Глубокое обучение
Рекуррентные нейросети

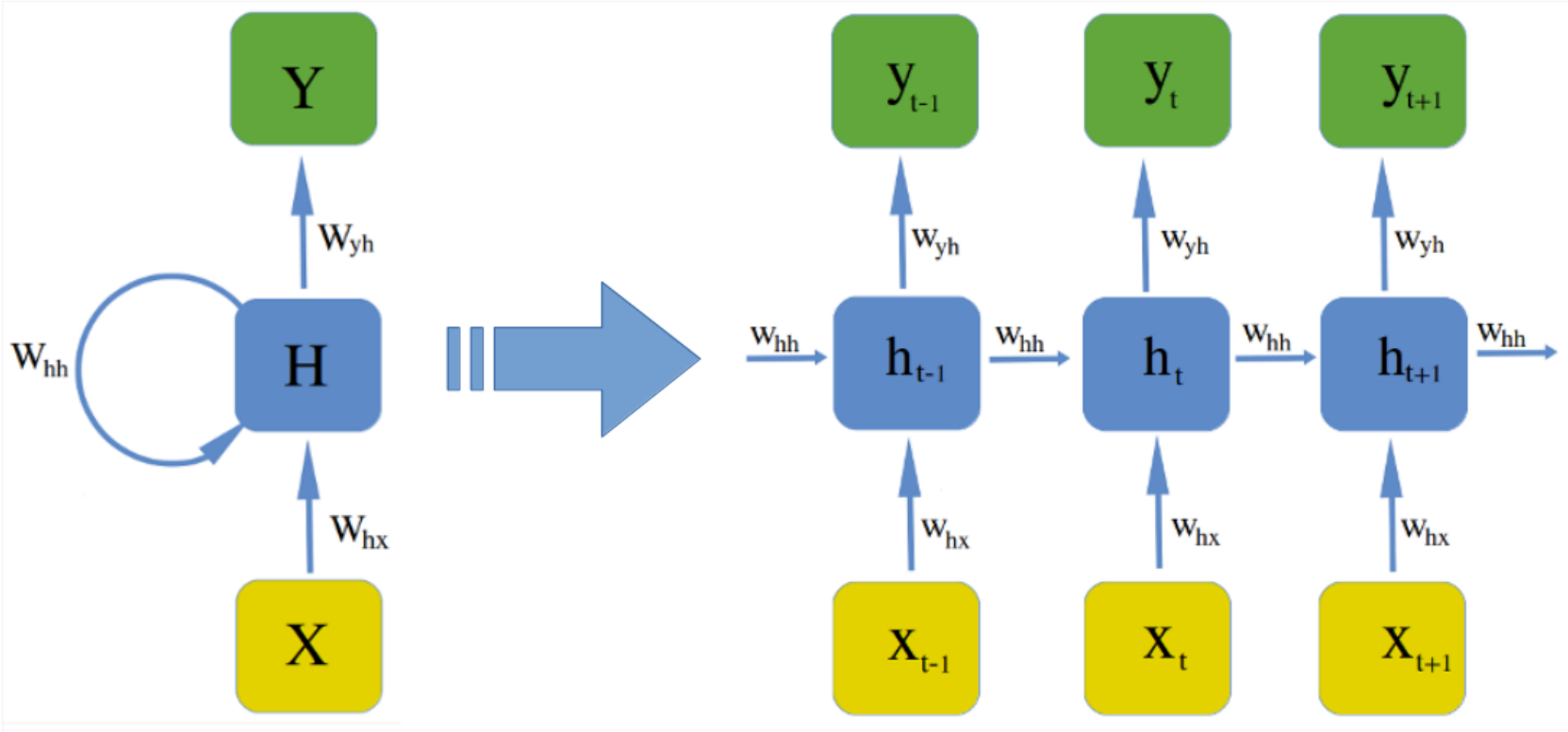
Дьяконов А.Г.

Московский государственный университет имени М.В. Ломоносова (Москва, Россия)

Рекуррентная нейросеть (RNN = Recurrent neural network)

– для обработки последовательностей

легко масштабируется при увеличении длины последовательностей



$$p(x_1,\ldots,x_T)=\prod_{t=1}^{T} p(x_t \mid x_{t-1},\ldots,x_1)$$

http://www.jefkine.com/general/2018/05/21/2018-05-21-vanishing-and-exploding-gradient-problems/

Рекуррентная нейросеть (RNN = Recurrent neural network)

Главная идея – разделение параметров (Parameter sharing)

как и в свёртках;)

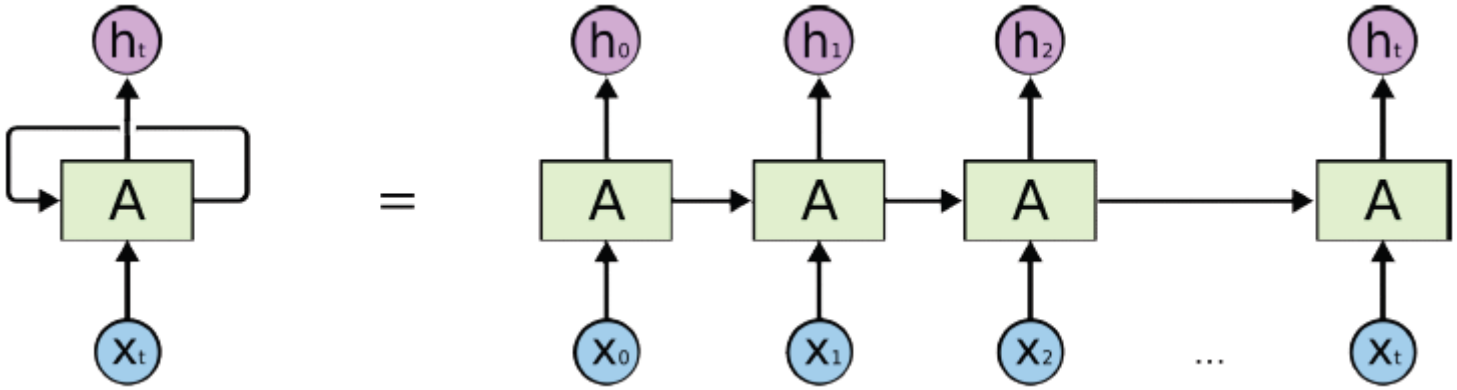
Матрицы весов одинаковые при обработке любого элемента последовательности (символ, слово, …)

Учим одну модель, которая применяется на каждом шаге к последовательности любой длины

Дальше использованы рисунки из
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

RNN (базовый блок)

Vanilla RNN



$$h_0 = \sigma(W_{xh} x_0)$$

$$\dots$$

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

$$y_t = g(W_{hy} h_t)$$

Это для однослойной сети!

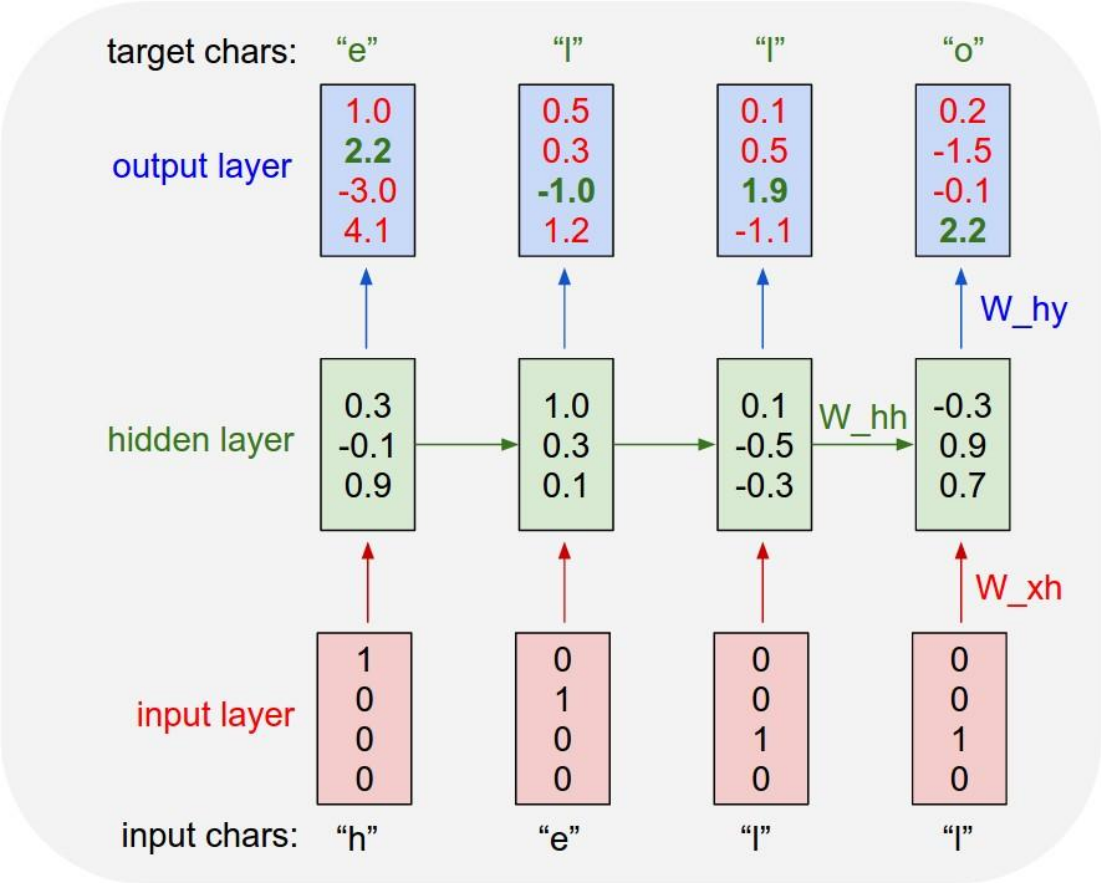
линейный слой + нелинейность

без свободного члена

индексы!!!

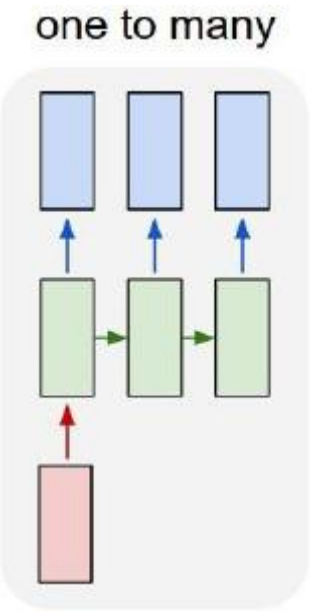
Обратное распространение во времени (BPTT = Backpropagation through time): пройтись по последовательности вперёд и назад

