ТЕХНОСФЕРА

Лекция 9
Рекуррентные нейронные сети

Нестеров Павел. Храбров Кузьма

3 апреля 2017 г.

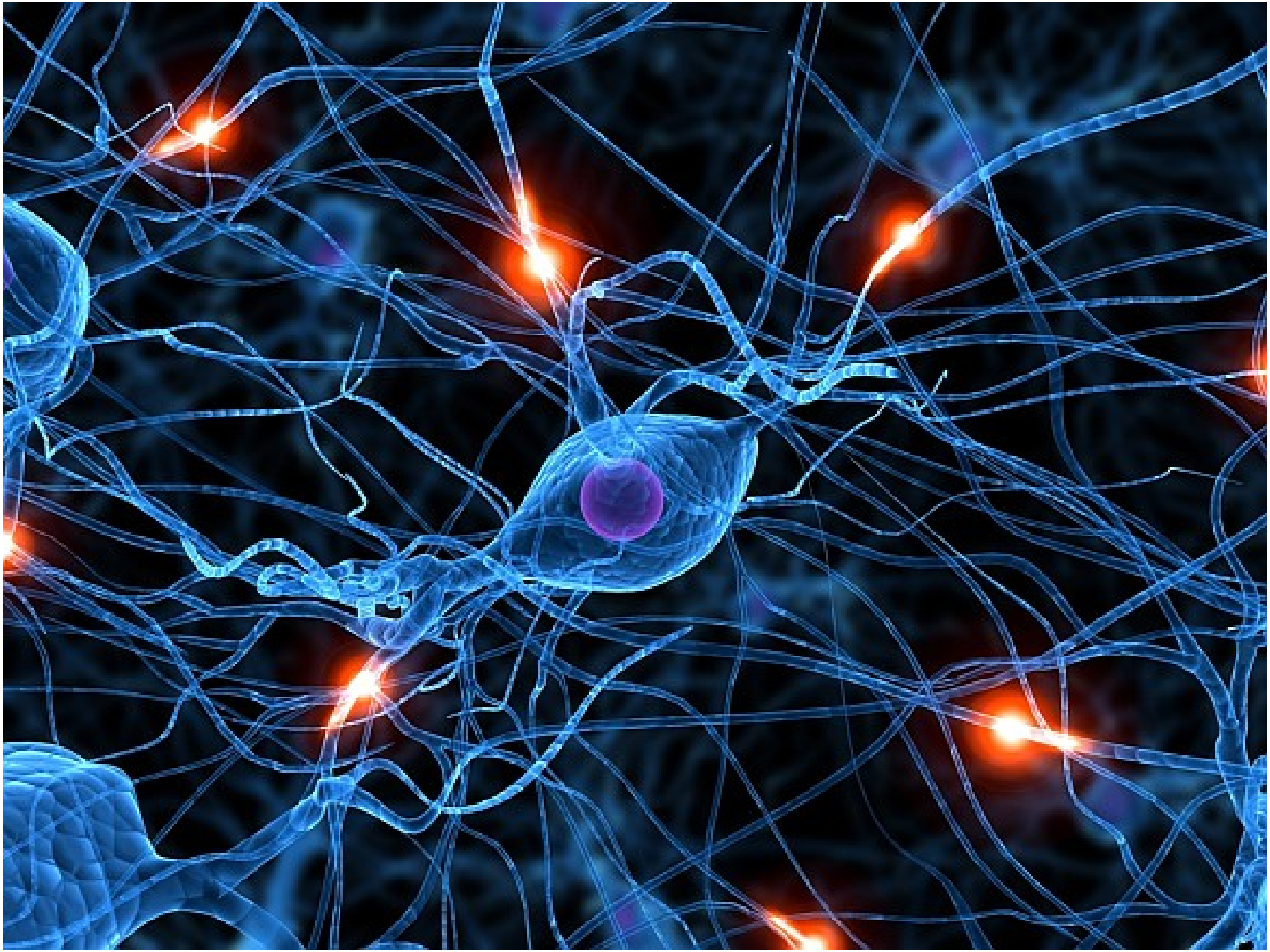
План лекции

Предпосылки

Backpropagation through time

Развитие RNN

Биологическая нейронная сеть



Искусственная сеть прямого распространения

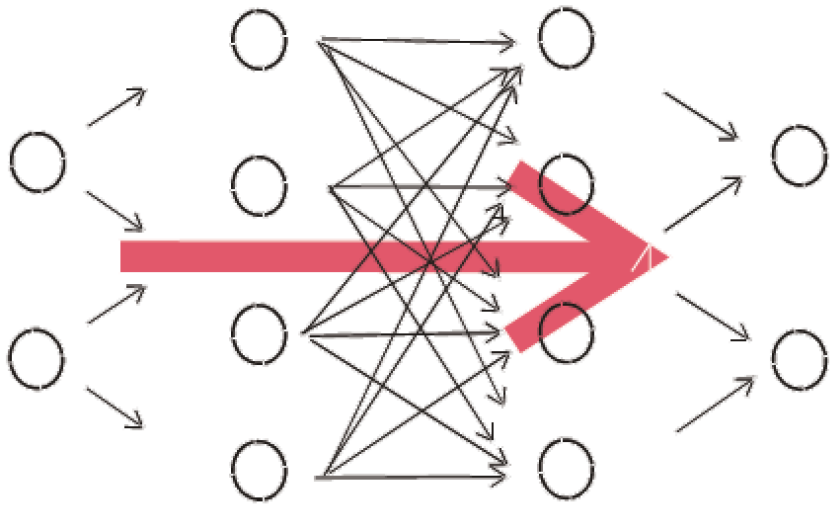


Рис.: Модель многослойной сети прямого распространения

- аппроксимирует любую функцию
- 95% публикаций именно об этой модели
- до последнего времени в основном только они применялись на практике

