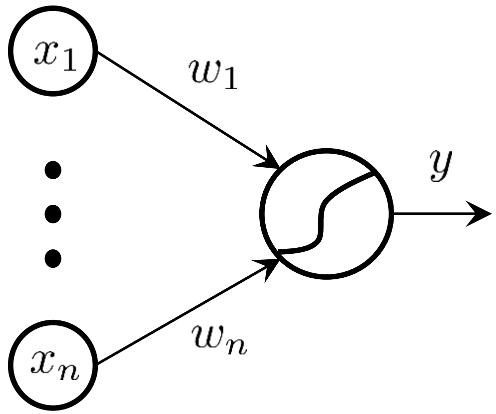
RNN and LSTM

Ekaterina Lobacheva, Dmitry Vetrov

lobacheva.tjulja@gmail.com, vetrovd@yandex.ru

Moscow
2015

Artificial neuron

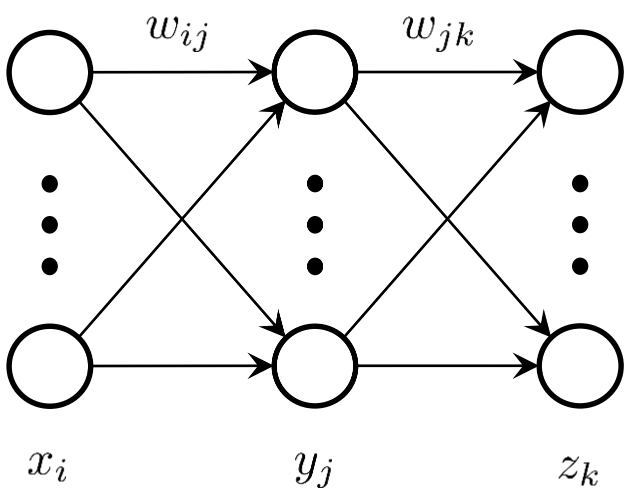


$$net = \sum_{i=1}^{n} w_i x_i$$

$$y = f(net)$$

nonlinear function

Artificial neural network



Training samples:

$$(\bar{x}(t), a(t))_{t=1}^{T}$$

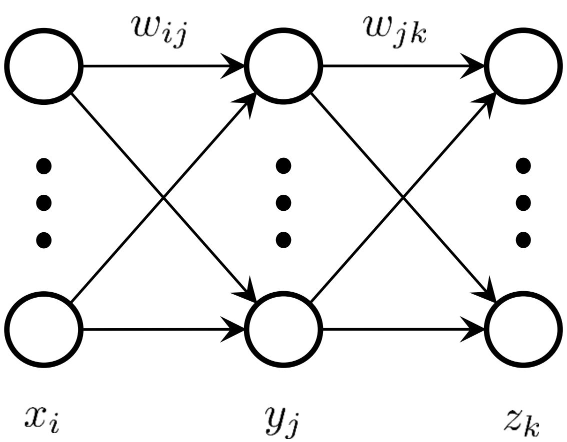
Parameters:

$$\theta = \{w_{ij}, w_{jk}\} \quad \text{for all} \quad i, j, k$$

Loss function:

$$F(Z(\theta), A) = \sum_{t=1}^{T} F_t(\bar{z}(t, \theta), a(t)) \underset{\theta}{\rightarrow} min$$

Backpropagation



$$net_j = \sum_i w_{ij} x_i \qquad y_j = f(net_j)$$

$$net_k = \sum_j w_{jk} y_j \qquad z_k = f(net_k)$$

Gradient descent method (usually stochastic)

Forward pass:

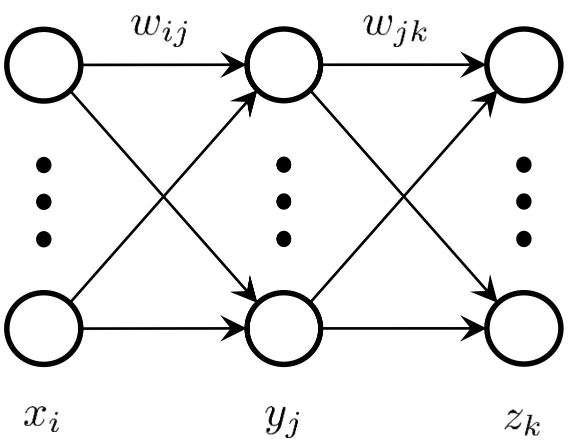
$$net_j(t), y_j(t), net_k(t), z_k(t)$$

Backward pass:

$$\frac{\partial F}{\partial w_{ij}}, \frac{\partial F}{\partial w_{jk}}$$

Backpropagation:

gradients



$$\frac{\partial F}{\partial z_k(t)} = \frac{\partial F_t}{\partial z_k(t)}$$

$$\frac{\partial F}{\partial net_k(t)} = \frac{\partial F_t}{\partial z_k(t)} \frac{\partial z_k(t)}{\partial net_k(t)} = \frac{\partial F_t}{\partial z_k(t)} f'(net_k(t))$$

$$\frac{\partial F}{\partial w_{jk}} = \sum_{t=1}^{T} \frac{\partial F_t}{\partial net_k(t)} \frac{\partial net_k(t)}{\partial w_{jk}} = \sum_{t=1}^{T} \frac{\partial F_t}{\partial net_k(t)} y_j(t)$$

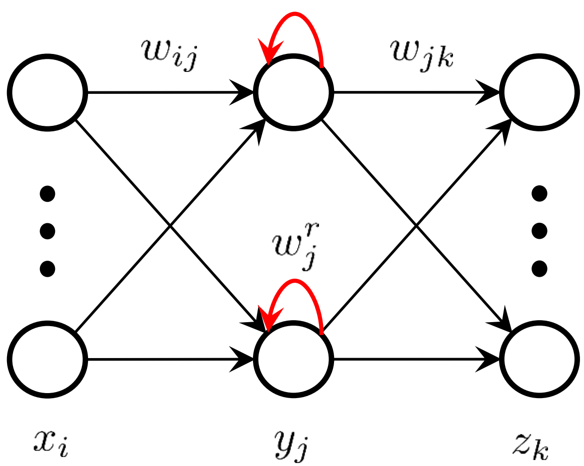
$$\frac{\partial F}{\partial y_j(t)} = \sum_{k} \frac{\partial F_t}{\partial net_k(t)} \frac{\partial net_k(t)}{\partial y_j(t)} = \sum_{k} \frac{\partial F_t}{\partial net_k(t)} w_{jk}$$

$$\frac{\partial F}{\partial net_j(t)} = \frac{\partial F_t}{\partial y_j(t)} f'(net_j(t))$$

$$\frac{\partial F}{\partial w_{ij}} = \sum_{t=1}^{T} \frac{\partial F_t}{\partial net_j(t)} \frac{\partial net_j(t)}{\partial w_{ij}} = \sum_{t=1}^{T} \frac{\partial F_t}{\partial net_j(t)} x_i(t)$$

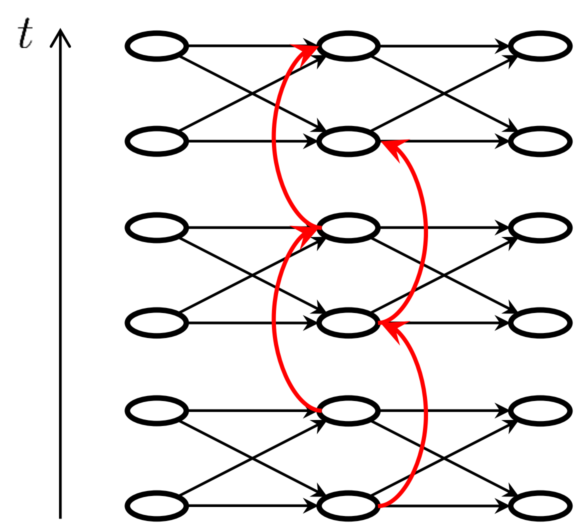
Recurrent neural network

What if we want to capture dependencies between x_t and $a(t+\tau)$?