Пример работы RNN

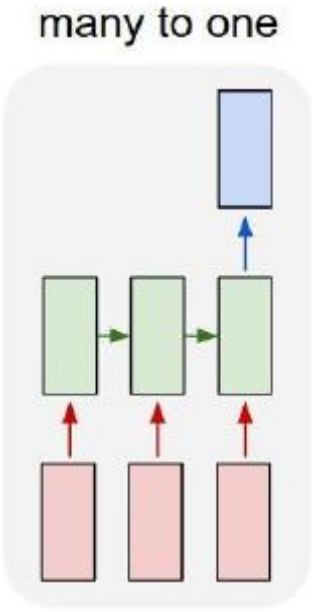


http://karpathy.github.io/2015/05/21/rnn-effectiveness/

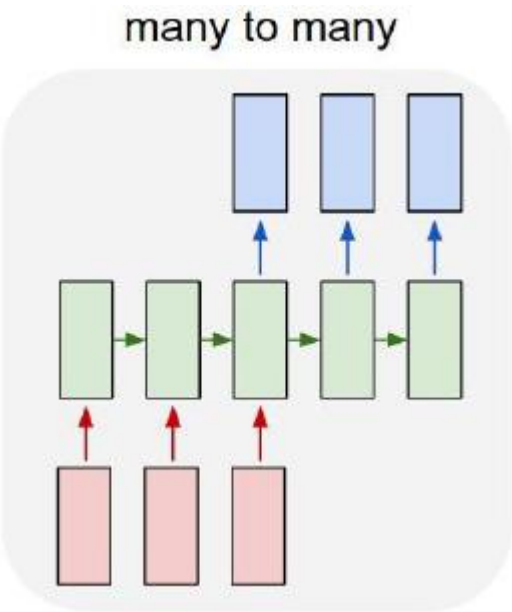
Применение RNN



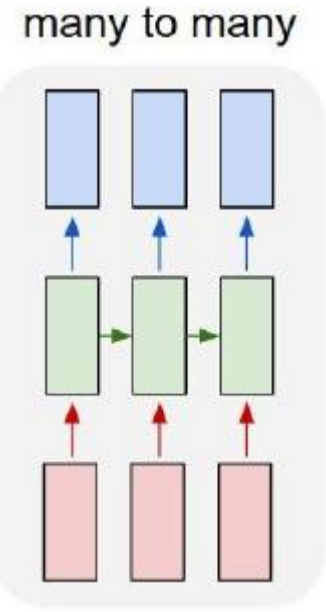
описание изображения



тема / настроение текста



машинный перевод

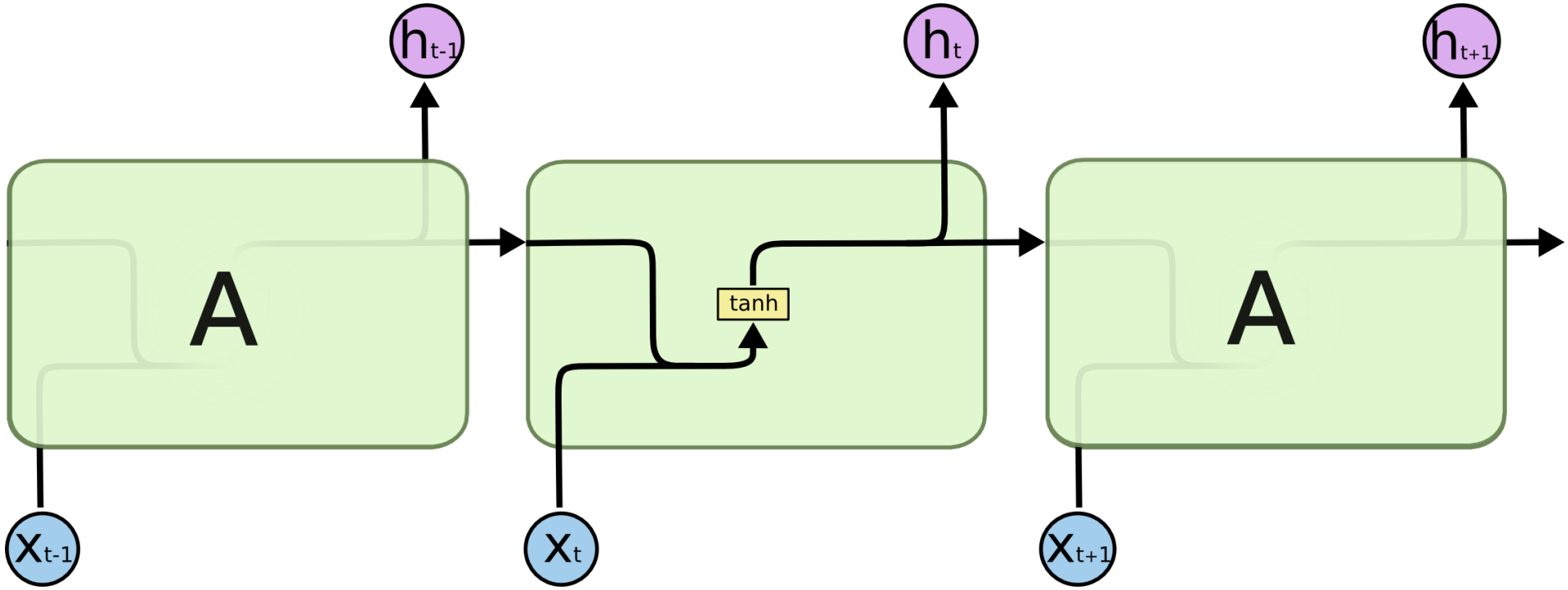


классификация фреймов видео

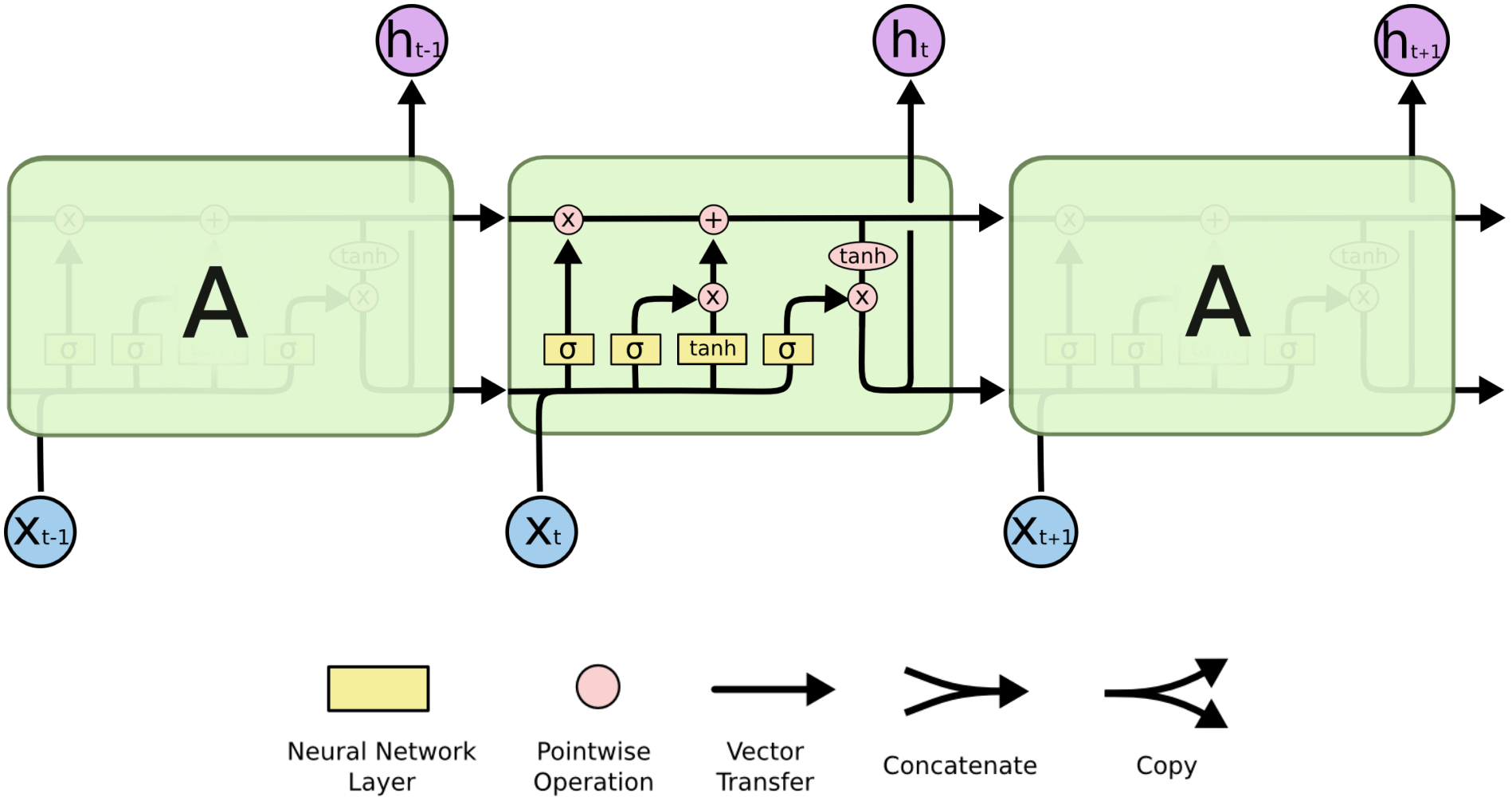
Можно по-разному собирать блоки – для решения разных задач

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Стандартная RNN



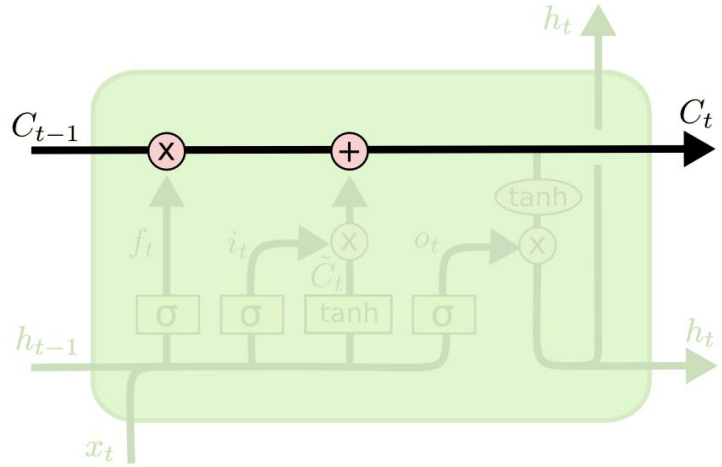
LSTM (другой базовый блок)



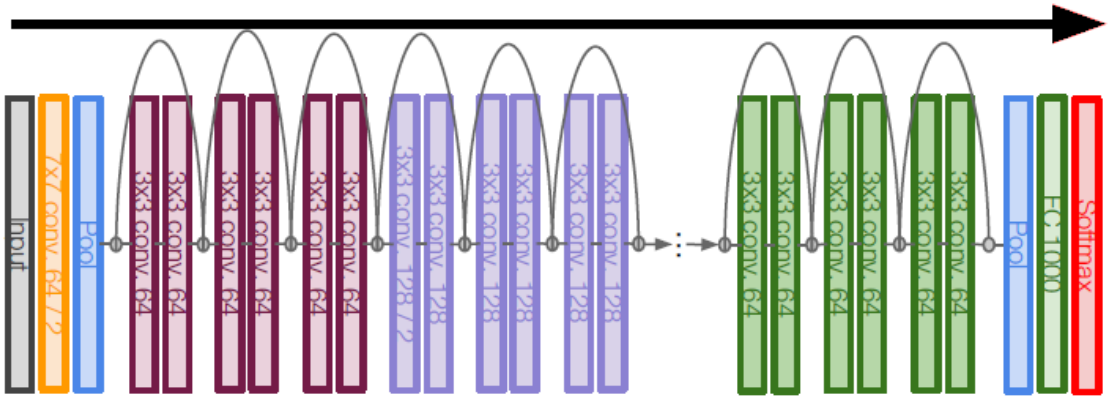
[Hochreiter&Schmidhuber, 1997]

Ключевая идея LSTM

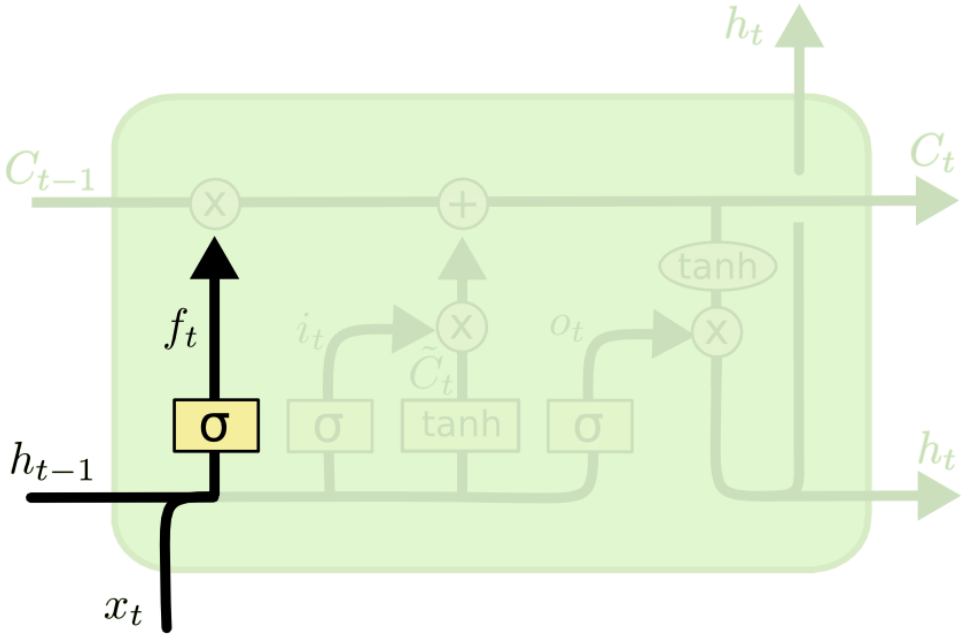
«состояние ячейки/блока» – проходит через все блоки



- **память** перенос информации, которая должна «слабо меняться»
- **борьба с затухающим градиентом** свободно протекает, как в ResNet



Забывающий гейт (Forget Gate)

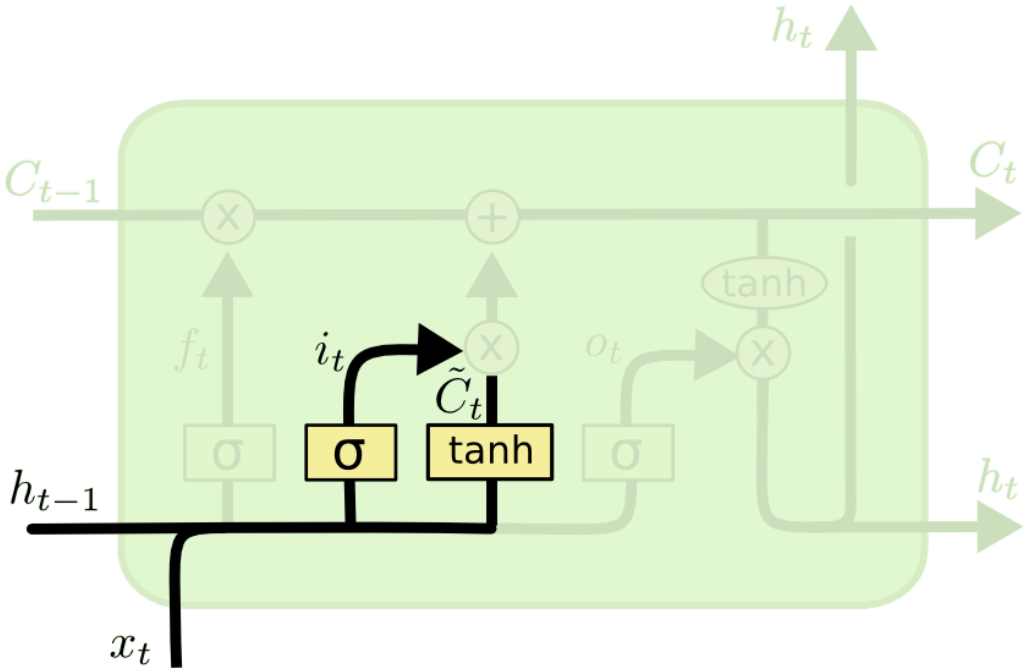


$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_t)$$

если = 1 – передаём полностью состояние блока

если = 0 – то забываем предыдущее состояние

Входной гейт (Input Gate)