Искусственная рекуррентная нейронная сеть

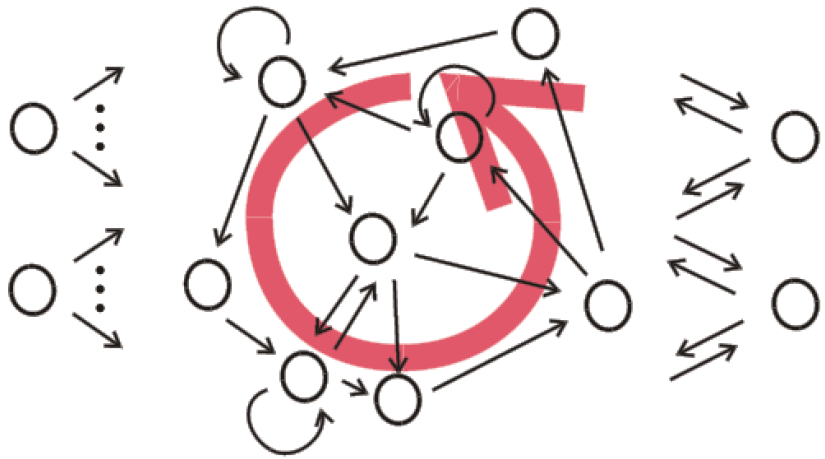


Рис.: Модель рекуррентной сети[1]

- все биологические сети - рекуррентные
- RNN моделирует динамическую систему
- существует несколько алгоритмов обучения без явного лидера
- только недавно стали использоваться на практике

[1] http://minds.jacobs-university.de/sites/default/files/uploads/papers/ESNTutorialRev.pdf

Универсальная теорема аппроксимации

- MLP: аппроксимирует любую функцию;
- RNN: аппроксимирует поведение любой динамической системы [2];
- RNN: все машины Тьюринга могут быть смоделированы полносвязной рекуррентной нейронной сетью с сигмоидальной функцией активации[3].

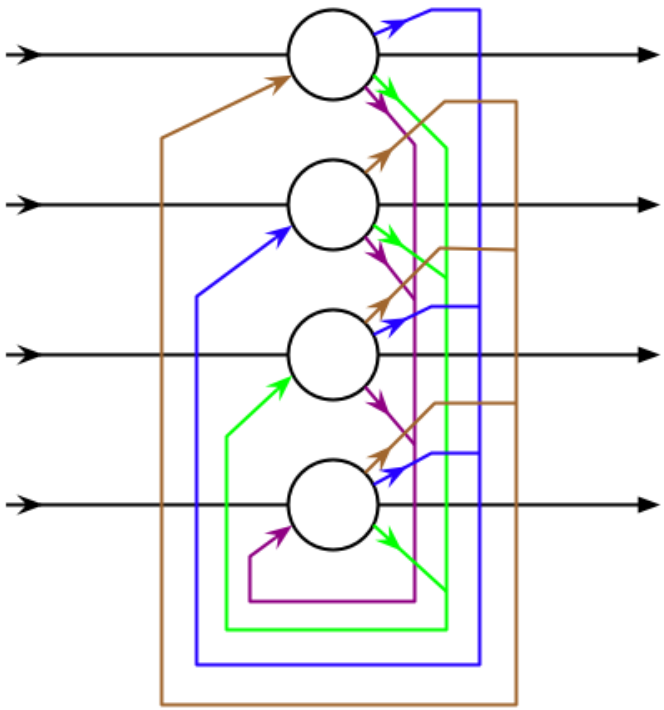
Таким образом: тренировка многослойного персептрона - это оптимизация функций, а тренировка рекуррентной сети - это оптимизация программ.

---

[2] `http://minds.jacobs-university.de/sites/default/files/uploads/papers/ESNTutorialRev.pdf`

[3] Siegelmann & Sontag, 1991, Applied Mathematics Letters, vol 4, pp 77-80.

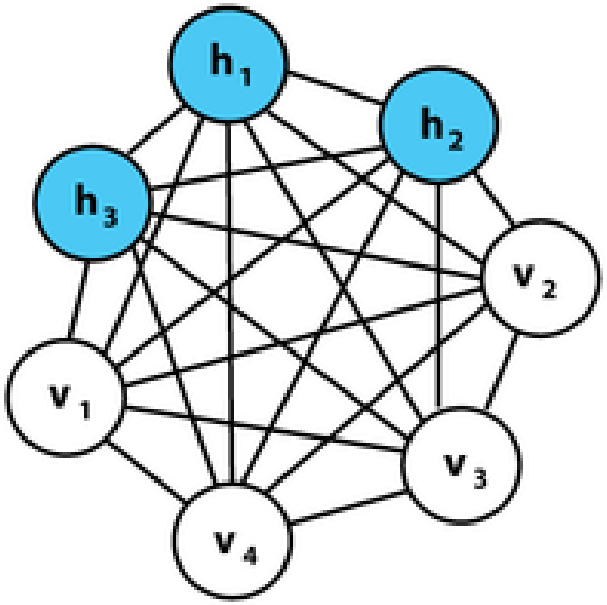
Нейросеть Хопфилда



- ассоциативная память
- обратная связь
- пороговая функция активации

Такая сеть (рекуррентная нейронная сеть) может находится как в стабильном состоянии, осциллировать, или даже проявлять признаки детерминированного хаоса.
Хопфилд показал, что при симметричной матрице весов, существует такая функция энергии бинарных состояний системы, что при симуляции система эволюционирует в одно из низко-энергетических состояний.

Машина Больцмана - стохастический генеративный вариант сети Хопфилда



- энергия не изменилась: $E = -\sum_i s_i b_i - \sum_{i<j} s_i s_j w_{ij}$
- симметричная матрица весов $w_{ij} = w_{ji}$, но нет обратных связей: $w_{ii} = 0$
- появились скрытые состояния (система ищет такую конфигурацию скрытых состояний которая лучшим образом описывает видимые состояния)

- $\forall i : s_i \in \{0, 1\}$
- стохастический нейрон
- цель: научиться описывать видимые переменные $\vec{v}$ с помощью скрытых $\vec{h}$ (*напоминает автоенкодер?*)

Моделирование последовательностей

- преобразование последовательности одной природы в последовательность другой природы
  - последовательность звуков в последовательность слов
- предсказание следующего члена последовательности (граница между обучением с учителем и без учителя становится все тоньше, вспомним хотя бы автоенкодеры)
  - временные ряды
  - изображения: прогнозирование следующего пикселя на основе окружения (смотрим house generate.gif)
  - видео: следующий кадр на основе предыдущих
  - текст: генерация следующего слова