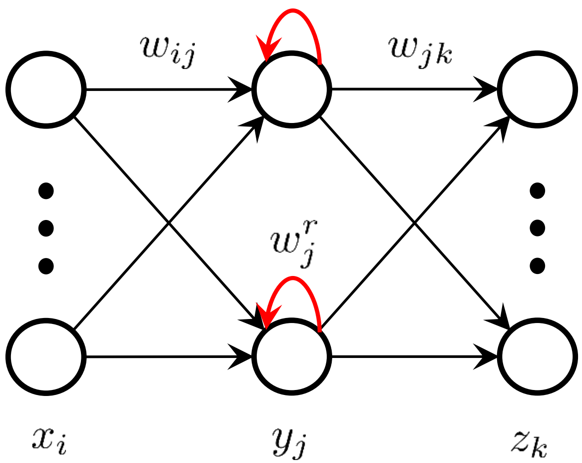


$$y_j(t) = f\left(\sum_i w_{ij} x_i(t) + w_j^r y_j(t-1)\right)$$

trough time



Recurrent neural network



Training sample - sequence:

$$(\bar{x}(t), a(t))_{t=1}^{T}$$

Parameters:

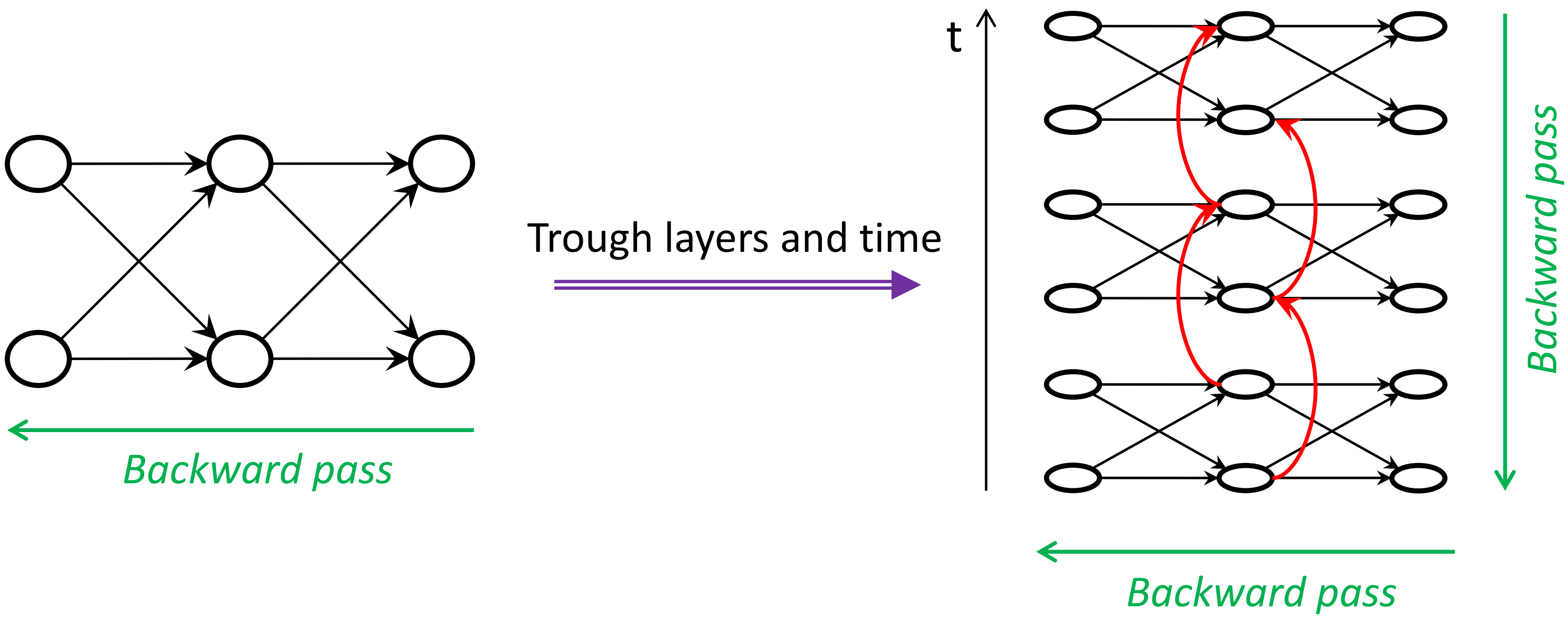
$$\theta = \{w_{ij}, w_{jk}, w_j^r\} \quad \text{for all} \quad i, j, k$$

Loss function:

$$F(Z(\theta), A) = \sum_{t=1}^{T} F_t(\underline{\bar{z}(t, \theta)}, a(t)) \underset{\theta}{\rightarrow} min$$

depends on $x(1), \ldots, x(t)$

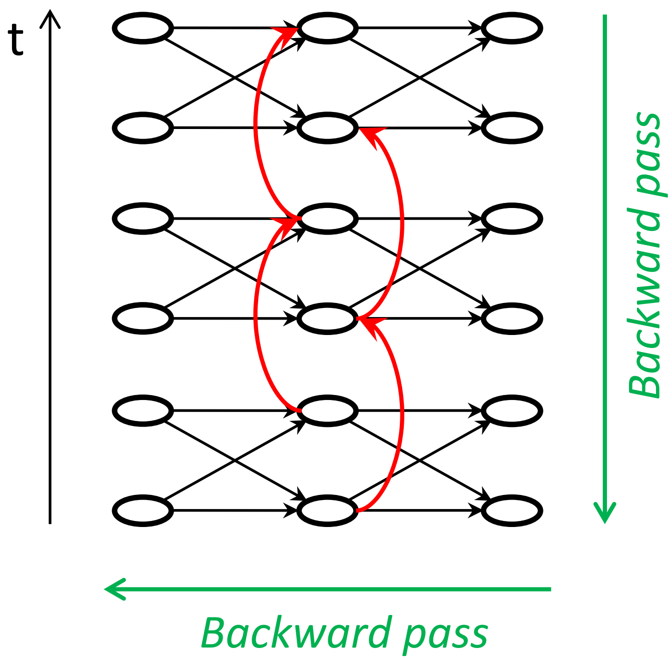
Backpropagation through time



$$\frac{\partial F}{\partial y_j(t)} = \frac{\partial F_t}{\partial y_j(t)}$$

$$\frac{\partial F}{\partial y_j(t)} = \sum_{\tau=t}^{T} \frac{\partial F_\tau}{\partial y_j(t)}$$

Backpropagation through time



Trough layers and time

• • •

$$\frac{\partial F}{\partial y_j(t)} = \sum_{\tau=t}^{T} \frac{\partial F_\tau}{\partial y_j(t)}$$

$$\frac{\partial F_t}{\partial y_j(t)} = \sum_k \frac{\partial F_t}{\partial net_k(t)} w_{jk}$$