входной гейт:

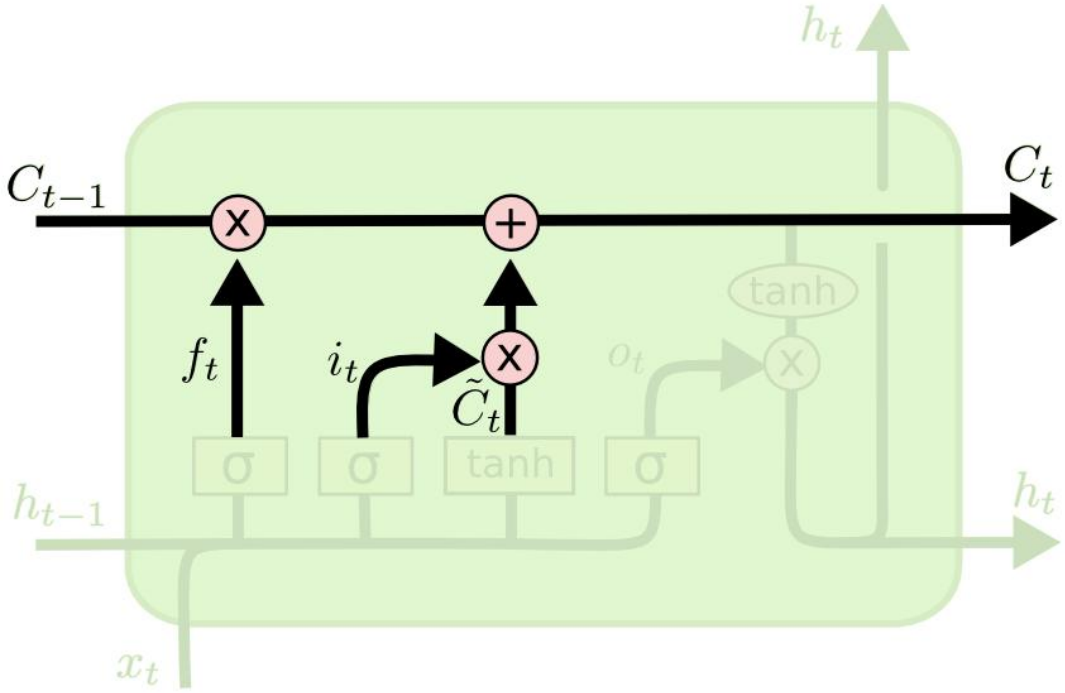
$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$

текущее состояние:

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

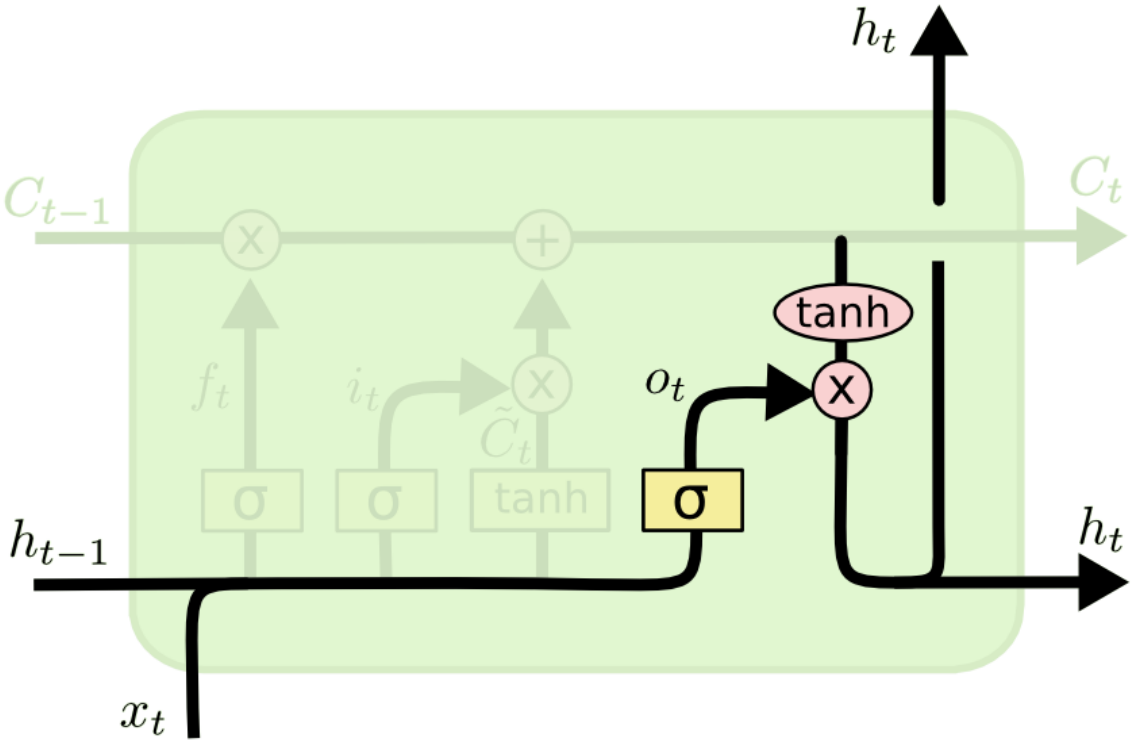
Какую новую информацию учитываем в состоянии…

Обновление состояния (Cell update)



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

Выходной гейт (Output Gate)



выходной гейт:

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

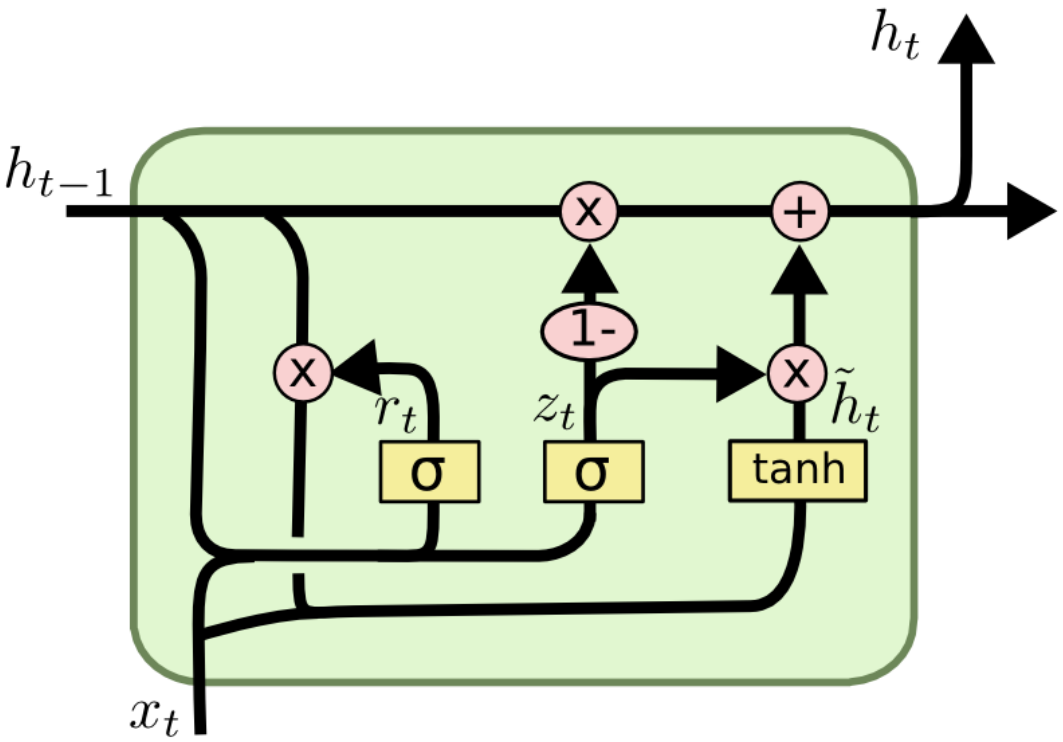
скрытое состояние:

$$h_t = o_t \tanh(C_t)$$

LSTM (Long Short Term Memory)

Есть и другие варианты,
которые отличаются построением базового блока

Gated Recurrent Unit (GRU)



$$z_t = \sigma(W_i[h_{t-1}, x_t] + b_z)$$

$$r_t = \sigma(W_r[h_{t-1}, x_t] + b_r)$$

$$\tilde{h}_t = \tanh(W[r_t h_{t-1}, x_t])$$

$$h_t = (1 - z_t)h_{t-1} + z_t\tilde{h}_t$$

гейт обновления = забывающий + входной

состояние = состояние + скрытое состояние

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, Yoshua Bengio Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, 2014 // https://arxiv.org/abs/1406.1078

Какие блоки лучше?

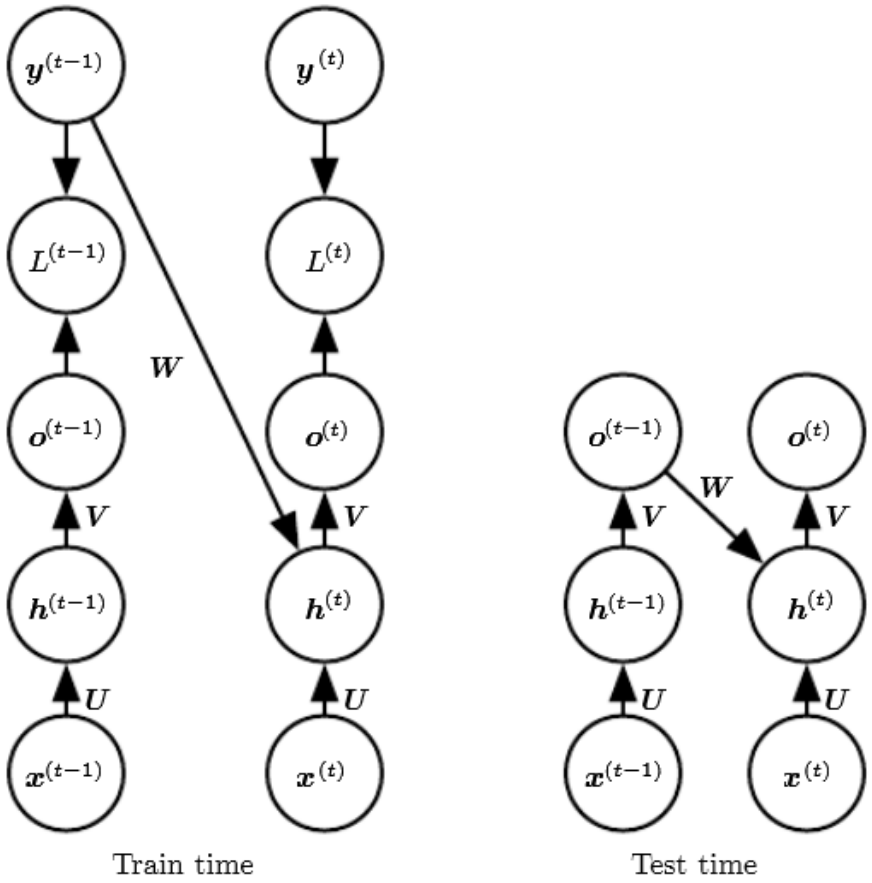
Есть обзоры

LSTM: A Search Space Odyssey 2015 https://arxiv.org/pdf/1503.04069.pdf

An Empirical Exploration of Recurrent Network Architectures 2015
http://proceedings.mlr.press/v37/jozefowicz15.pdf

Приёмы: метод форсирования учителя (teacher forcing)

