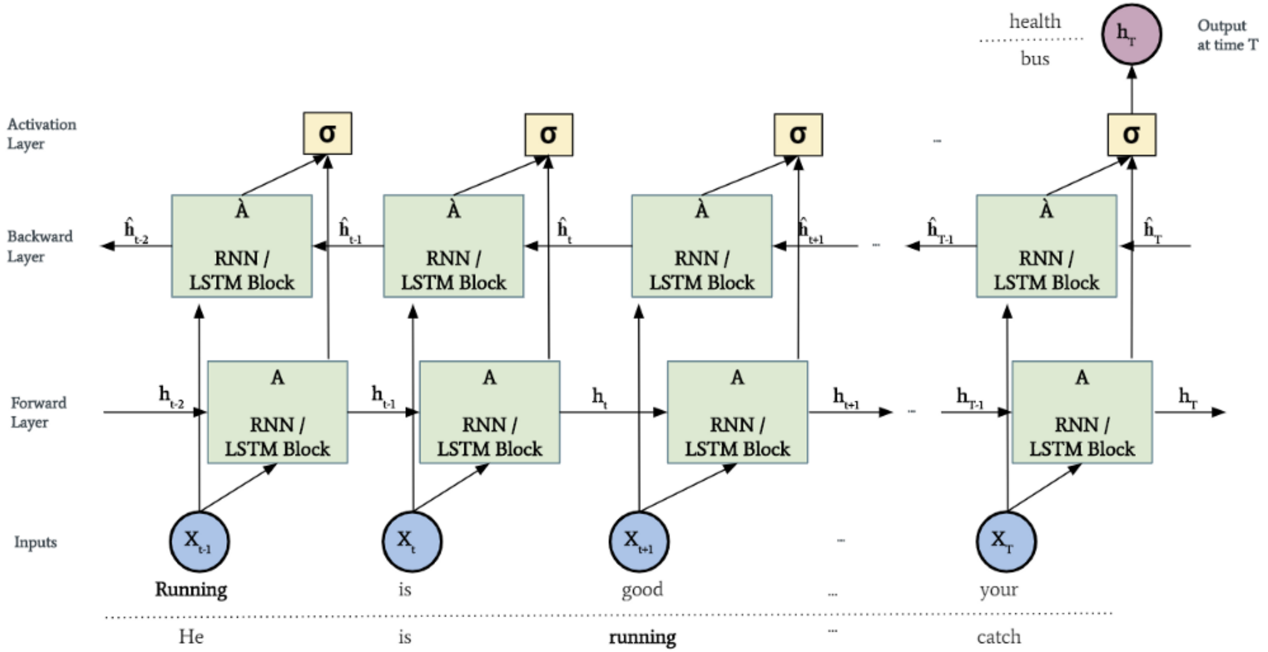
Towards a better recurrent block

- LSTM architecture is ad-hoc and has a substantial number of components whose purpose is not immediately apparent
- Like the LSTM, it is hard to tell, at a glance, which part of the GRU is essential for its functioning.

- Let's compare 8 LSTM variants and hope for the best by search over the space of hyperparameters with 5400 runs in total
- No significant improvement over common LSTM
- Some advices on hyperparameters selection

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. 2016. LSTM: A search space odyssey. IEEE transactions on neural networks and learning systems