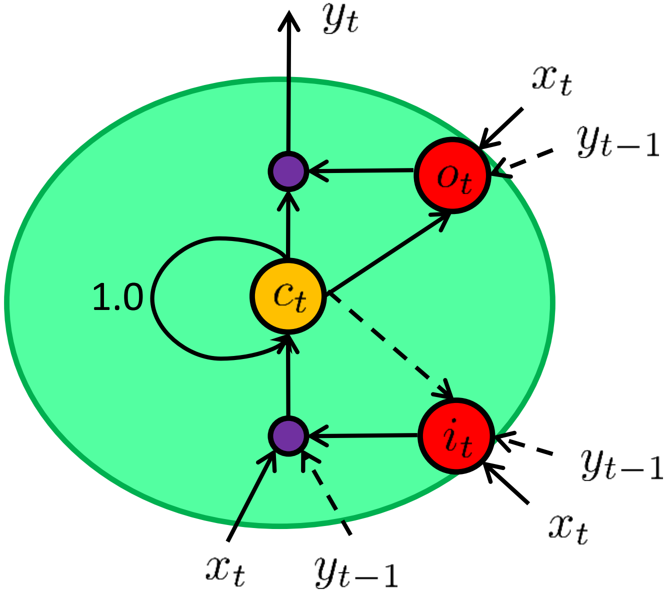
$$\frac{\partial F_\tau}{\partial y_j(t)} = \frac{\partial F_\tau}{\partial y_j(\tau)} \prod_{l=t}^{\tau-1} \frac{\partial y_j(l+1)}{\partial y_j(l)} = \frac{\partial F_\tau}{\partial y_j(\tau)} \prod_{l=t}^{\tau-1} \boxed{f'(net_j(l+1))w_j^r}$$

exploding or vanishing gradients

no long-range dependencies

Long short term memory:

Version 0



i_t, o_t - input and output gates from 0 to 1

c_t - memory

x_t - input, y_t - output

$$i_t = \sigma(w_{ix}x_t + w_{ic}c_{t-1} + w_{iy}y_{t-1} + b_i)$$

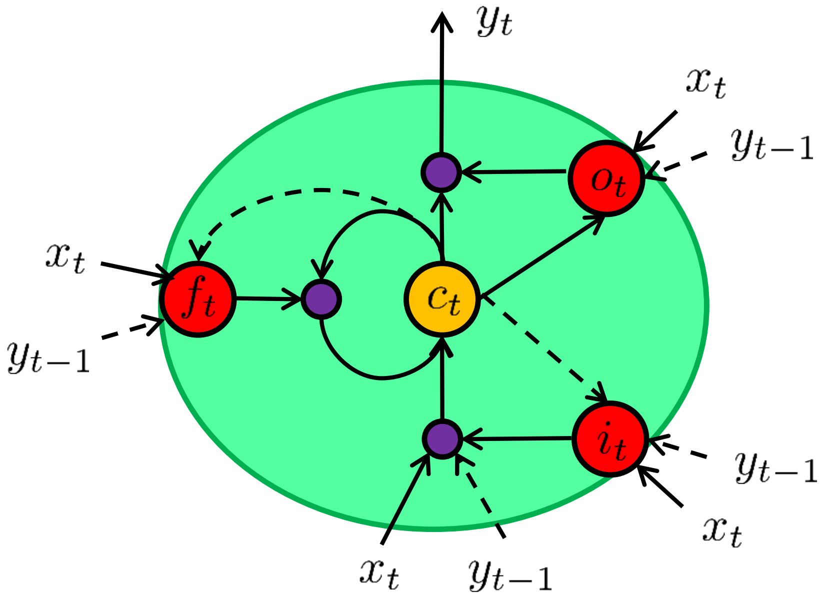
$$o_t = \sigma(w_{ox}x_t + w_{oc}c_t + w_{oy}y_{t-1} + b_o)$$

$$c_t = c_{t-1} + i_t \cdot tanh(w_{cx}x_t + w_{cy}y_{t-1}) \qquad y_t = o_t \cdot tanh(c_t)$$

Now we have infinitely long memory

Long short term memory:

Version 1



i_t, f_t, o_t - input, forget and output gates from 0 to 1

c_t - memory

x_t - input, y_t - output

$$i_t = \sigma(w_{ix}x_t + w_{ic}c_{t-1} + w_{iy}y_{t-1} + b_i)$$

$$f_t = \sigma(w_{fx}x_t + w_{fc}c_{t-1} + w_{fy}y_{t-1} + b_f)$$

$$o_t = \sigma(w_{ox}x_t + w_{oc}c_t + w_{oy}y_{t-1} + b_o)$$

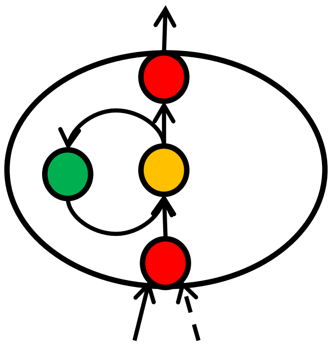
$$c_t = f_t c_{t-1} + i_t \cdot tanh(w_{cx}x_t + w_{cy}y_{t-1}) \qquad y_t = o_t \cdot tanh(c_t)$$

Now we can restart memory

Long short term memory:

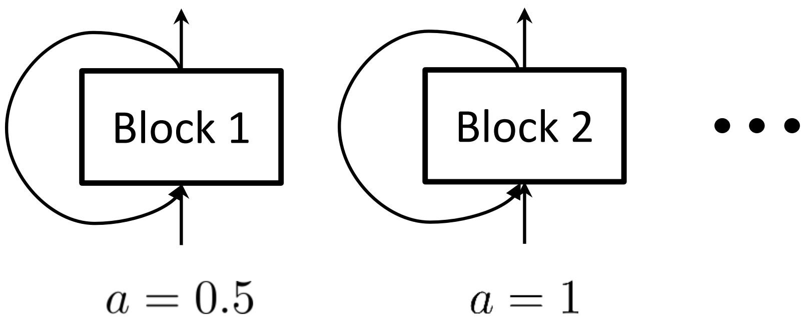
Initialization

LSTM cell:



$$b_i = -a < 0$$
$$b_f = a > 0$$
$$b_o = -a < 0$$

LSTM layer:

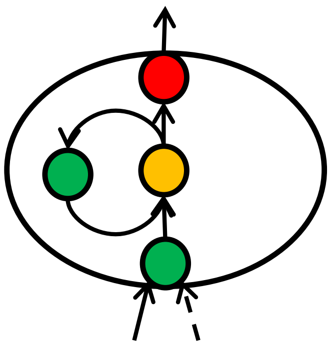


- Blocks start work one by one
- Blocks capture dependencies with different ranges
- We learn to forget only if it necessary

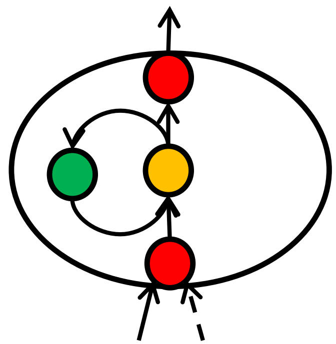
Long short term memory:

Examples

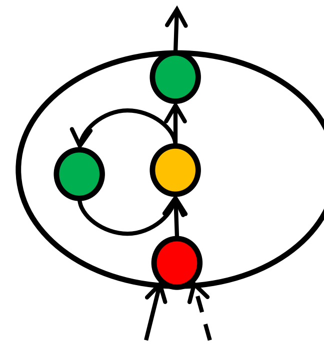
Captures info



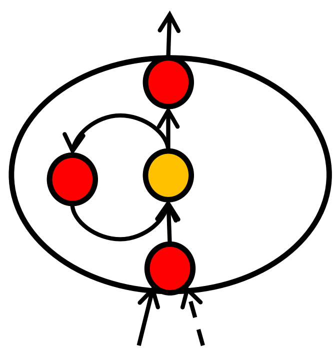
Keeps info



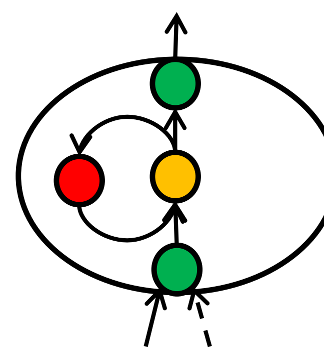
Releases info

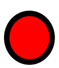


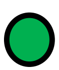
Erases info



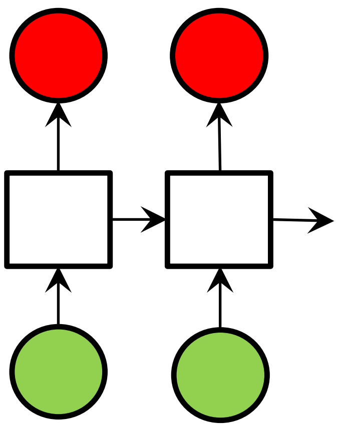
= RNN



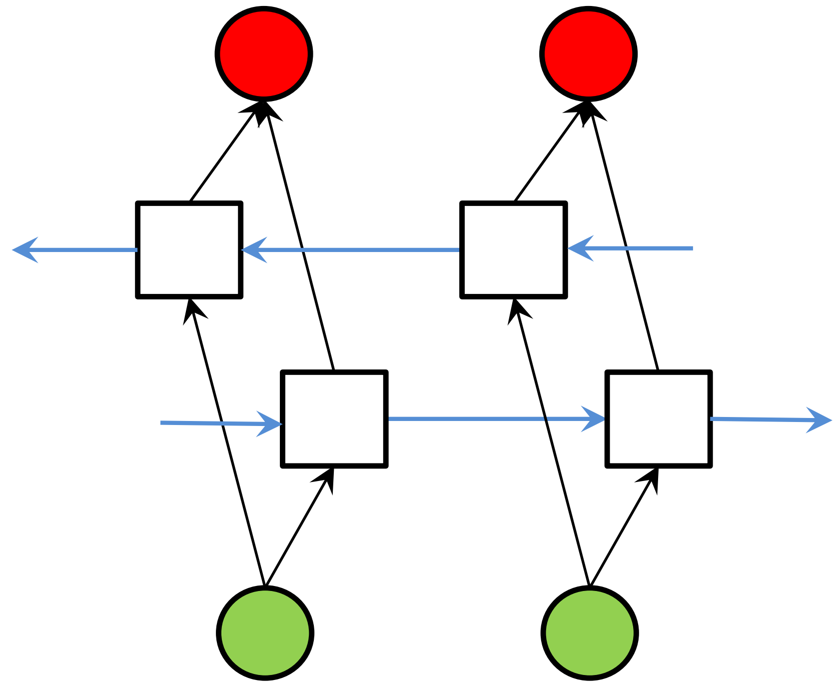
 - gate is close

 - gate is open

Bidirectional RNN



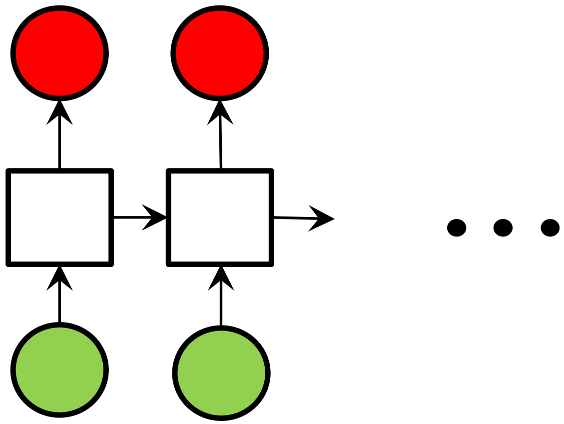
RNN

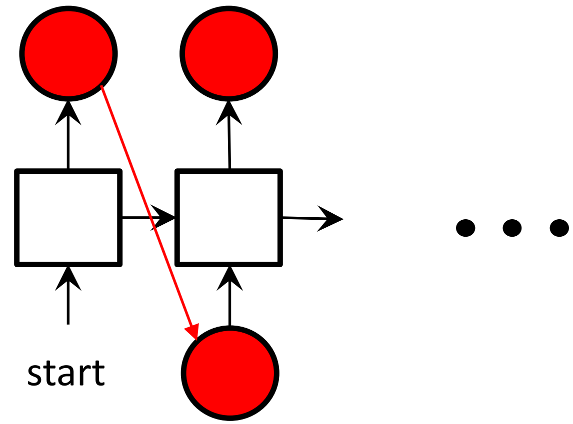


Bidirectional RNN

RNN and LSTM examples

Architecture 1





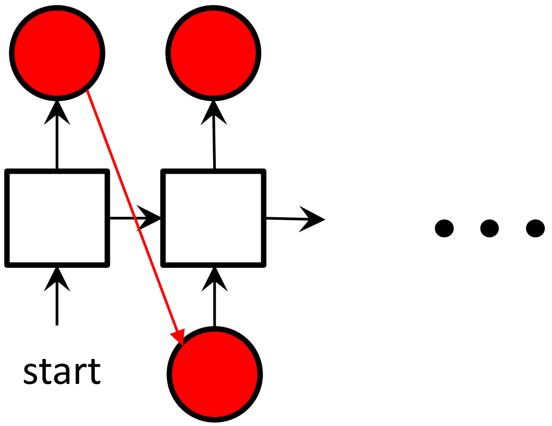
- Element-wise classification of a sequence
- Element-wise translation of a sequence

- Sequence generation

Text generation

Next symbol/word

Current symbol/word



[Demo](#) (character-wise, MRNN)

Text generation:

Example results of the word-wise RNN

YOU WOULD NOT SUFFER WHAT HE WAS PROMOTING IN A NATION IN THE CENTRAL INDUSTRY AND CAME TO IRAN AND HE DID AND HE HAVE PROMISED THEY'LL BE ANNOUNCING HE'S FREE THE PEACE PROCESS

SHARON STONE SAID THAT WAS THE INFORMATION UNDER SURVEILLING SEPARATION SQUADS

PEOPLE KEPT INFORMED OF WHAT DID THEY SAY WAS THAT %HESITATION

WELL I'M ACTUALLY A DANGER TO THE COUNTRY THE FEAR THE PROSECUTION WILL LIKELY MOVE

WELL THAT DOES NOT MAKE SENSE

THE WHITE HOUSE ANNOUNCED YESTERDAY THAT THE CLINTON ADMINISTRATION ARRESTED THIS PRESIDENT OFTEN CONSPICUOUSLY RELIEVED LAST DECEMBER AND AS A MEMBER OF THE SPECIAL COMMITTEE