Вместо выхода модели на предыдущем шаге подаём истинную метку



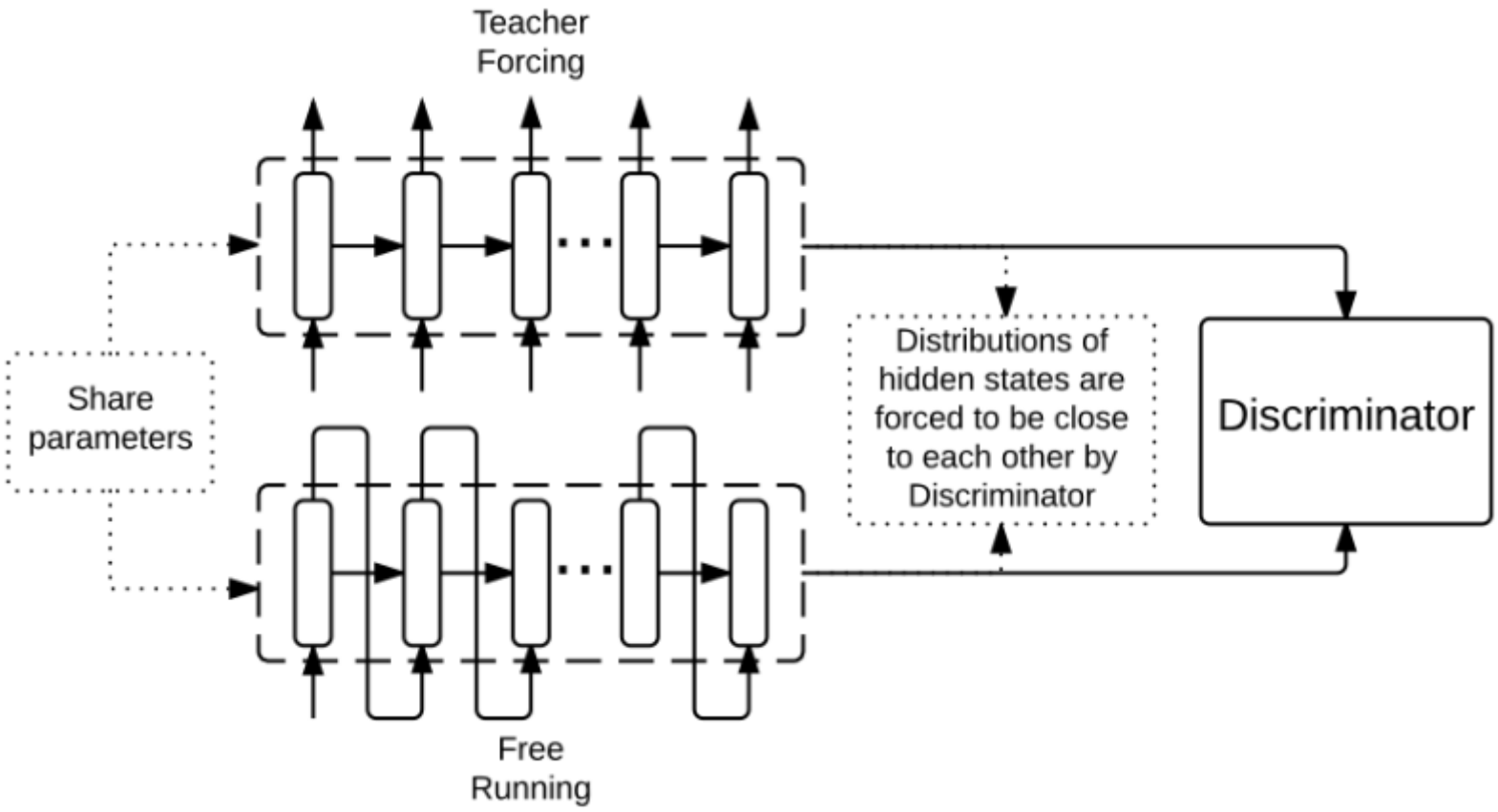
Только если связь типа «выход-модель» (не передаётся скрытая переменная)

+ можно не делать BPTT

– то что видит при тестировании и обучении может отличаться

+ можно использовать для предтренировки

Приёмы: метод форсирования учителя (teacher forcing)

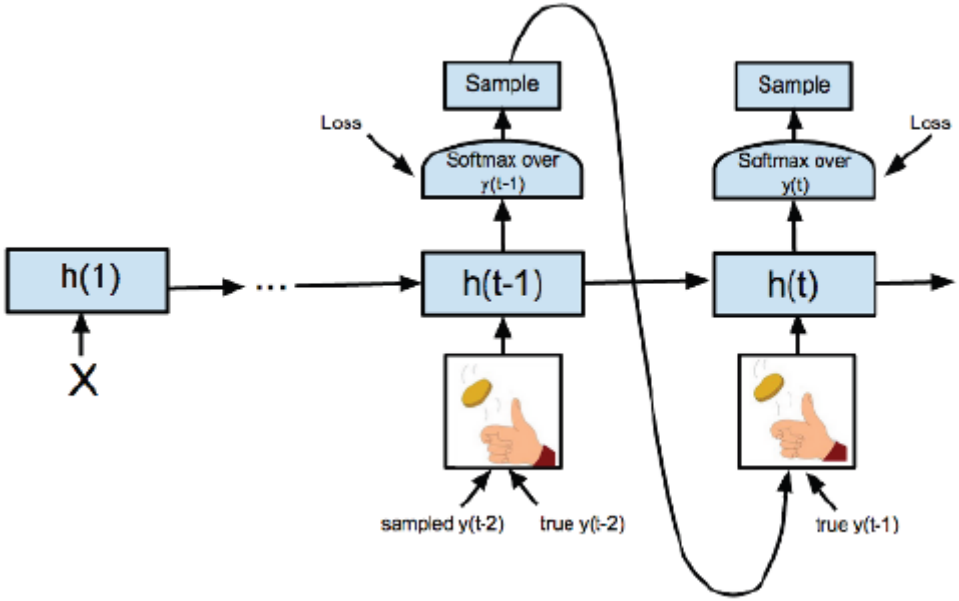


можно хитрее: одновременно истинная метка и сгенерированная

Приёмы: Scheduled sampling

Проблема при обучении RNN
В обучении на вход последовательность из выборки
При тесте – сгенерированная (может накапливаться ошибка)

Выход – Scheduled sampling (S. Bengio et al, NIPS 2015)
при обучении «смешиваем» значение из выборки с генерированным



Приёмы: остановка

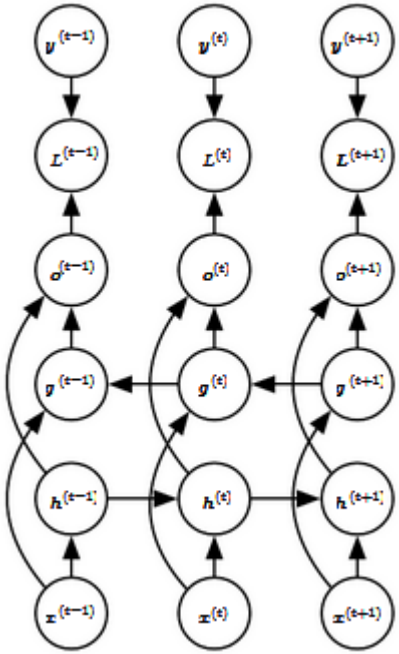
Когда останавливать генерацию последовательности с помощью RNN?

- **ввести спецсимвол «конец»**

- **ещё один выход – вероятность конца работы**

годится и для вывода последовательности чисел

Двунаправленные (Bidirectional) RNN



распознавание рукописных текстов

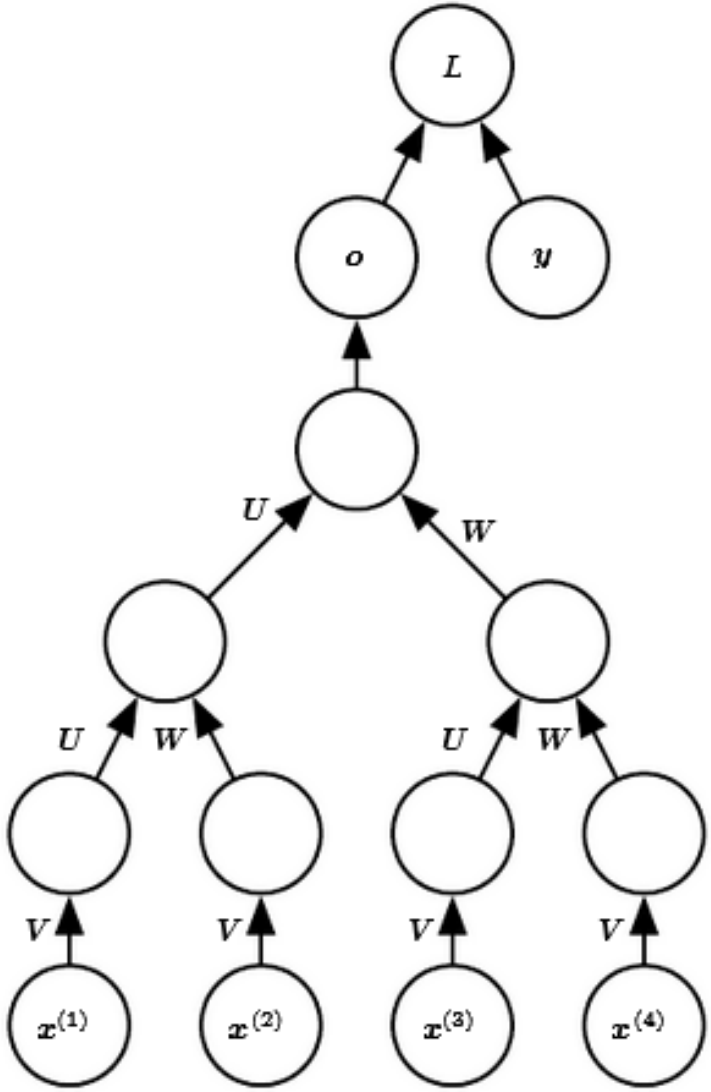
распознавание речи

биоинформатика

Многонаправленные RNN

...

Рекурсивные (Recursive Neural Networks) НС



Самая главная проблема RNN – Exploding / Vanishing gradients

$$h_0 = \sigma(W_{xh} x_0)$$

$$\dots$$

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

$$y_t = g(W_{hy} h_t)$$

Делаем BPTT…

$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L}{\partial y_t} \frac{\partial y_t}{\partial h_t} \frac{\partial h_t}{\partial h_k} \frac{\partial h_k}{\partial W}$$

$$\frac{\partial h_t}{\partial h_k} = \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial h_{t-2}} \dots \frac{\partial h_{k+1}}{\partial h_k} = \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} =$$

– произведение Якобианов

$$= \prod_{i=k+1}^{t} W^{\mathrm{T}} \operatorname{diag}(\sigma'(h_i))$$

Самая главная проблема RNN – Exploding / Vanishing gradients

Чем плохо произведение Якобианов?

Даже если просто «рекуррентно» умножать на матрицу

$$h_t = W_{hh} h_{t-1}$$

т.е. $\sigma(z) = z$**. Получаем…**

$$\prod_{i=k+1}^{t} W^{\mathrm{T}} \operatorname{diag}(\sigma'(h_i)) = (W^{\mathrm{T}})^{t-k}$$

Возведение в степень…
или экспоненциальное возрастание
или экспоненциальное убывание

В обычных сетях это не такая проблема… там перемножаются разные матрицы, а здесь одна.

Самая главная проблема RNN – Exploding / Vanishing gradients

В обычных сетях можно просто «отнормировать» темпы обучения в слоях… но тут все веса одинаковые

Собственные значения Якобианов > 1 – Градиенты взрываются (gradients explode)

Собственные значения Якобианов < 1 – Градиенты исчезают (gradients vanish)

Собственные значения случайны – дисперсия нарастает

Самая главная проблема RNN – Exploding / Vanishing gradients

$$h_t = W_{hh}^t h_0$$

Если спектральное разложение…

$$W_{hh} = U \Lambda U^{\mathrm{T}}$$

то

$$h_t = U \Lambda^t U^{\mathrm{T}}$$

тут можно и с транспонированной так делать

Решение проблемы «Exploding gradients»

- **Регуляризация**

знаем…

- **Обрезка градиентов (Clipping gradients)**

было…

- **Метод форсирования учителя (Teacher Forcing)**

было…

- **Ограничение шагов обратного распространения (Truncated Backpropagation Through Time)**

в формуле $\frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial h_{t-2}} \cdots \frac{\partial h_{k+1}}{\partial h_k}$

- **Эхо-сети (Echo State Networks)**

будет

Не учить матрицы переходов…

Решение проблемы «Vanishing gradients»

- **Специальные блоки (Gated self-loops: LSTM, GRU)**

- **Использование методов оптимизации с Гессианом**

Автоматическое масштабирование (первая производная делится на вторую) чаще Momentum заодно – распространение долговременных зависимостей

- **Leaky Integration Units – аналог прокидывания связи**

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

- **Специальная регуляризация (Vanishing Gradient Regularization / Gradient propagation regularizer)**

сложная формула;)

- **Инициализация**

Ex: у орт. матриц все с.з. = 1

Резервуарные вычисления (Reservoir Computing)

Эхо-сети (Echo State Networks)

Метод текучих состояний (Liquid state machines)