Способы: авторегрессионная модель

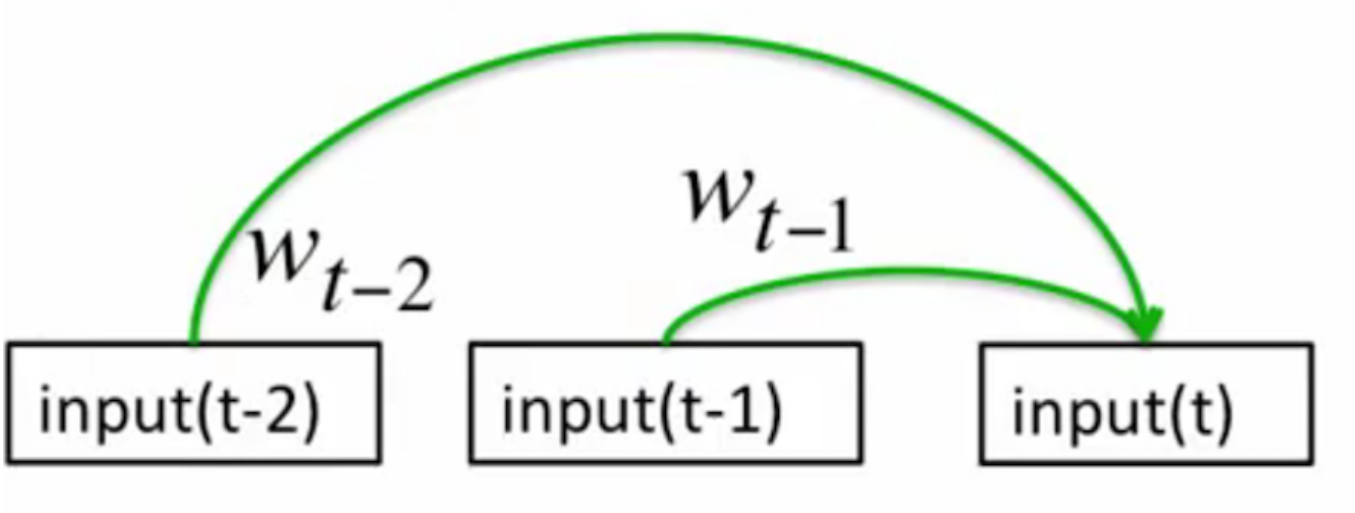
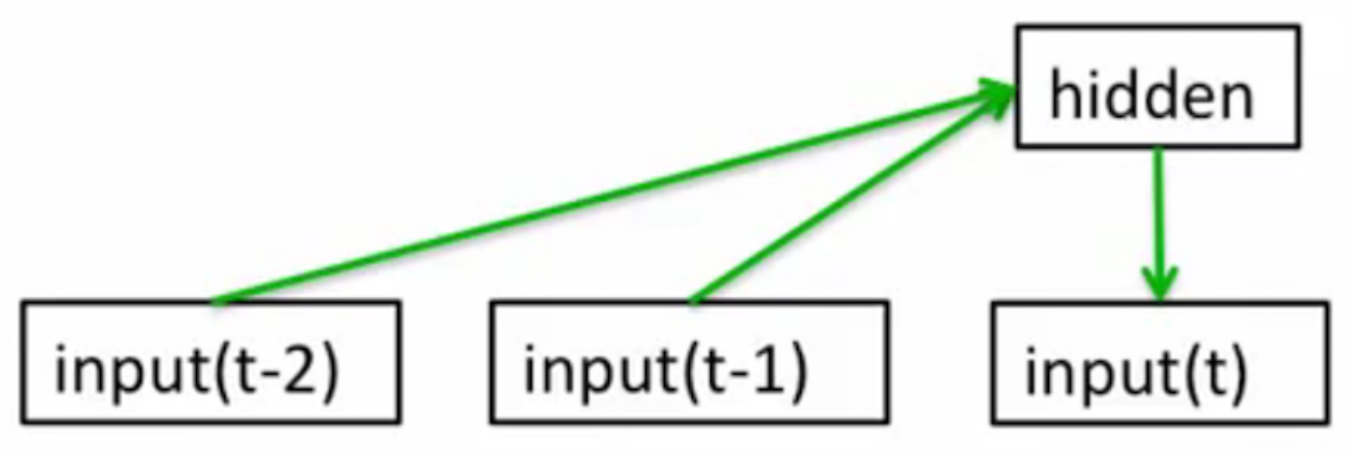
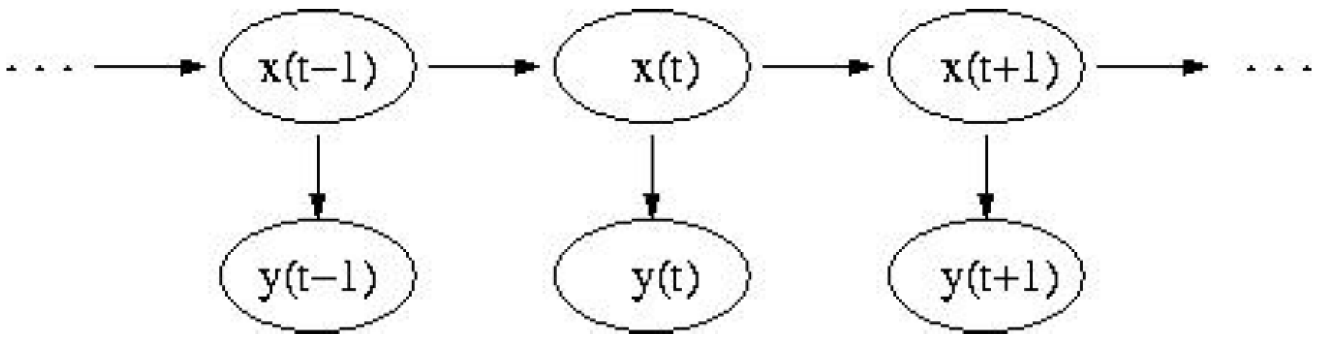