THE GUARDIAN EXPRESSED ALL DESIRE TO LET START THE INVESTIGATION

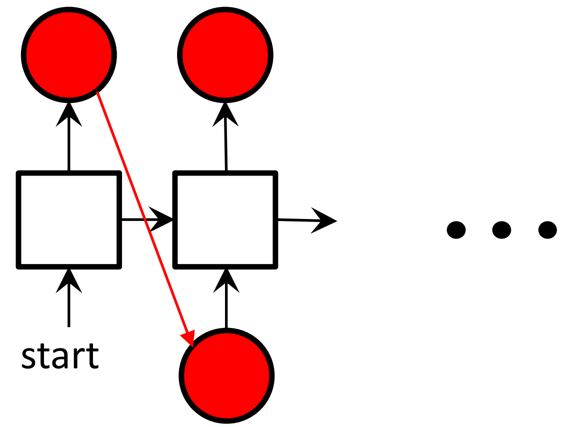
IN NORTH KOREA THIS IS A JOKE

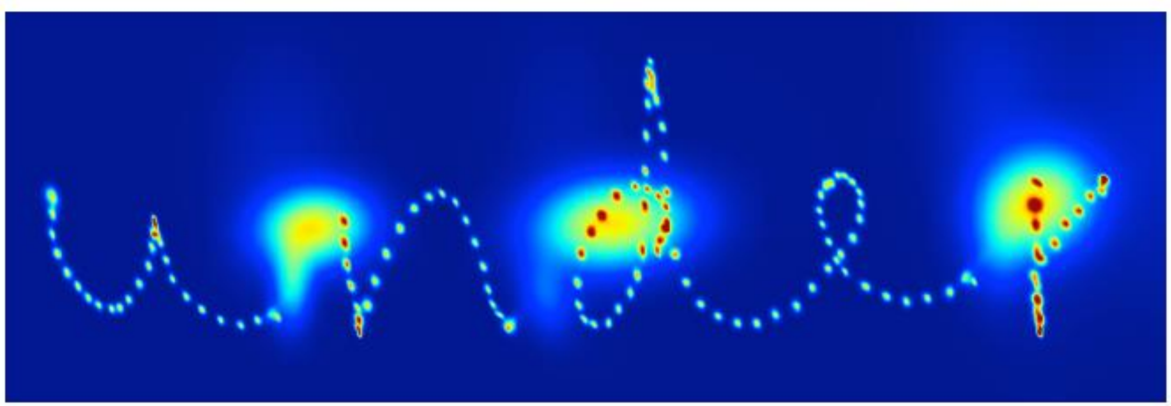
Handwriting generation:

handwriting -> handwriting

Next pen position (we predict parameters):
x1,x2 - mixture of bivariate Gaussians
x3 - Bernoulli distribution

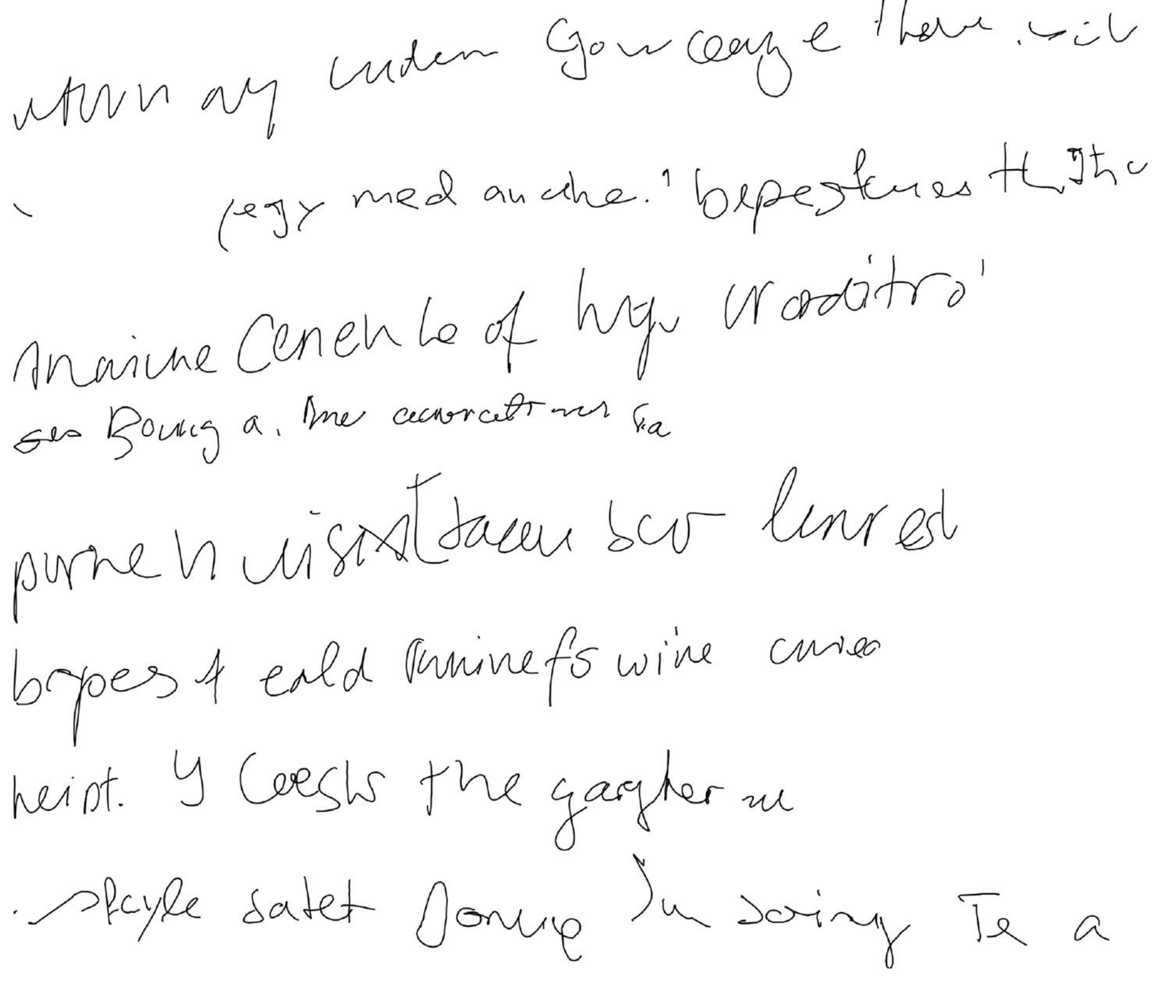
Current pen position:
x1,x2 – pen offset
x3 – is it end of the stroke