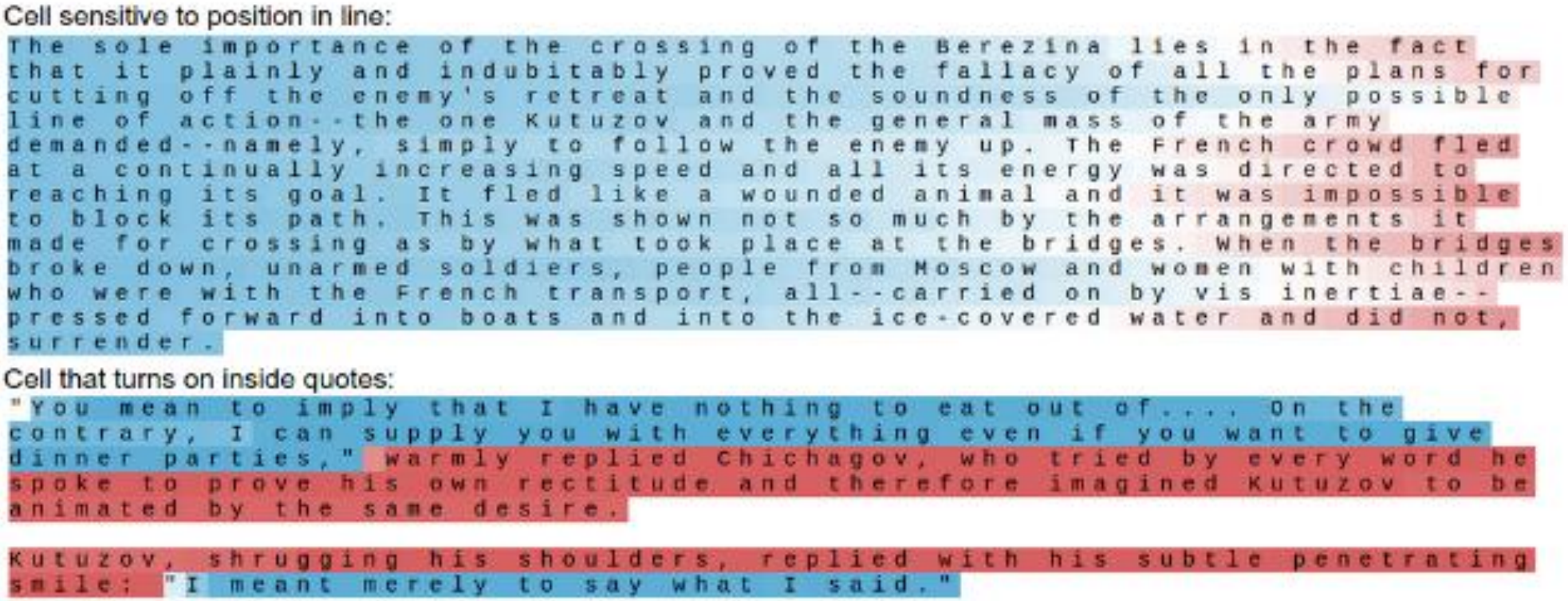
импульсные нейроны с бинарным входом

Задать рекуррентные веса специальным образом,

(чтобы запоминалась история)

обучать только выходные веса

Интерпретация RNN



Отдельные нейроны – «счётчики числа слов в предложении», «индикатор – текст в кавычках»

Karpathy, Johnson, and Fei-Fei: Visualizing and Understanding Recurrent Networks, ICLR Workshop 2016 http://vision.stanford.edu/pdf/KarpathyICLR2016.pdf

Image Captioning

a group of boats sitting on top of a river

a bathroom with a sink and a toilet

a group of people sitting on top of a wooden bench

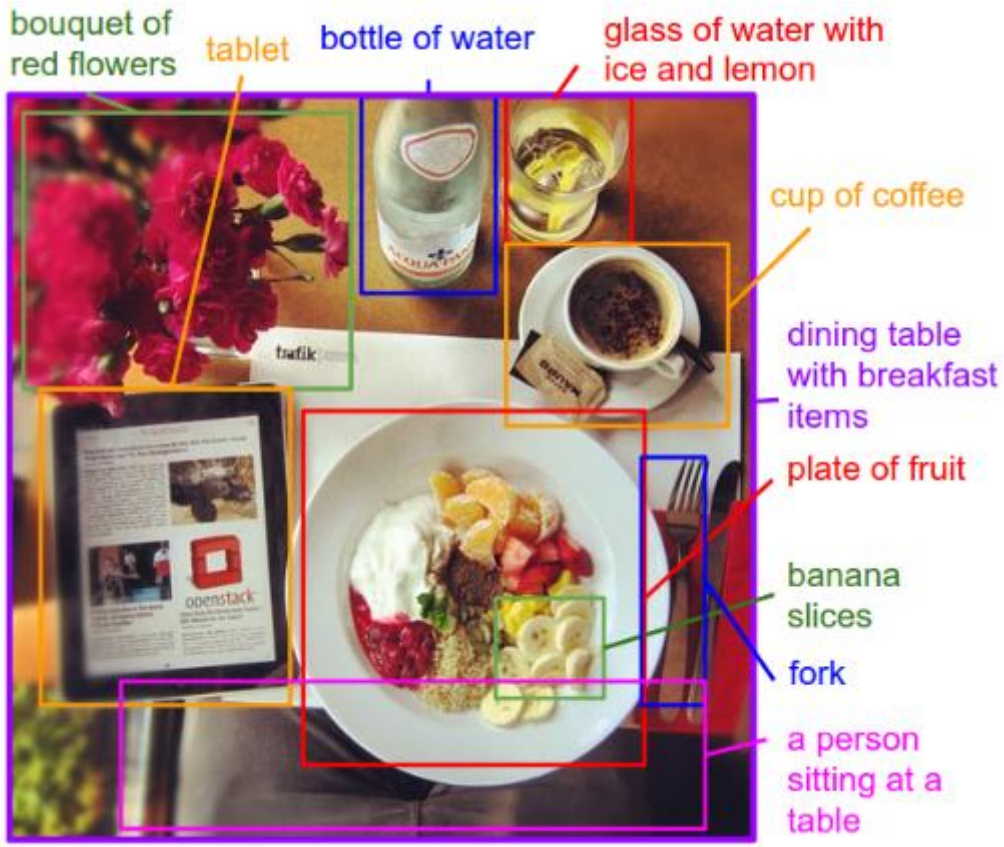
a young man riding a skateboard down the side of a ramp

a man riding skis down a snow covered slope

NEURAL TALK 2 [https://github.com/karpathy/neuraltalk2]

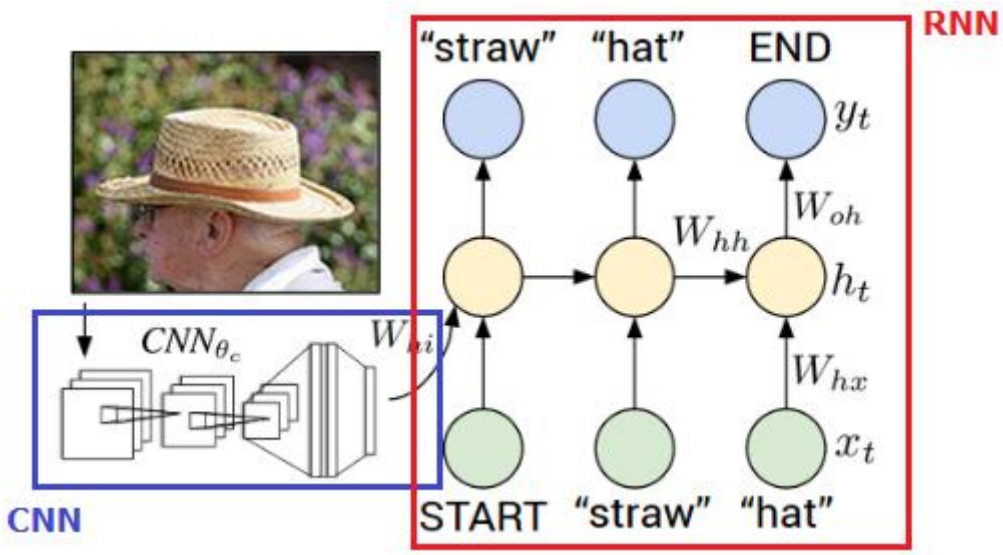
изображение → текст

Image Captioning



изображение (регион) → текст

Image Captioning



Простая идея:

извлечь признаки из CNN → начальное состояние RNN → текст

«Deep Visual-Semantic Alignments for Generating Image Descriptions»
[Karpathy et a, CVPR 2015 https://cs.stanford.edu/people/karpathy/cvpr2015.pdf]

Image Captioning

Более сложная идея:

изображение

→ изображение + 19 регионов на нём (наиболее вероятных)

→ 19 4096-мерных векторов (активации CNN) CNN_i

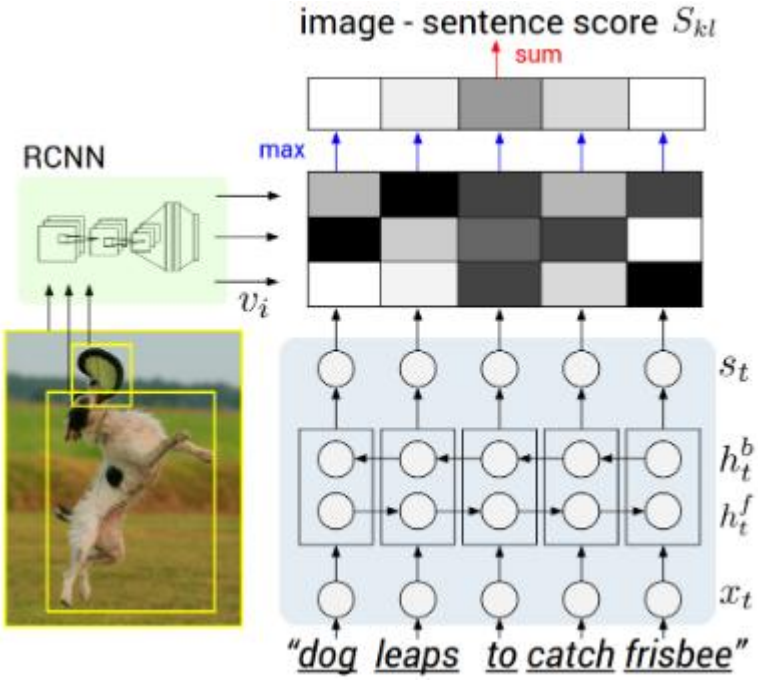
→ 19 h**-мерных векторов** $W_{h\times 4096}\cdot \mathrm{CNN}_i + b$

текст → на вход Bidirectional Recurrent Neural Network (BRNN)

каждое слово → h**-мерный вектор**

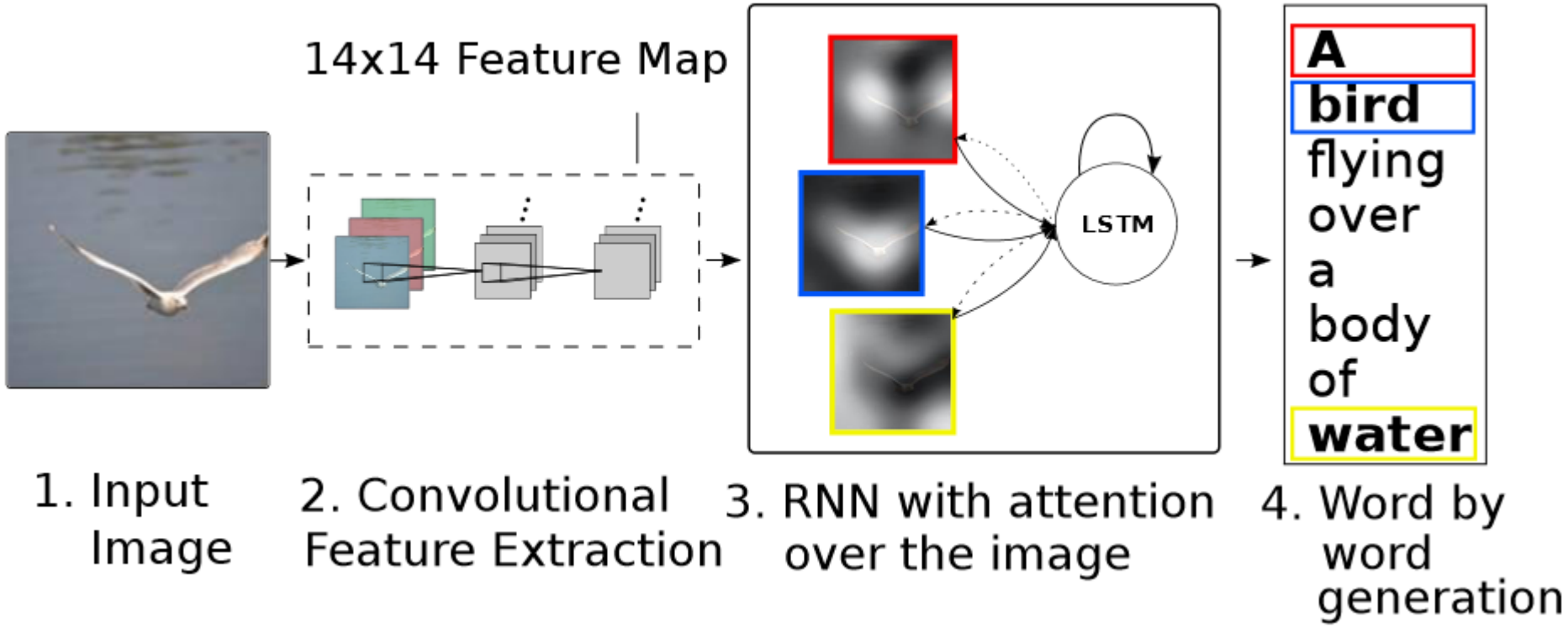
Image Captioning

Более сложная идея:



«Deep Visual-Semantic Alignments for Generating Image Descriptions»
[Karpathy et a, CVPR 2015 https://cs.stanford.edu/people/karpathy/cvpr2015.pdf]

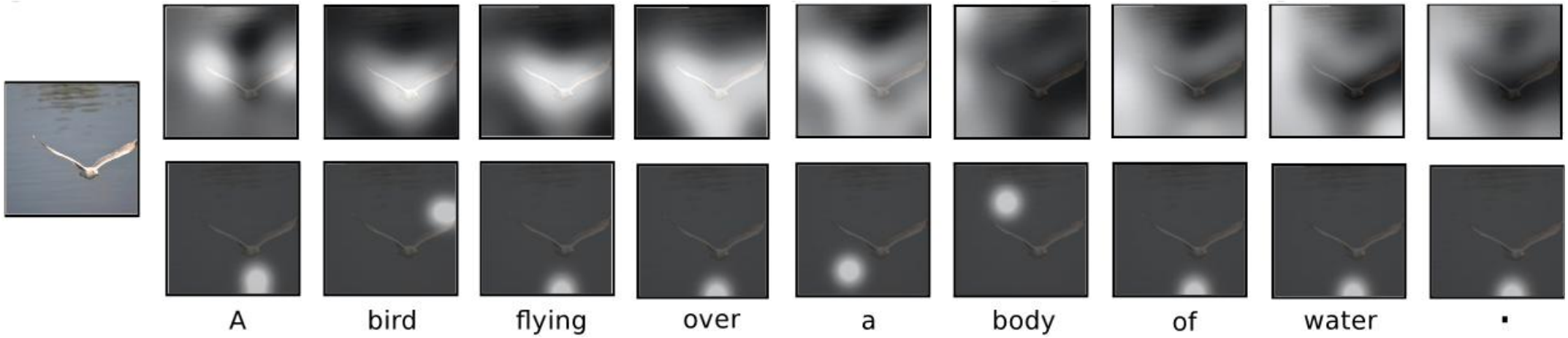
Image Captioning with Attention



«Show, Attend and Tell: Neural Image Caption Generation with Visual Attention» [Kelvin Xu и др. 2016 https://arxiv.org/abs/1502.03044]

Image Captioning with Attention

Как распределяется внимание при генерации очередного слова



Верхний ряд – большая область внимания, нижний – маленькая

Заодно инструмент «что модель видит»… – и почему ошибается…

A large white bird standing in a forest.