Other architectures: bidirectional LSTM

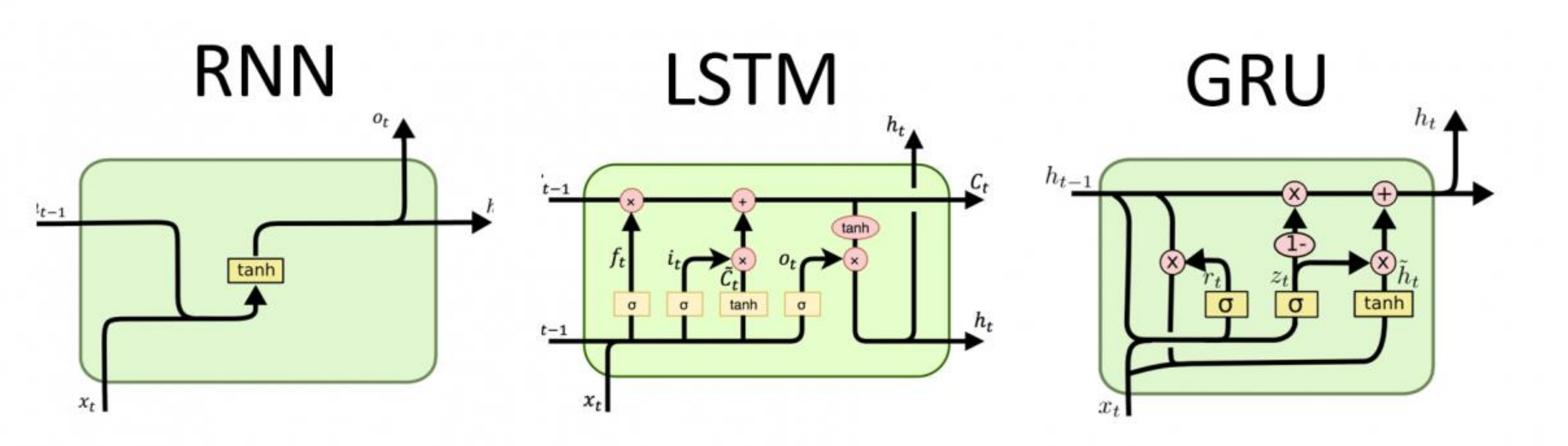


https://mc.ai/review-generation-using-bidirectional-long-short-term-memorylstm/

Other architectures: bidirectional LSTM

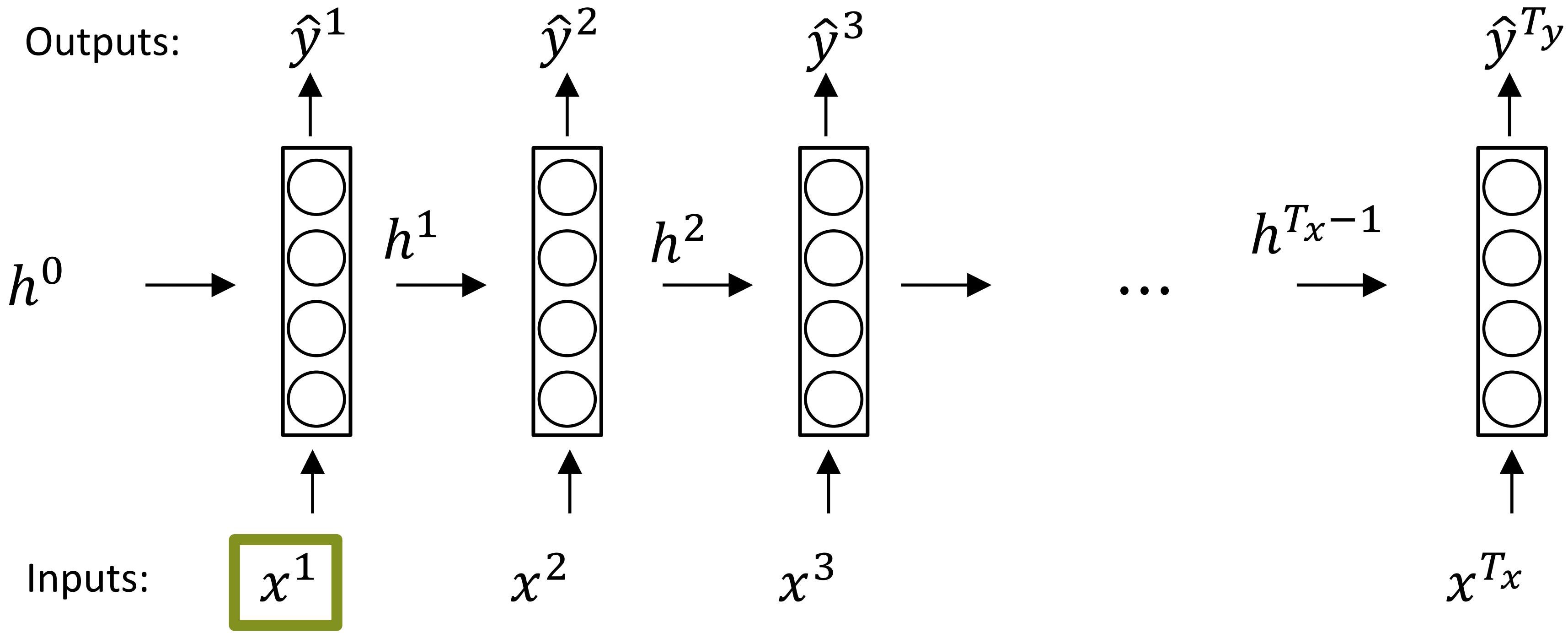
Bidirectional LSTM are useful when we benefit from the future data:

- Handwriting Recognition
- Speech Recognition
- protein Structure Prediction (bioinformatics)