Рис.: Модель авторегрессии

Способы: MLP



Способы: скрытые модели Маркова



Backpropagation through time, #1

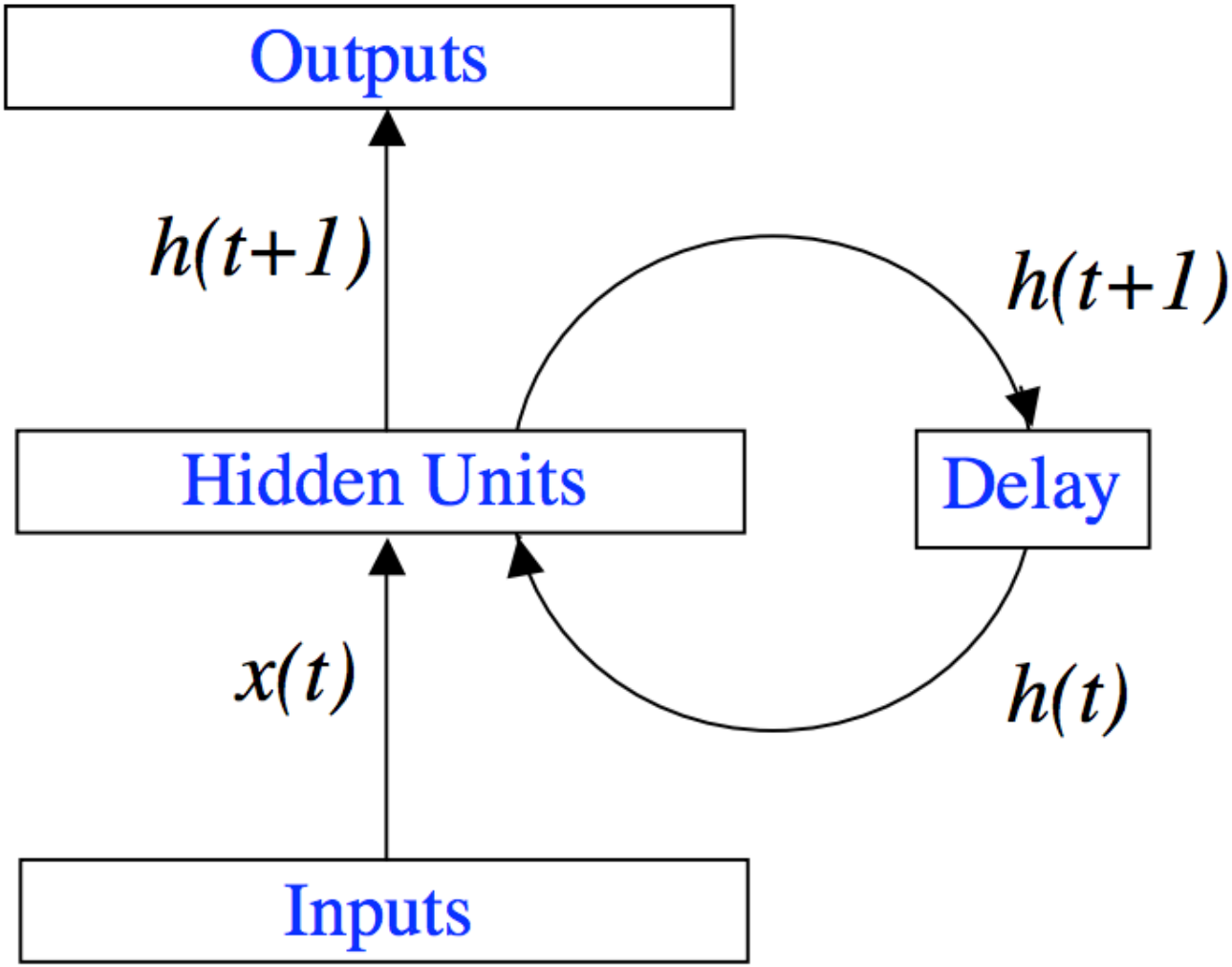


Рис.: RNN с задержкой на скрытом слое[4]