A woman holding a clock in her hand.

Image Captioning with Attention

Идея:

изображение → (с помощью CNN) набор векторов (каждый описывает свою часть изображения)

Использовали признаки со свёрточной части (а не полносвязной), зато «видно» какой части изображения соответствуют признаки

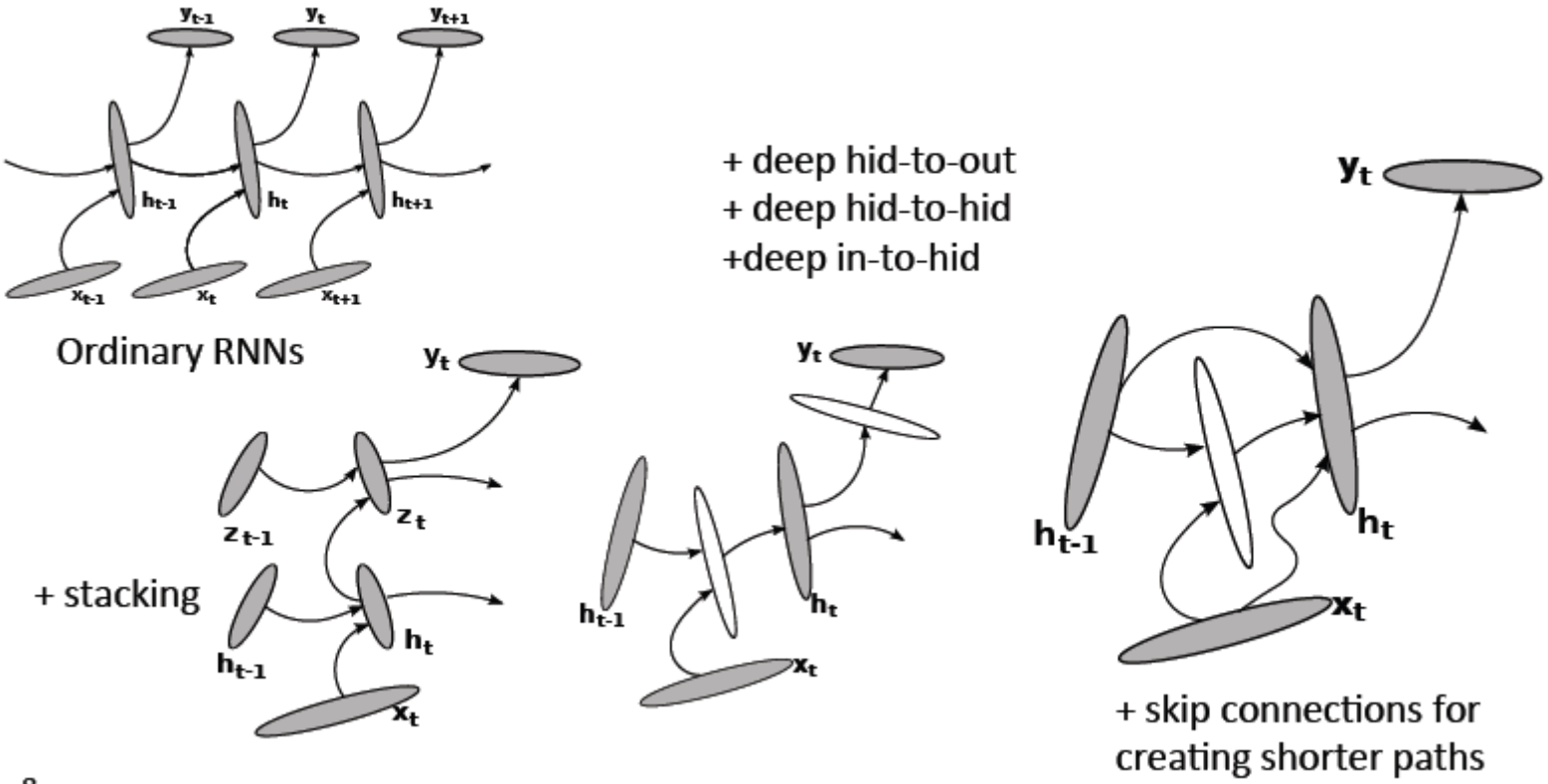
LSTM для генерации текста

Про лучевой поиск

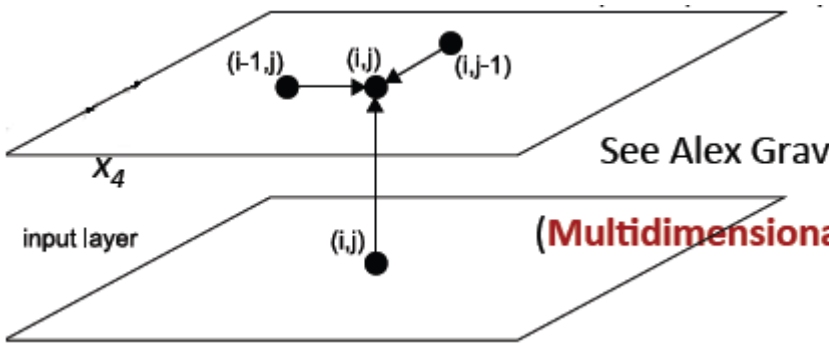
рассказать без слайдов

Как строить глубокие RNN

(пример – вариантов много!)



Многомерные RNN

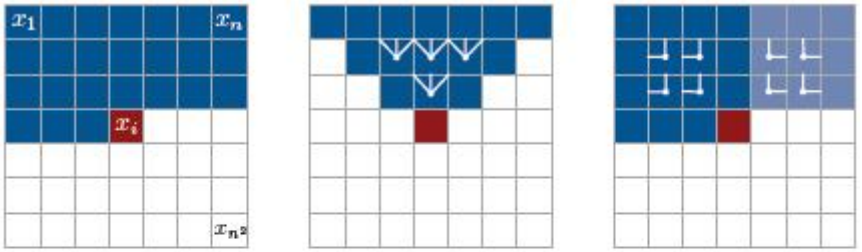


See Alex Graves's work, e.g., 2012

(**Multidimensional** RNNs, Graves et al 2007)

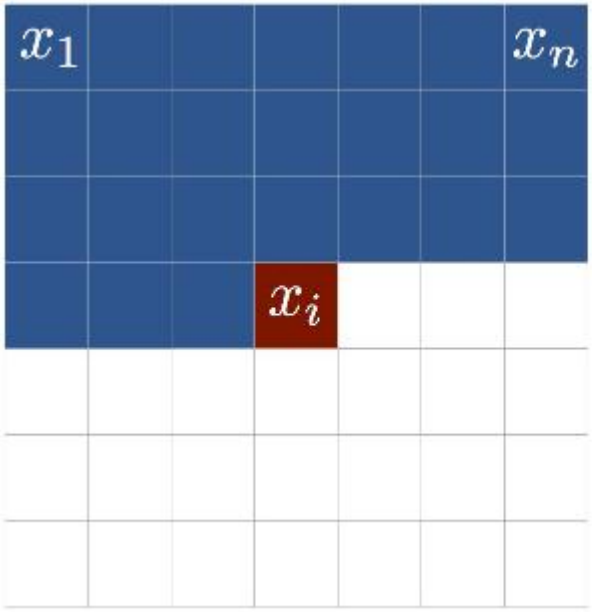
Пиксельные RNN

хорошо учат текстуру



van den Oord (DeepMind) et al ICML 2016, best paper

Пиксельные RNN



$$p(\mathbf{x}) = \prod_{i=1}^{n^2} p(x_i | x_1, \ldots, x_{i-1})$$

MI (Multiplicative Integration)

$$\varphi(\alpha \circ Wx \circ Uz + \beta_1 \circ Wx + \beta_2 \circ Uz + b)$$

(произведение адамарово), вместо

$$\varphi(Wx + Uz + b)$$

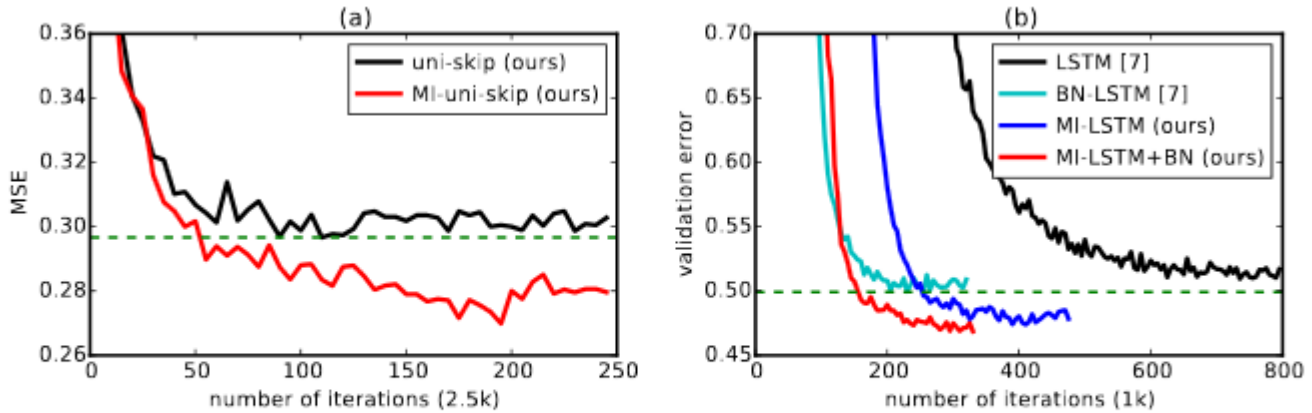


Figure 2: (a) MSE curves of uni-skip (ours) and MI-uni-skip (ours) on semantic relatedness task on SICK dataset. MI-uni-skip significantly outperforms baseline uni-skip. (b) Validation error curves on attentive reader models. There is a clear margin between models with and without MI.