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

Representation learning is still here



Most of the time we also learn representations of objects in end2end manner

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

Sources

- Recurrent Neural Networks | in Deep learning course by MIT 6.S191
- Coursera course on Sequence models https://www.coursera.org/learn/nlp-sequence-models
- http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Backslides

Sequence processing with classic Fully connected neural networks

- Long-term memory
- Maintain order information
- Natural preprocessing
- Variable-length sequences processing

GOTO: sli.do/seq_1
and answer the question

Forward propagation through Recurrent Neural Network

Initial sequence:

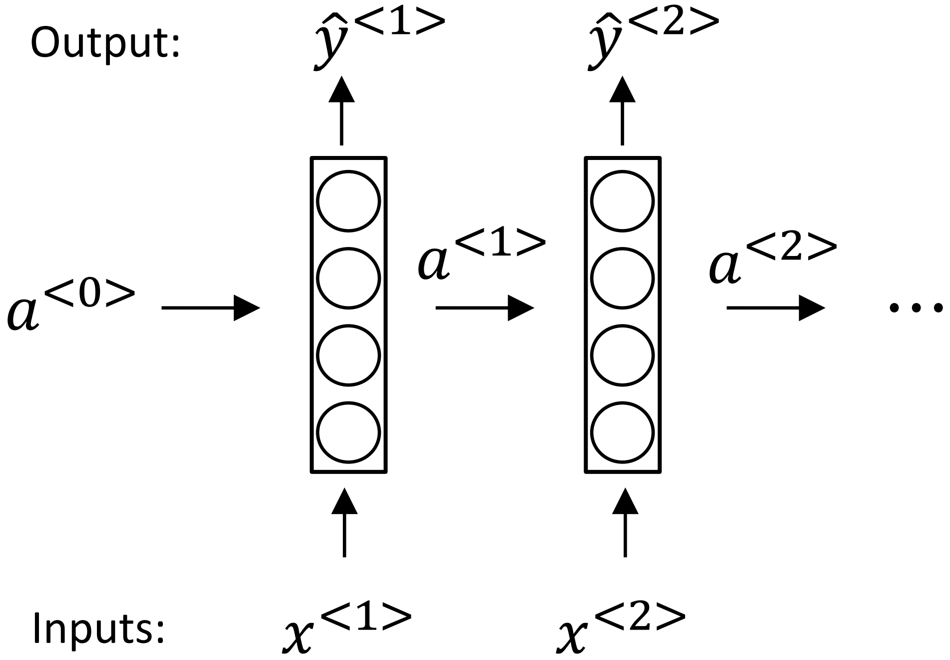
$x^{<1:n>} = x^{<1>}, x^{<2>}, \ldots, x^{<T>}, x_i \in \mathbb{R}^{d_{in}}$

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

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

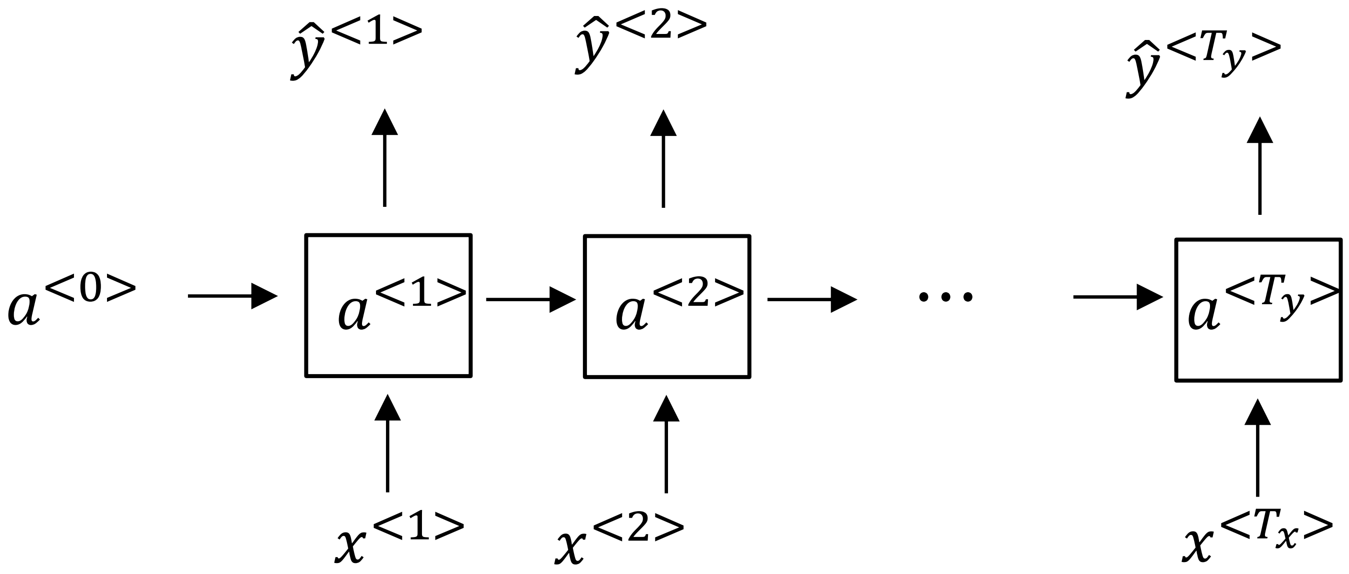
For the whole sequence $x^{<1:n>}$:

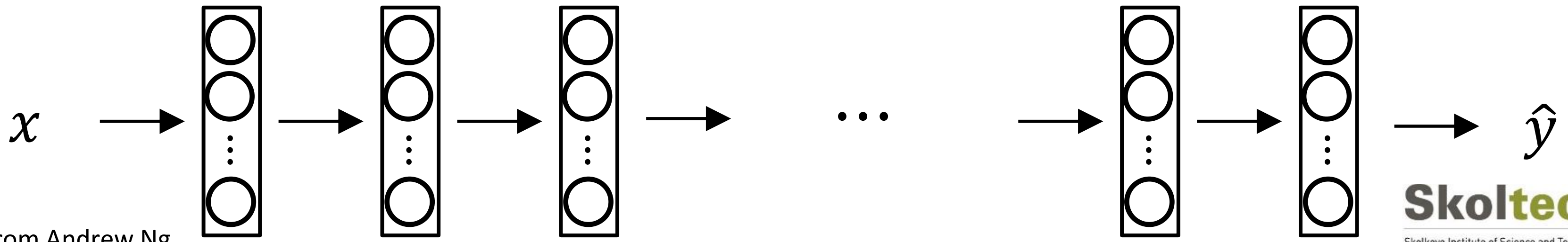
$y^{<1:n>} = RNN^*(x^{<1:n>}), y^{<i>} \in \mathbb{R}^{d_{out}}$



Adopted from Andrew Ng

Gradient vanish and/or explode

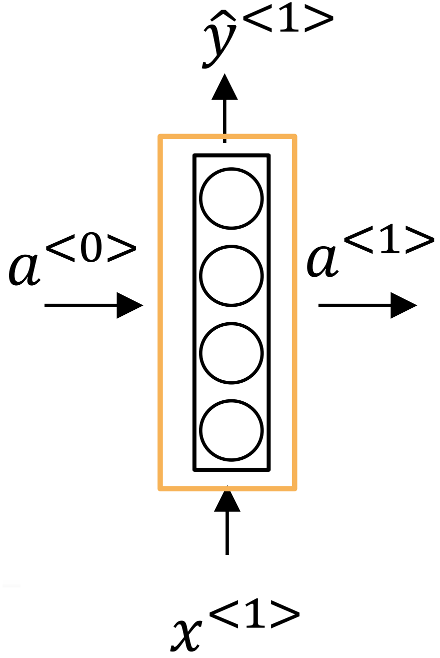




Adopted from Andrew Ng

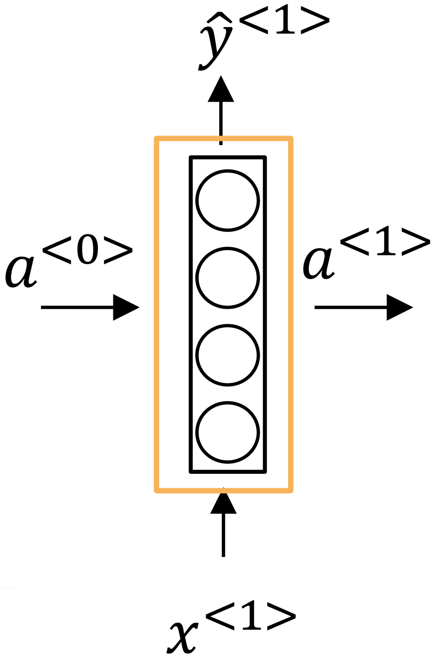
Simplest RNN unit: what is going on inside yellow rectangle?

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



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 $W^{hid} \in \mathbb{R}^{(d_{in}+d_{out}) \times d_{hid}}$, $W^{out} \in \mathbb{R}^{d_{hid} \times d_{out}}$