[4] http://www.cs.bham.ac.uk/~jxb/INC/l12.pdf

Backpropagation through time, #2

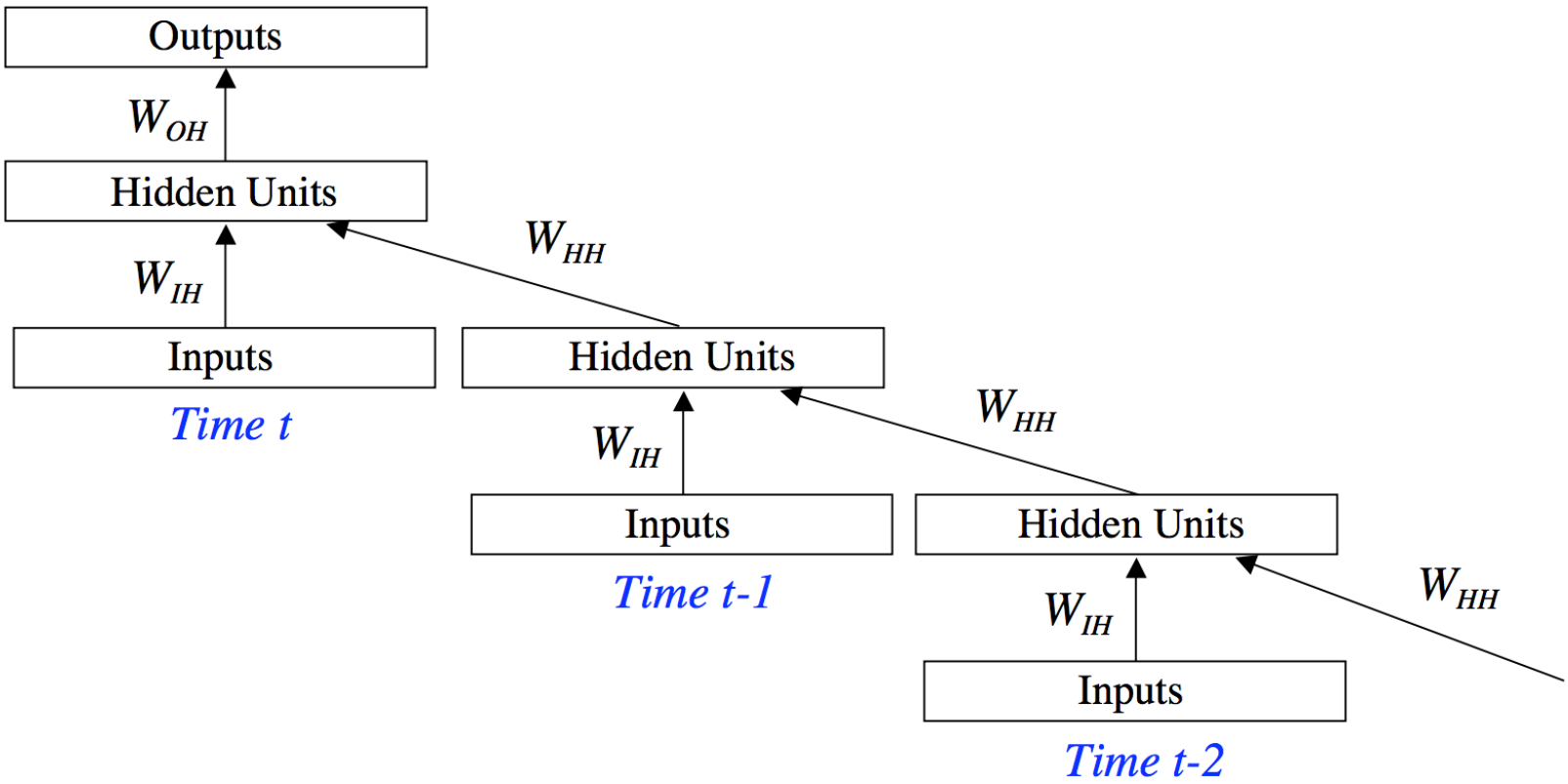


Рис.: RNN unfolding[5]

- *а что делать со значениями $t - n$, когда есть только первый член последовательности?*

[5] `http://www.cs.bham.ac.uk/~jxb/INC/l12.pdf`

Weights sharing

Алгоритм backprop легко модифицируется так, что бы можно было наложить любые линейные ограничения на веса. Допустим мы хотим, что бы $w_1 = w_2$:

- $w_1 = w_2 \Rightarrow \Delta w_1 = \Delta w_2 \Rightarrow \dfrac{\partial E}{\partial w_1} = \dfrac{\partial E}{\partial w_2}$
- $\Delta w_1^{\text{new}} = \Delta w_2^{\text{new}} = \dfrac{\partial E}{\partial w_1} + \dfrac{\partial E}{\partial w_2}$

Направления исследований RNN

- 1958, Розенблатт: персептрон с обратной связью, но после статьи Минского про него как то забыли;
- 1978, Хопфилд: энергетическая интерпретация сетей с обратной связью;
- 1986, Майкл Джордан: рекуррентная нейросеть с единичной задержкой;
- 1990, Джеф Элман: апгрейдит сеть Джордана и внедряет на практике;
- 1997, Ёрген Шмидтхубер: long short term memory[6]
- 2005+, все: глубокие сети и мультимодальное обучение