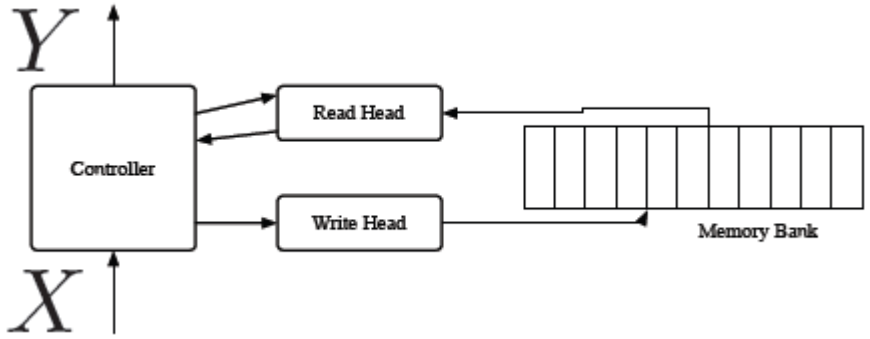
Yuhuai Wu, Saizheng Zhang, Ying Zhang, Yoshua Bengio, Ruslan Salakhutdinov «On Multiplicative Integration with Recurrent Neural Networks», 2016 // https://arxiv.org/pdf/1606.06630.pdf

Механизмы внимания (что-то есть в текстах…)

- **Neural Turing Machines (Graves et al 2014)**

- **Memory Networks (Weston et al 2014) будет в текстах**
  - **Fully Supervised MemNNs**
  - **End2End MemNNs**
  - **Key-Value MemNNs**
  - **Dynamic MemNNs**

- **Content-based attention mechanism (Bahdanau et al 2014) to control the read and write access into a memory**



Дифференцируемые структуры памяти (Differentiable Memory structures)

– **LSTM** [Hochreiter & Schmidhuber]

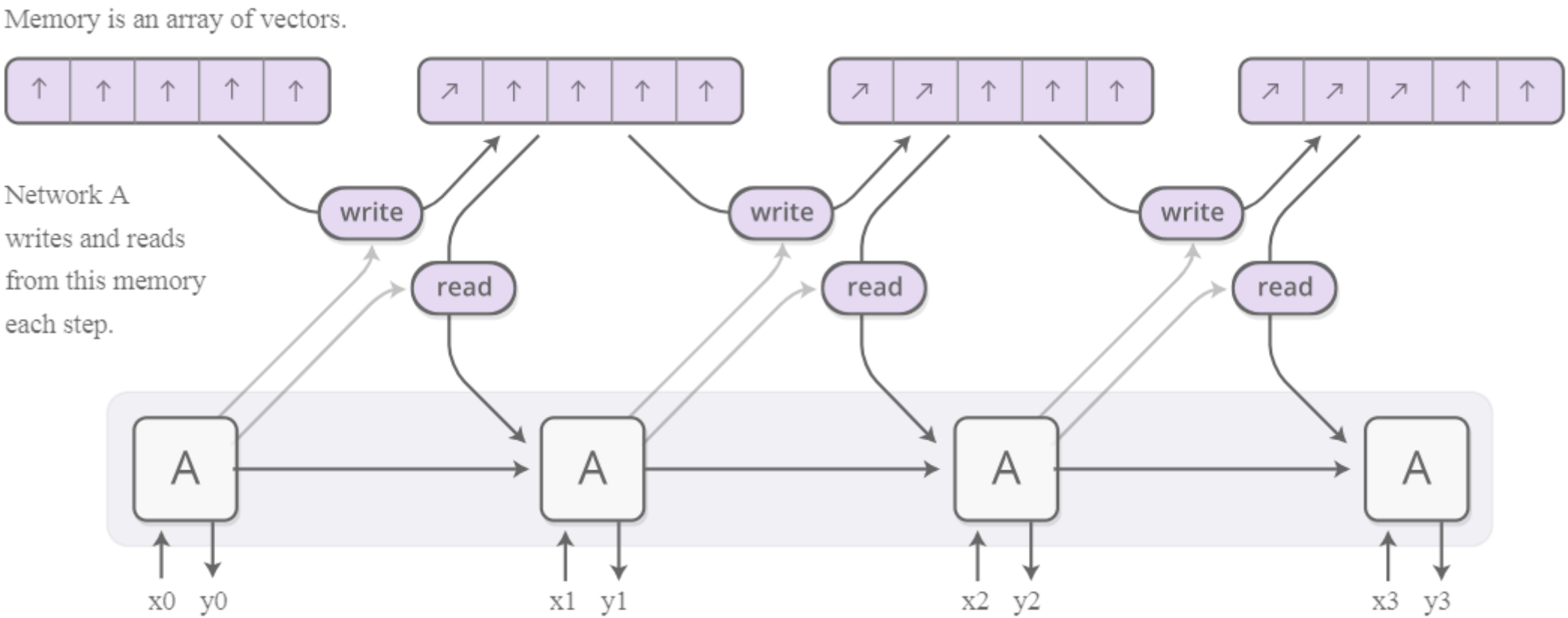
– **Tapes** [NTM, Graves et al'14]

– **Arrays** [Memory Nets, Weston et al'14]

– **Stacks** [Joulin & Mikolov'15]

Важна дифференцируемость для обучения…

Neural Turing Machines

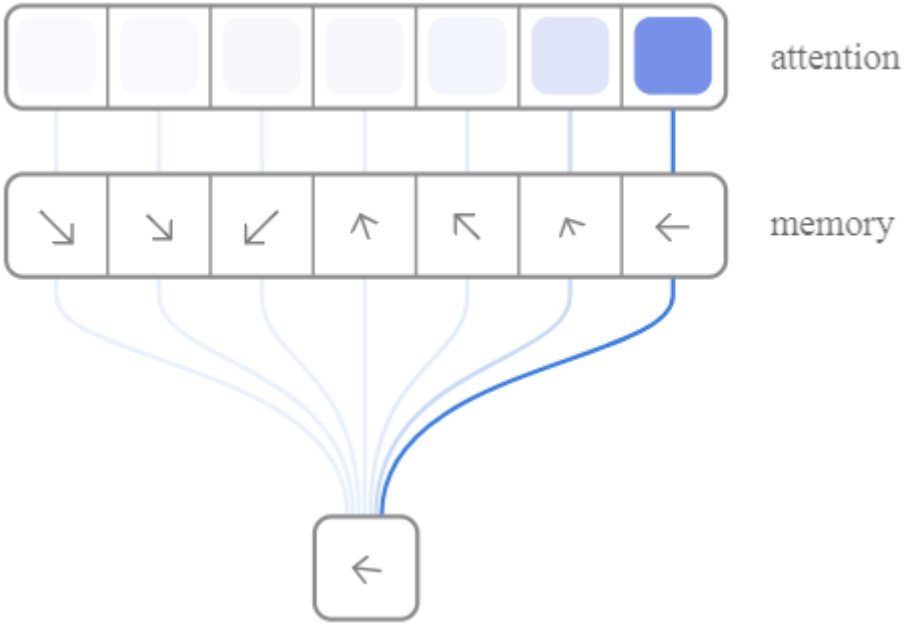


https://distill.pub/2016/augmented-rnns/#neural-turing-machines

A. Graves, G. Wayne, I. Danihelka «Neural Turing Machines», 2014 // https://arxiv.org/abs/1410.5401

Neural Turing Machines

читаем взвешенную сумму памяти
это нужно, в том числе, чтобы всё было дифференцируемо



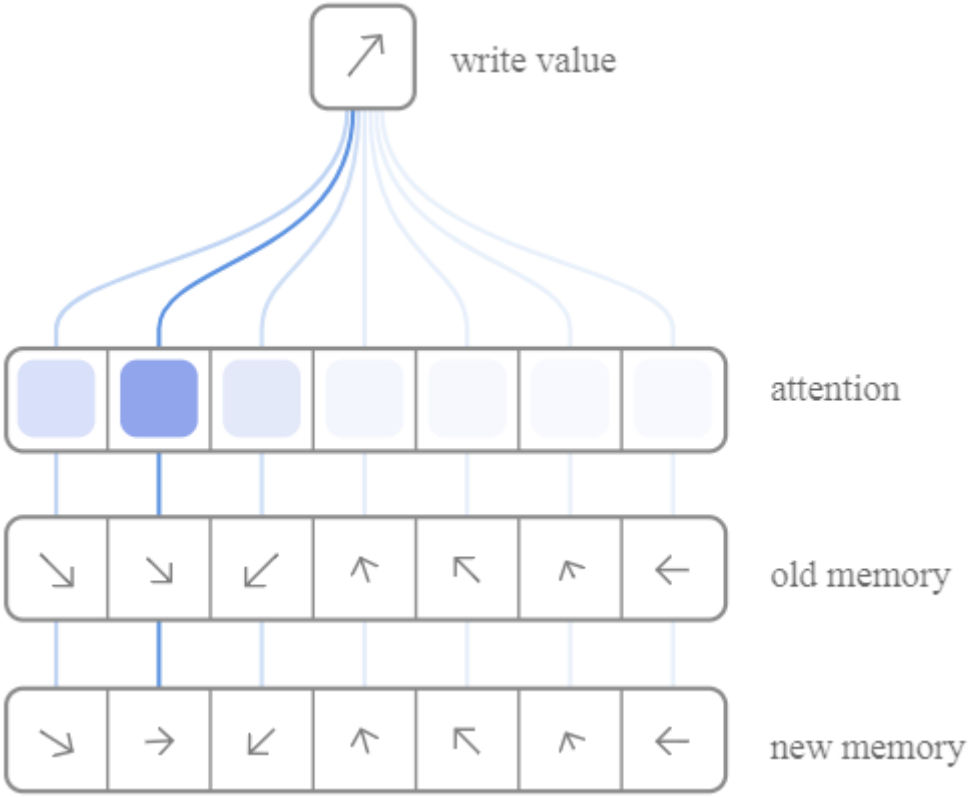
$$r = \sum_i a_i M_i$$

Коэффициенты регулируются с помощью «attention»

«Soft-attention reading»

Neural Turing Machines

Аналогично пишем в память



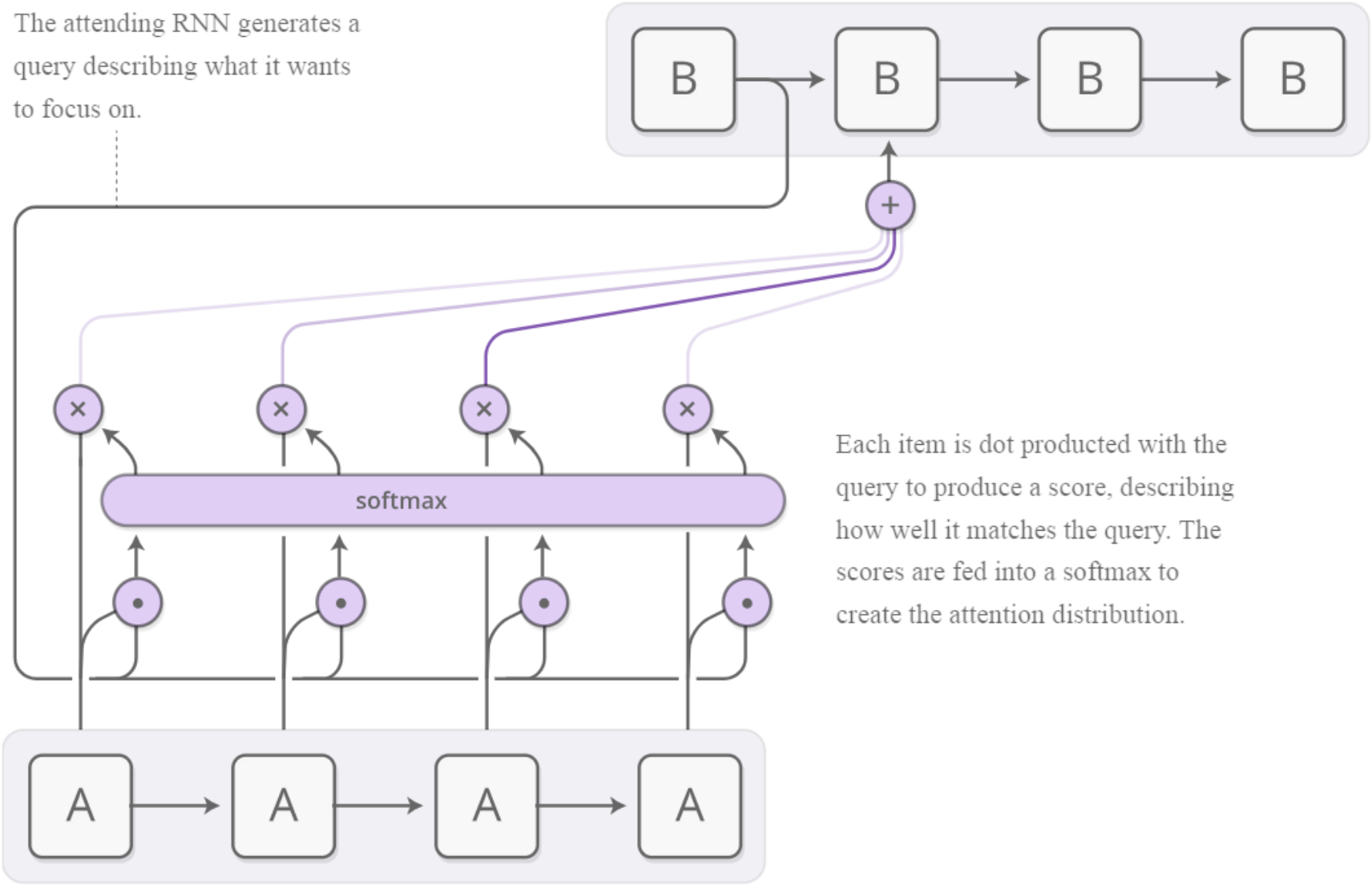
Пишем в каждую ячейку, но в «каком количестве» зависит от «attention»

$$M_i \equiv a_i w + (1 - a_i) M_i$$

Внимание Attentional Interfaces

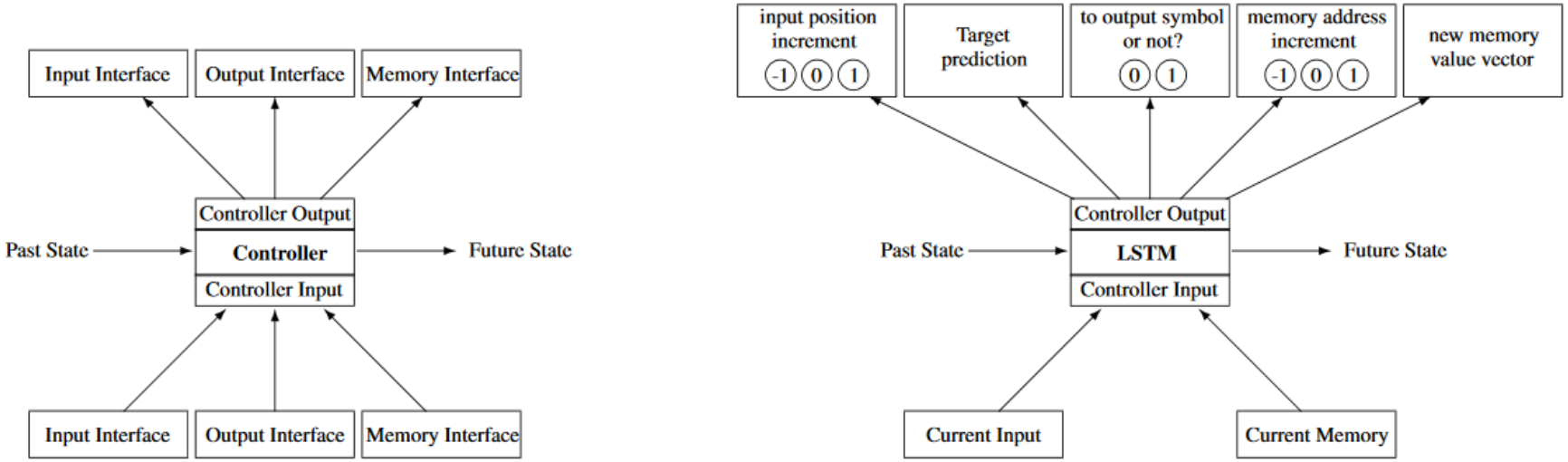
опять же смотрим сразу на все выходы другой RNN