Handwriting generation:

example



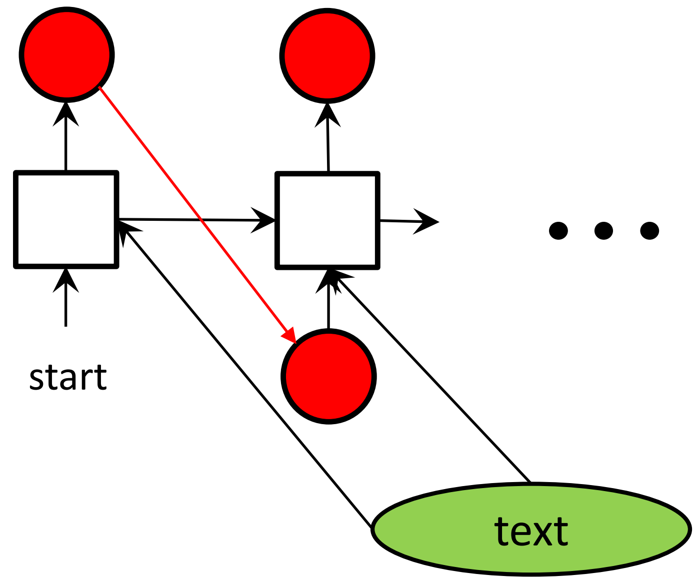
Handwriting synthesis:

text -> handwriting

Next pen position

Current pen position

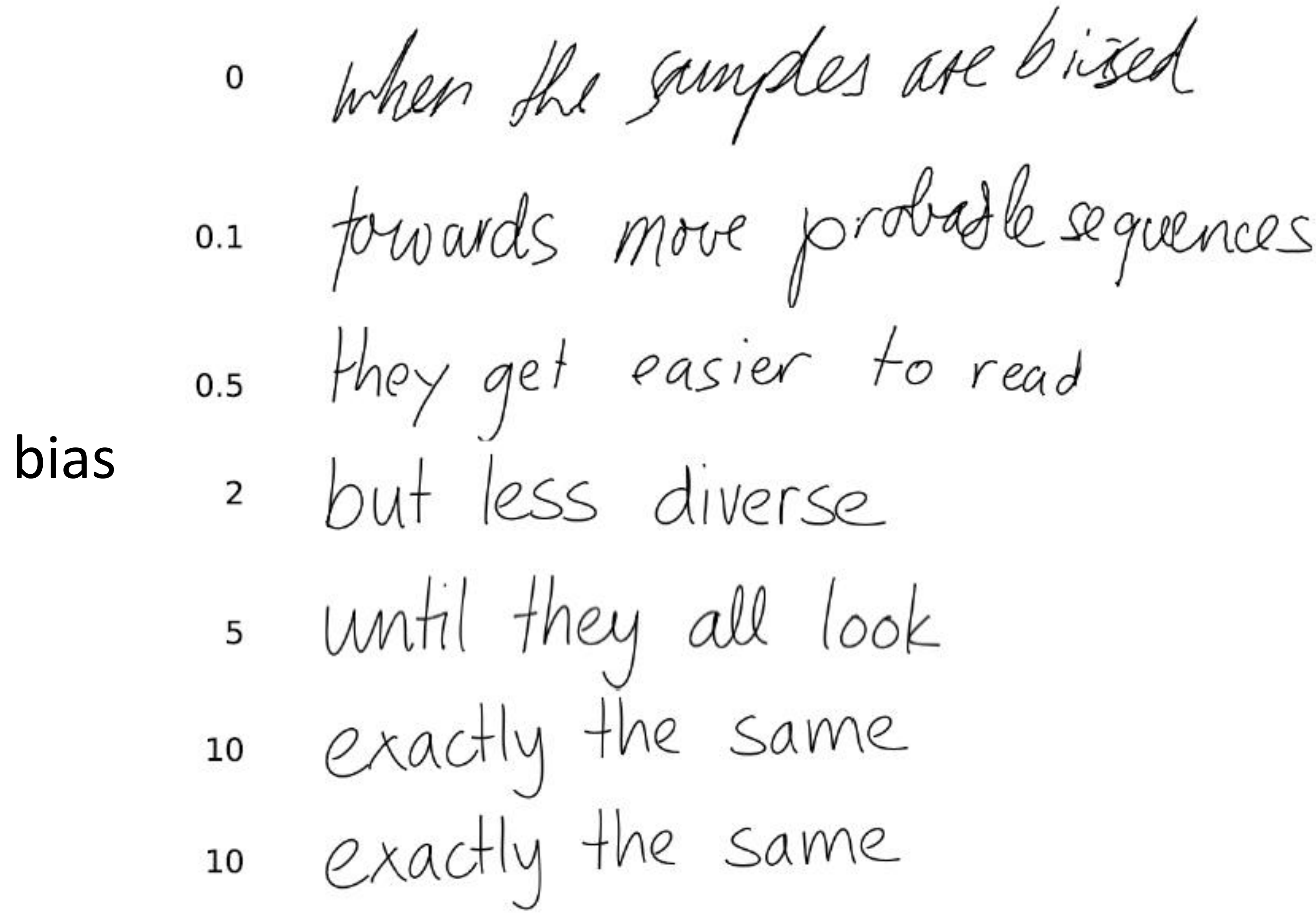
Which letter we write now



[Demo]

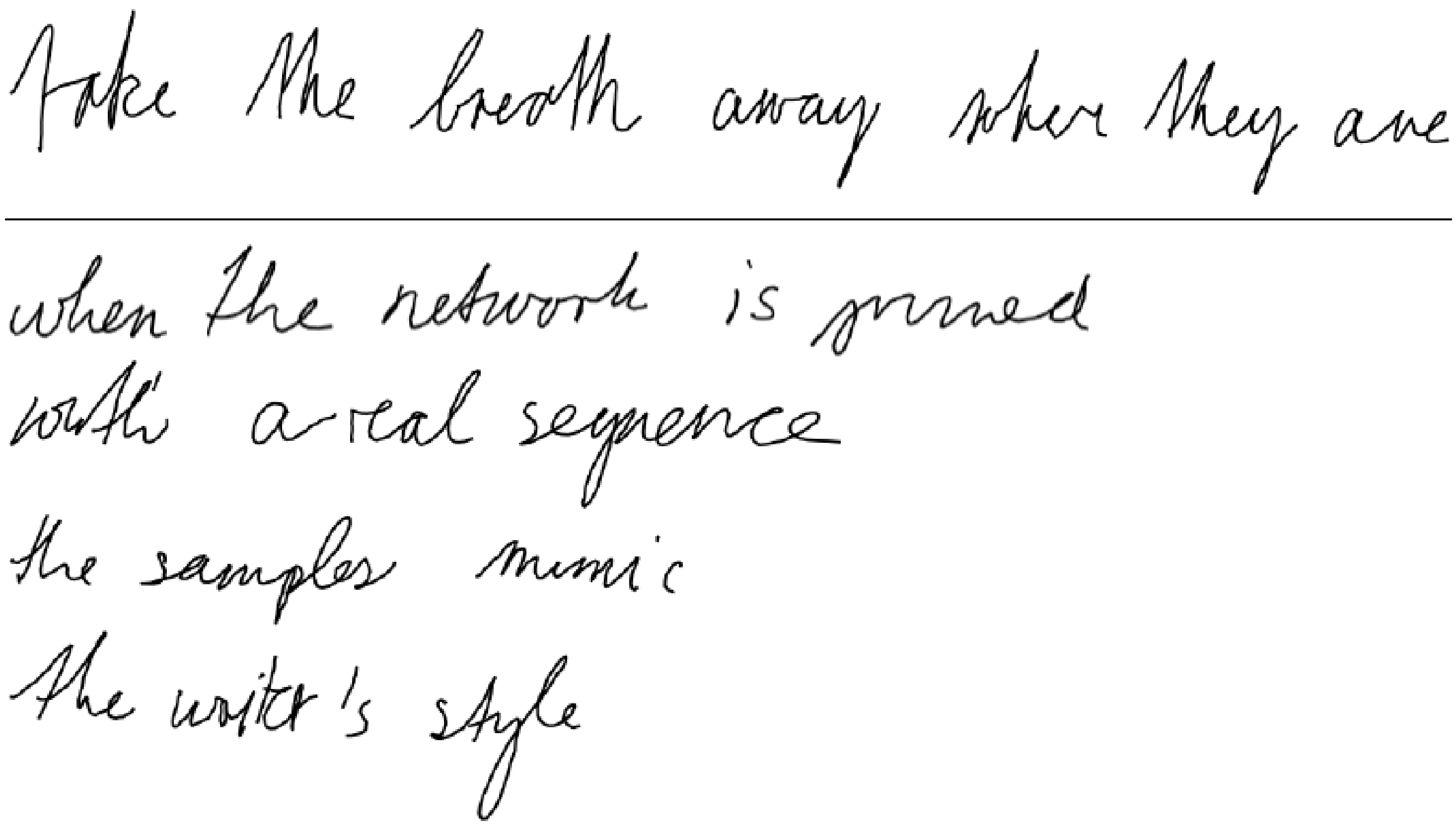
Handwriting synthesis:

biased sampling



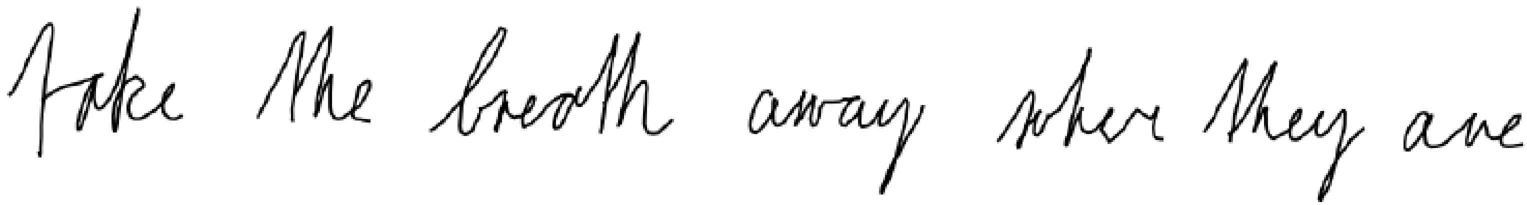
Handwriting synthesis:

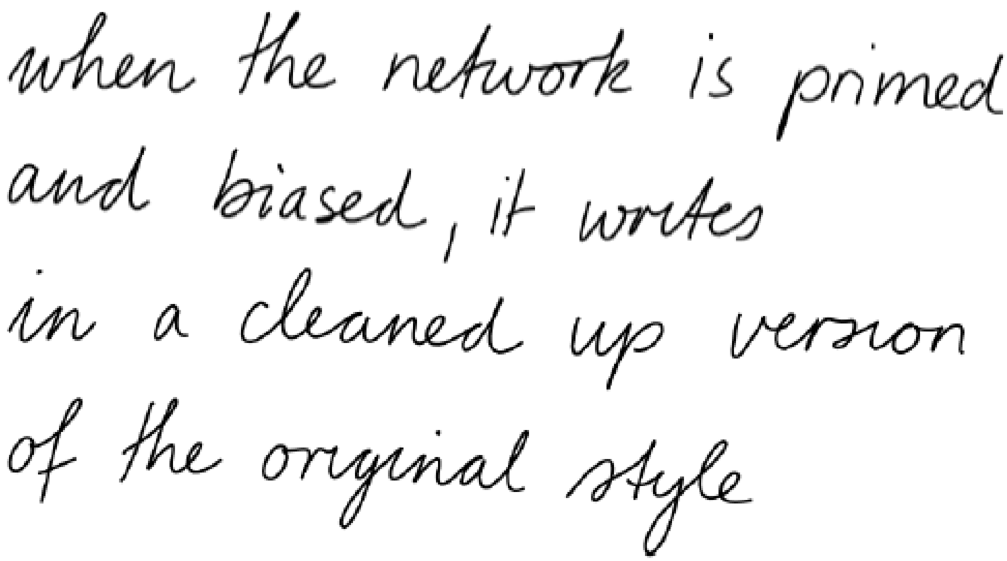
primed sampling



Handwriting synthesis:

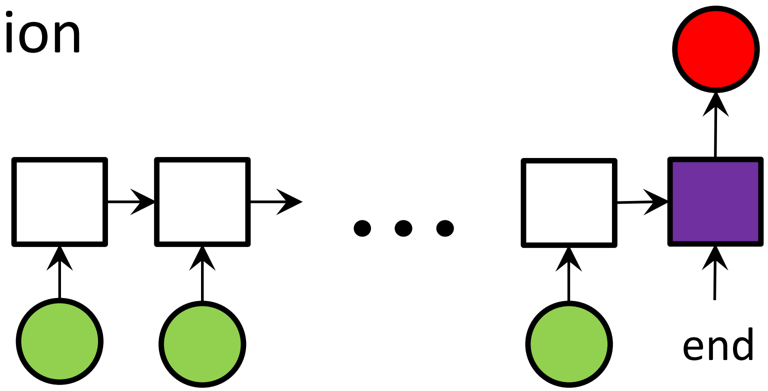
primed and biased sampling



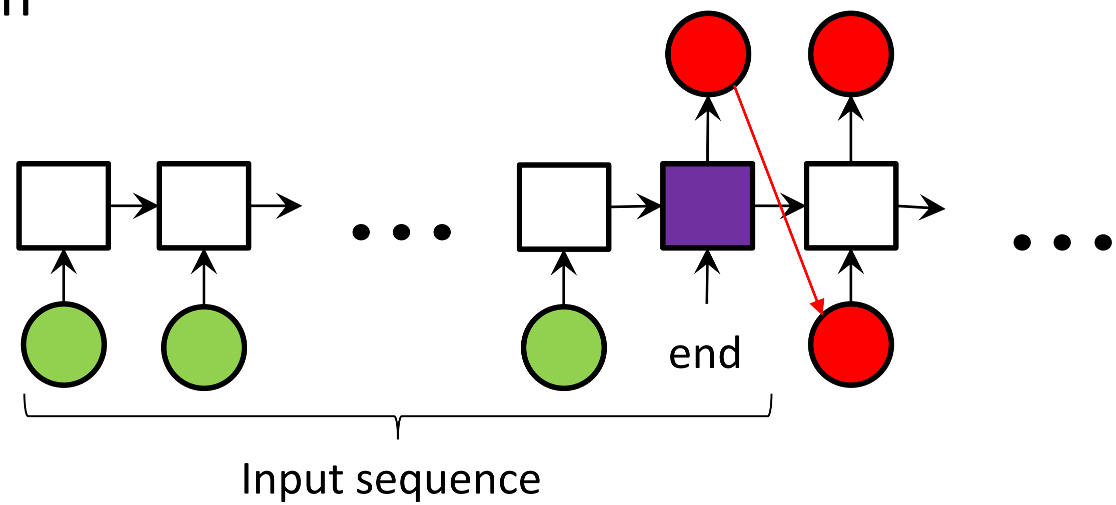


Architecture 2

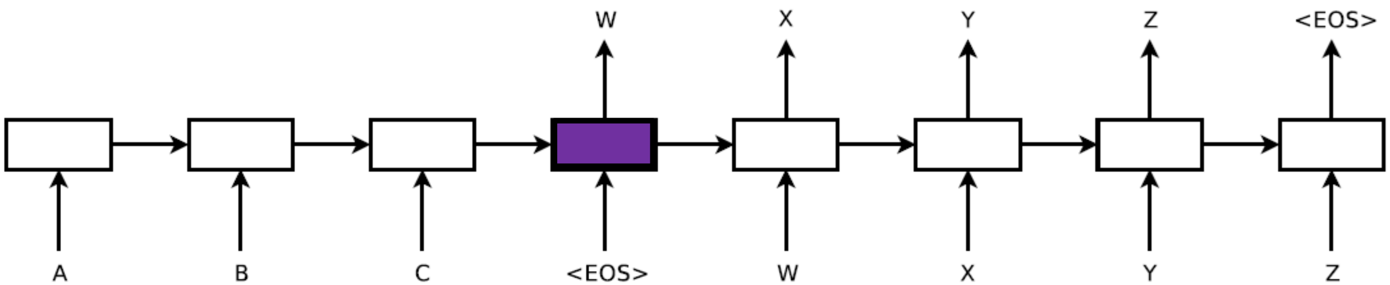
- Sequence classification



- Sequence translation



Language translation



Demo with bidirectional RNN

Architecture 3

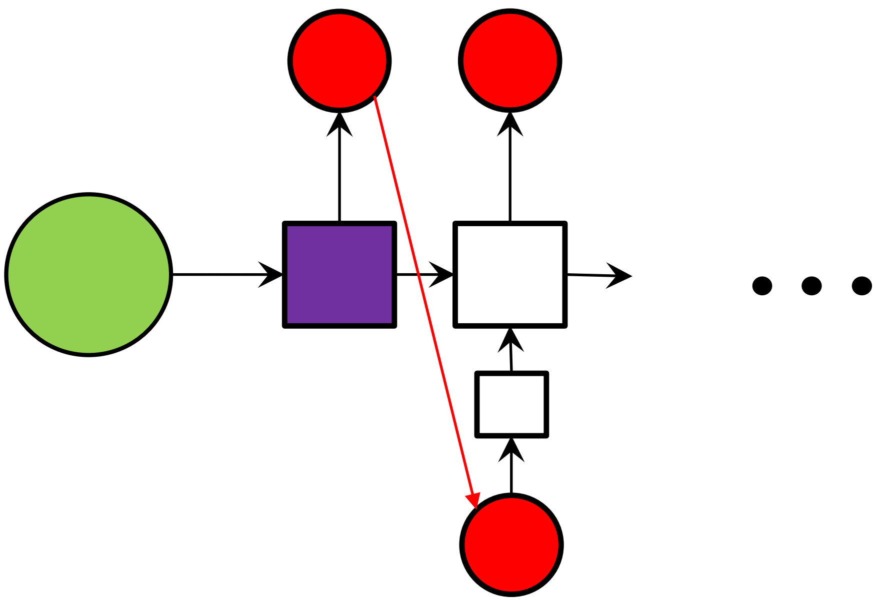
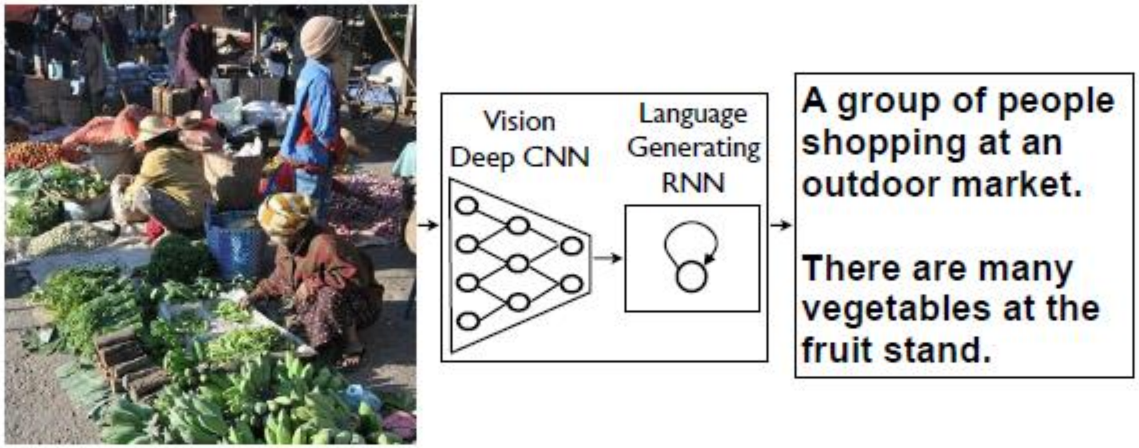


Image Caption Generation



[Demo](images) (images)

Demo (top images for test texts)

Demo (more sophisticated model)

Reference

Theory

Hochreiter, Sepp, and Jurgen Schmidhuber. Long short-term memory // Neural computation 9.8: 1735-1780. 1997.

F. A. Gers. Long Short-Term Memory in Recurrent Neural Networks // PhD thesis, Department of Computer Science, Swiss Federal Institute of Technology, Lausanne, EPFL, Switzerland, 2001.

J. Schmidhuber. Long Short-Term Memory: Tutorial on LSTM Recurrent Networks.

J. Schmidhuber. Deep Learning in Neural Networks: An Overview.

Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, Jürgen Schmidhuber. LSTM: A Search Space Odyssey.

Mike Schuster and Kuldip K. Paliwal . Bidirectional Recurrent Neural Networks. // IEEE TRANSACTIONS ON SIGNAL PROCESSING, VOL. 45, NO. 11, 1997

Libraries

Different deep architectures -

- torch
- pybrain
- theano (see tutorials)

RNN and LSTM library by Graves - rnnlib

RNN Language Models by Mikolov - rnnlm

Reference: examples

Sequence generation

- **Character-wise text generation with Multiplicative RNN**

  Ilya Sutskever, James Martens, and Geoffrey Hinton. Generating Text with Recurrent Neural Networks // ICML 2011.

  demo, slides

- **Word-wise text generation with RNN (RNN vs n-grams)**

  Mikolov Tomá, Karafiát Martin, Burget Luká, Èernocký Jan, Khudanpur Sanjeev. Recurrent neural network based language model. // Proceedings of the 11th Annual Conference of the International Speech Communication Association (INTERSPEECH 2010).

  Mikolov Tomá. Statistical Language Models based on Neural Networks // PhD thesis, Brno University of Technology, 2012.

  lib+demo

- **Both character and word-wise text generation + handwritten generation + handwritten synthesis (all with LSTM)**

  A. Graves. Generating Sequences With Recurrent Neural Networks.

  slides, handwritten synthesis demo

Reference: examples

Sequence translation

Ilya Sutskever, Oriol Vinyals, Quoc Le. Sequence to Sequence Learning with Neural Networks // NIPS 2014

K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation // EMNLP 2014.

demo

Image Caption Generation

O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator // *CVPR*, 2015.

Andrej Karpathy, Li Fei-Fei. Deep Visual-Semantic Alignments for Generating Image Descriptions // CVPR, 2015.

demo (images), demo (top images for test texts)

Ryan Kiros, Ruslan Salakhutdinov, Richard Zemel. Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models // TACL, 2015

demo