[6] http://en.wikipedia.org/wiki/Long_short_term_memory

Нейронная сеть Джордана

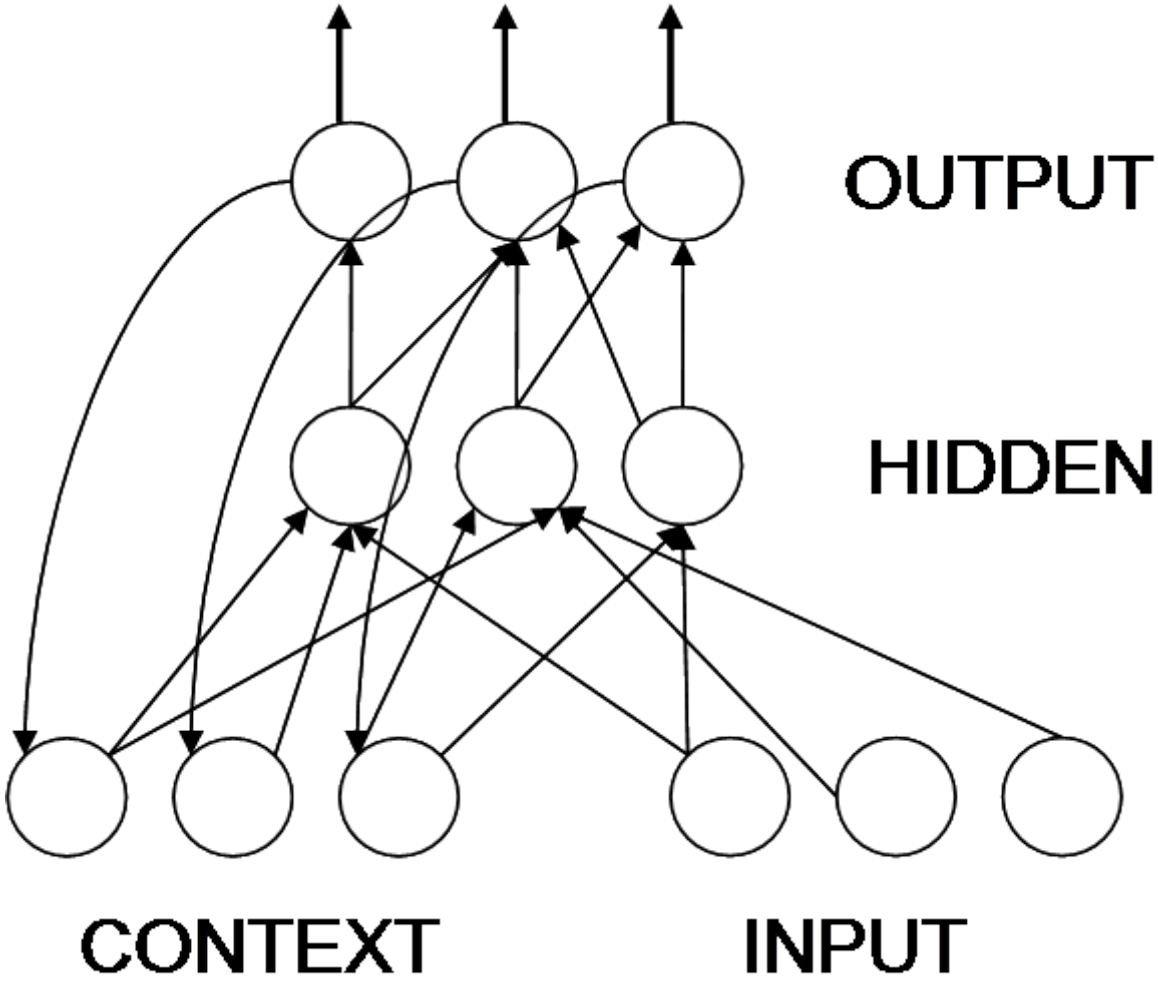


Рис.: Модель сети Джордана[7]

[7] http://www.cogsci.ucsd.edu/research/documents/technical/TR-8604.pdf

Нейронная сеть Элмана

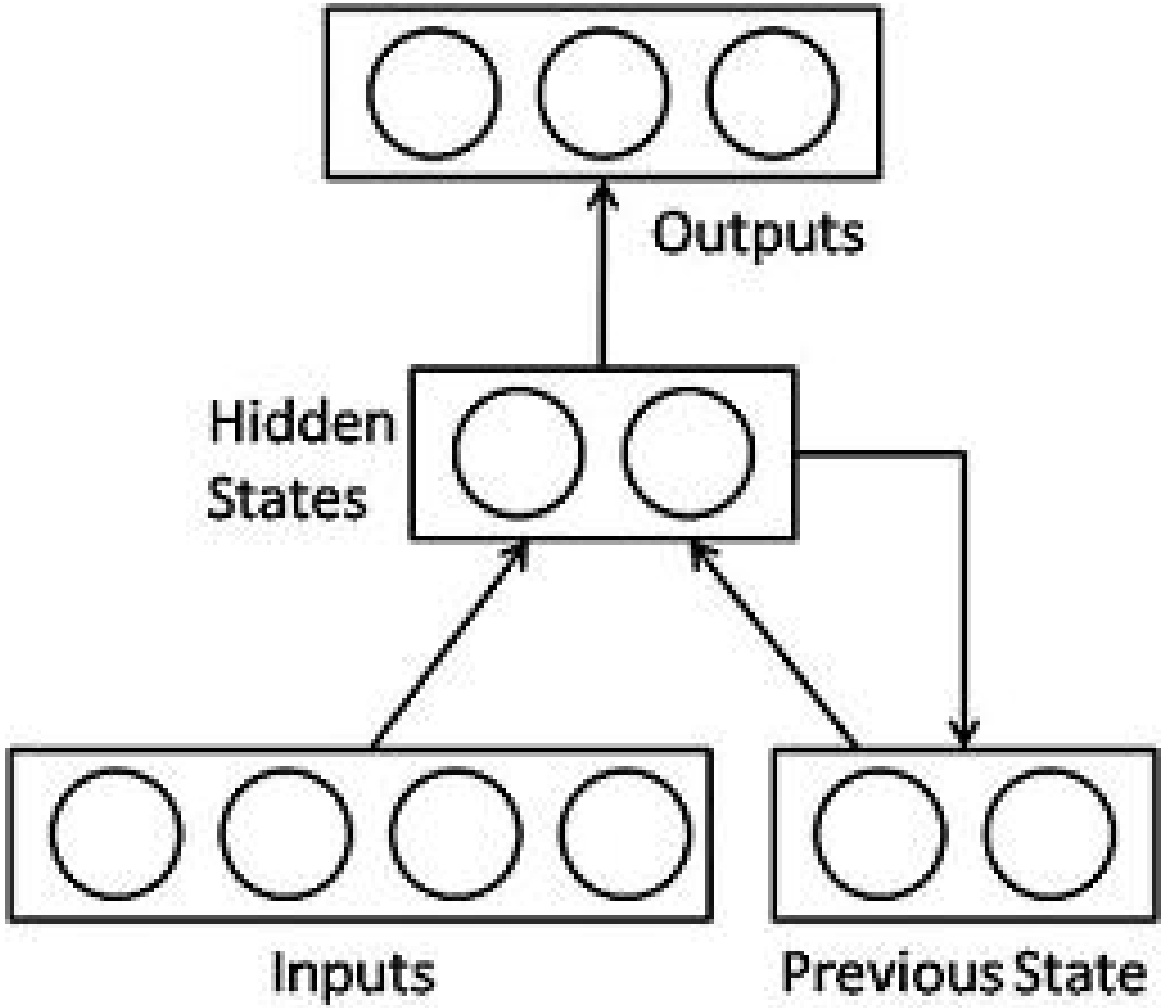
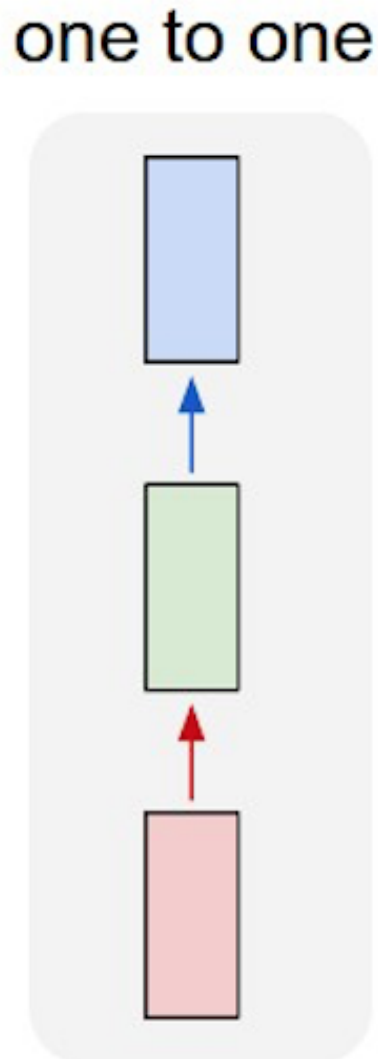