специальная сеть указывает, на чём фокусироваться



Discrete Read/Write: Reinforcement Learning Neural Turing Machines

RL для обучения сети, которая взаимодействует с дискретными структурами



An abstract Interface–Controller model

Our model as an Interface–Controller

Figure 1: **(Left)** The Interface–Controller abstraction, **(Right)** an instantiation of our model as an Interface–Controller. The bottom boxes are the read methods, and the top are the write methods. The RL–NTM makes discrete decisions regarding the move over the input tape, the memory tape, and whether to make a prediction at a given timestep. During training, the model's prediction is compared with the desired output, and is used to train the model when the RL-NTM chooses to advance its position on the output tape; otherwise it is ignored. The memory value vector is a vector of content that is stored in the memory cell.

Wojciech Zaremba, Ilya Sutskever «Reinforcement Learning Neural Turing Machines - Revised», 2016 // https://arxiv.org/abs/1505.00521

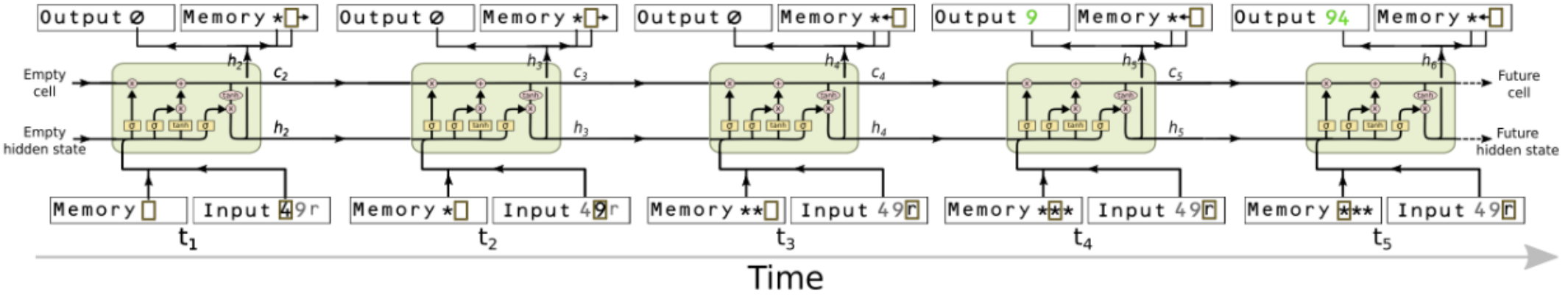
Discrete Read/Write: Reinforcement Learning Neural Turing Machines

Вход LSTM:

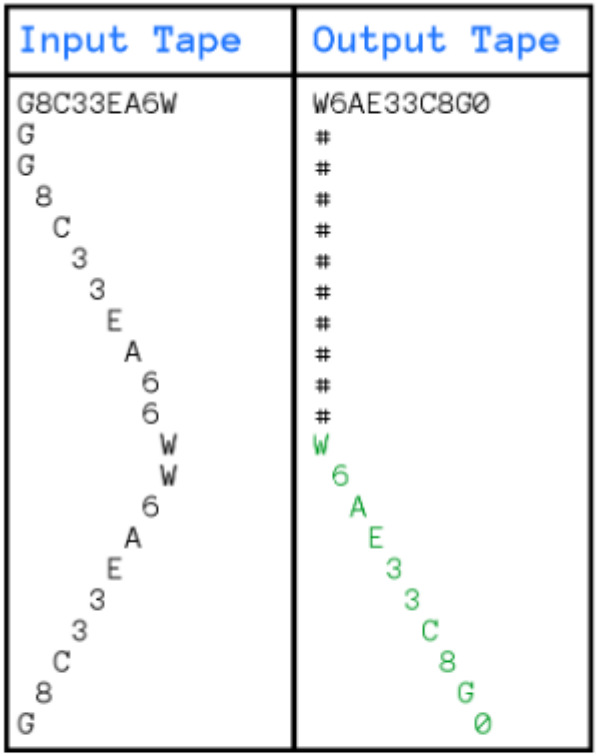
- вход (символ/ы с ленты)
- текущая ячейка/и памяти
- текущая память ячейки
- представление всех предыдущих действий (не изображено)

Выход LSTM:

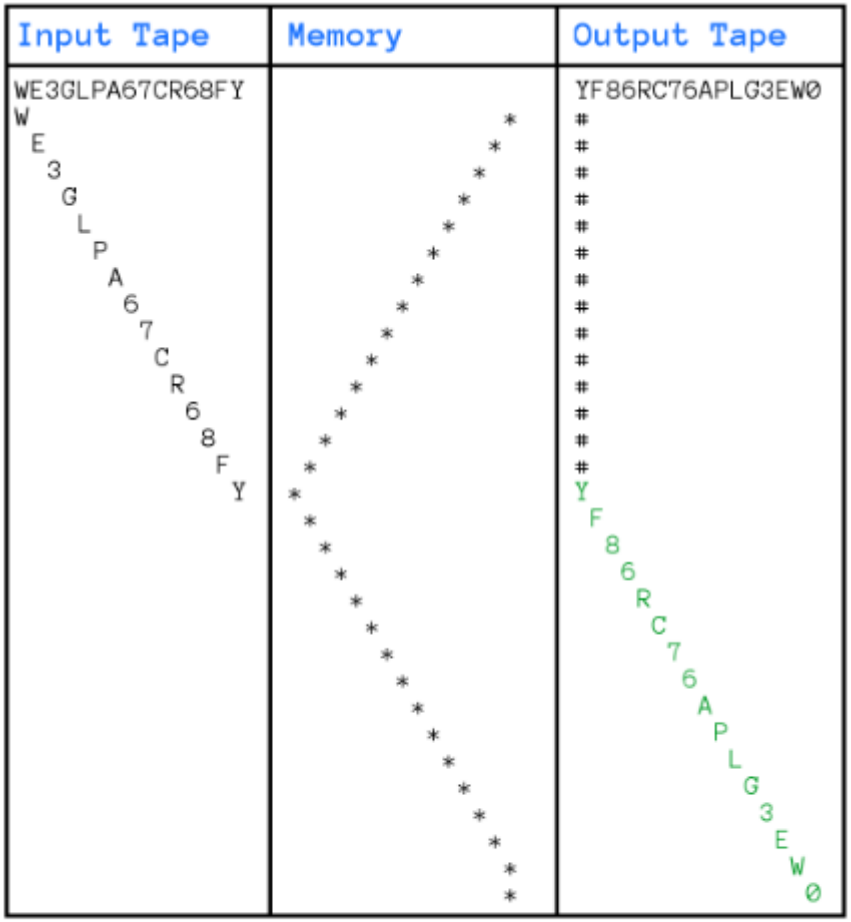
- выход (предсказание)
- значение для текущей ячейки памяти
- текущая память ячейки
- решение о смене ячейки памяти (←,→), позиции на ленте и т.п.



Discrete Read/Write: Reinforcement Learning Neural Turing Machines



An RL-NTM successfully solving a small instance of the Reverse problem (where the external memory is not used).



An RL-NTM successfully solving a small instance of the ForwardReverse problem, where the external memory is used.

Discrete Read/Write: Trainable memory addressing scheme

dynamic neural Turing machine (D-NTM)

каждая ячейка памяти = (контент, адрес)

2 главных модуля D-NTM

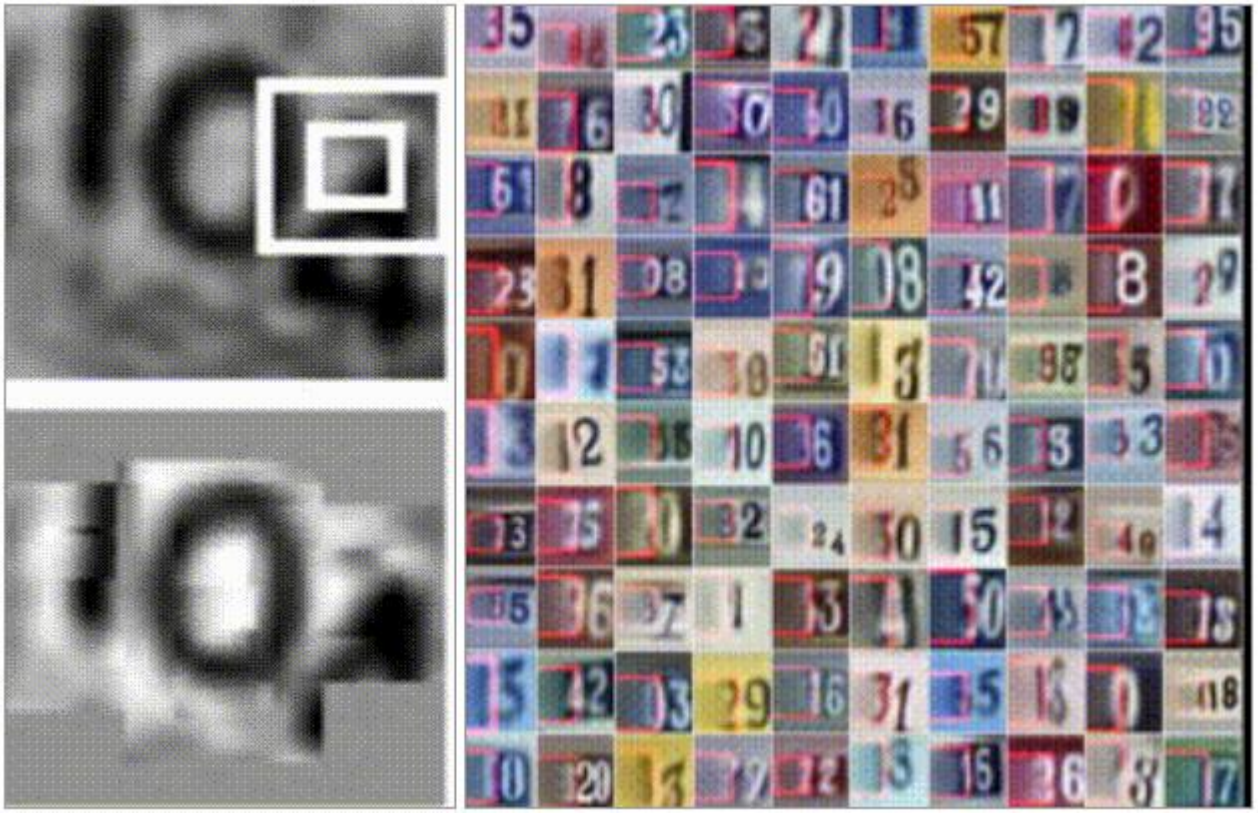
контроллер **память**

RNN ~ даёт команды памяти

Caglar Gulcehre, Sarath Chandar, Kyunghyun Cho, Yoshua Bengio «Dynamic Neural Turing Machine with Soft and Hard Addressing Schemes», 2016 // https://arxiv.org/abs/1607.00036

Применение RNN

Не только в задачах, где в явном виде даны последовательности



а где можно переформулировать задачу в нужном виде

https://arxiv.org/abs/1412.7755

https://arxiv.org/abs/1502.04623

Применение RNN

NLP/Speech

speech to text

http://proceedings.mlr.press/v32/graves14.pdf

machine translation

https://arxiv.org/abs/1409.3215

handwritten text generation

http://www.cs.toronto.edu/~graves/handwriting.html

Computer Vision

frame-level video classification

https://arxiv.org/abs/1411.4389

image captioning

https://arxiv.org/abs/1411.4555

video captioning

https://arxiv.org/abs/1505.00487

visual question answering

https://arxiv.org/abs/1505.02074

Применение RNN

Language Modeling

– предсказать следующее слово

Text Generation

– генерация предложений по начальной информации (ex: несколько слов)

Ссылки

deeplearningbook

https://www.deeplearningbook.org/

Блог DeepGrid «Organic Deep Learning»

http://www.jefkine.com/general/2018/05/21/2018-05-21-vanishing-and-exploding-gradient-problems/

Блог «Machine Learning Research Should Be Clear, Dynamic and Vivid»

https://distill.pub/