Рис.: Модель сети Элмана[8]

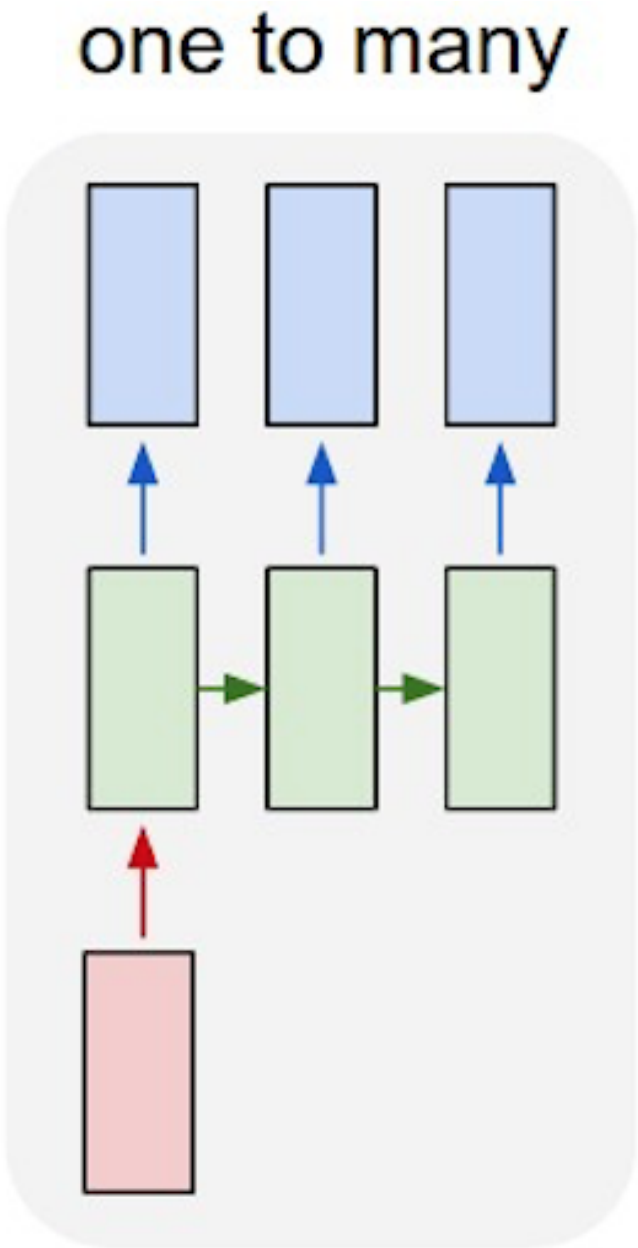
[8] http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=87F767D3C21027CEF557110E2867E556?doi=10.1.1.117.1928&rep=rep1&type=pdf

Обычный MLP



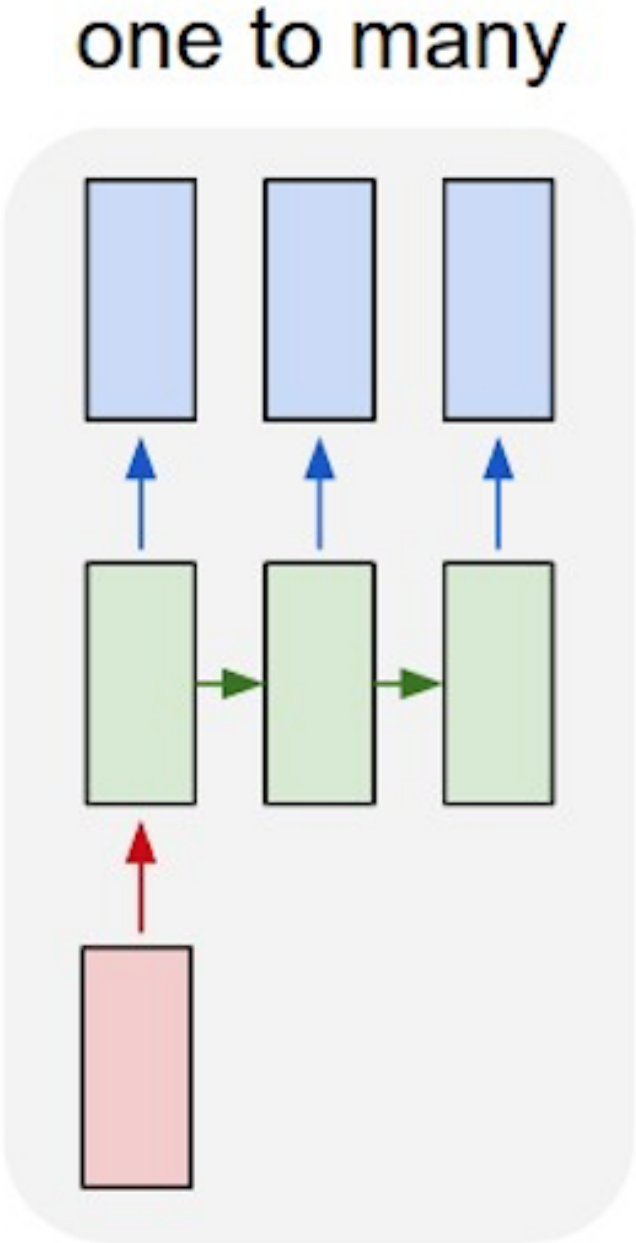
используется для отображения одного вектора/примера в другой, например для классификации

RNN один ко многим, #1



для чего?

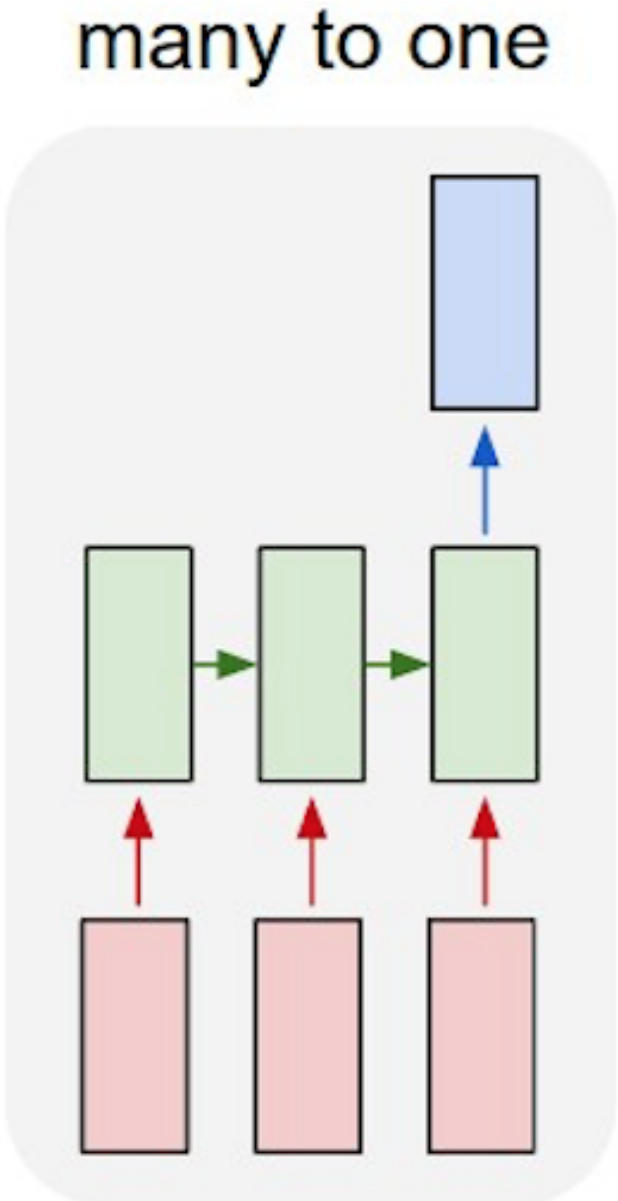
RNN один ко многим, #2



отображение одного примера в последовательность

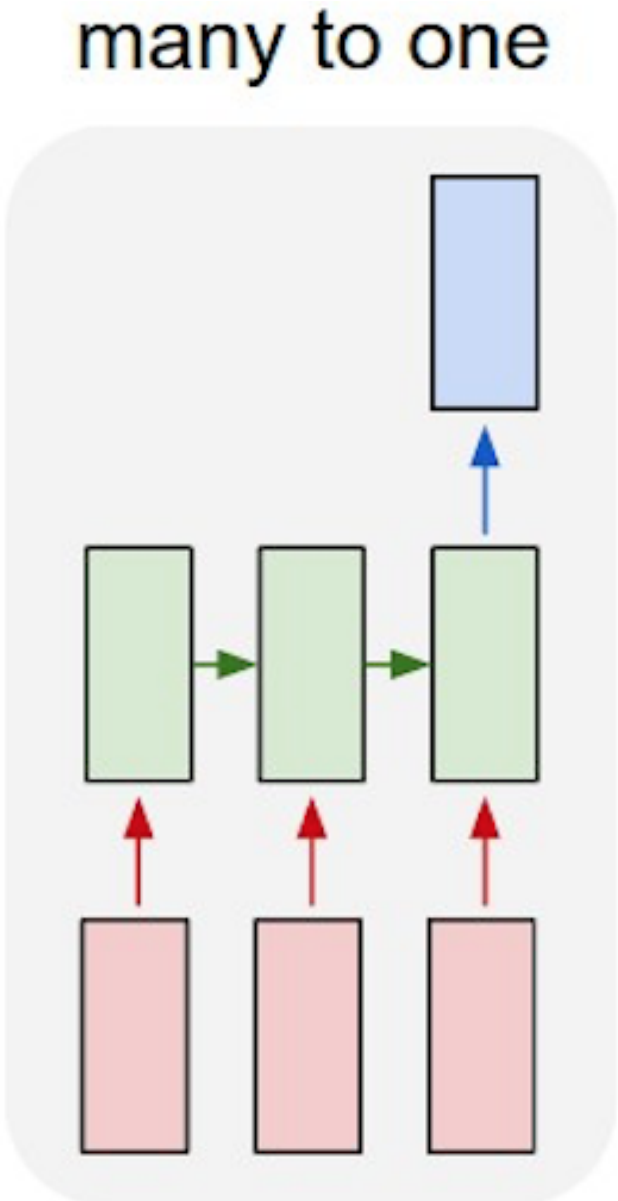
- описание картинки естественным языком
- генерация музыки по стилю

RNN многие к одному, #1



для чего?

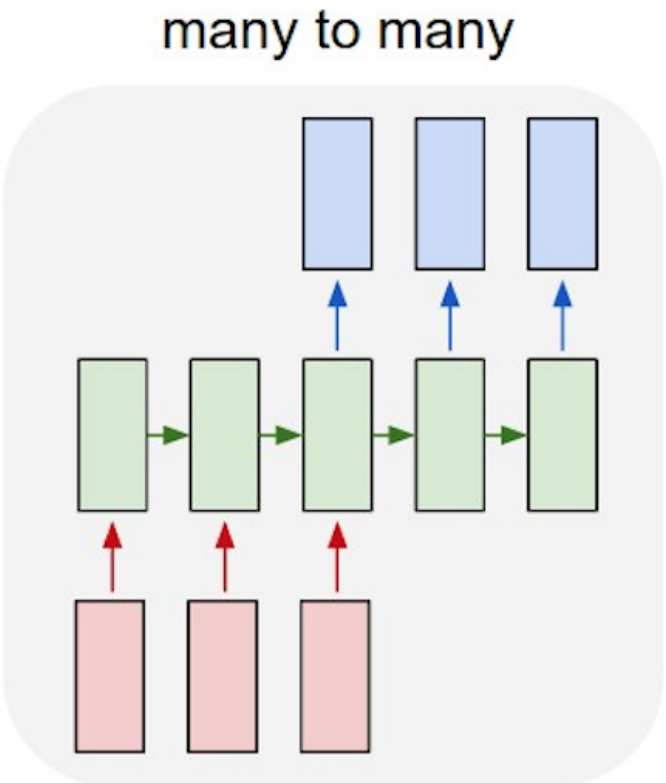
RNN многие к одному, #2



последовательность в пример

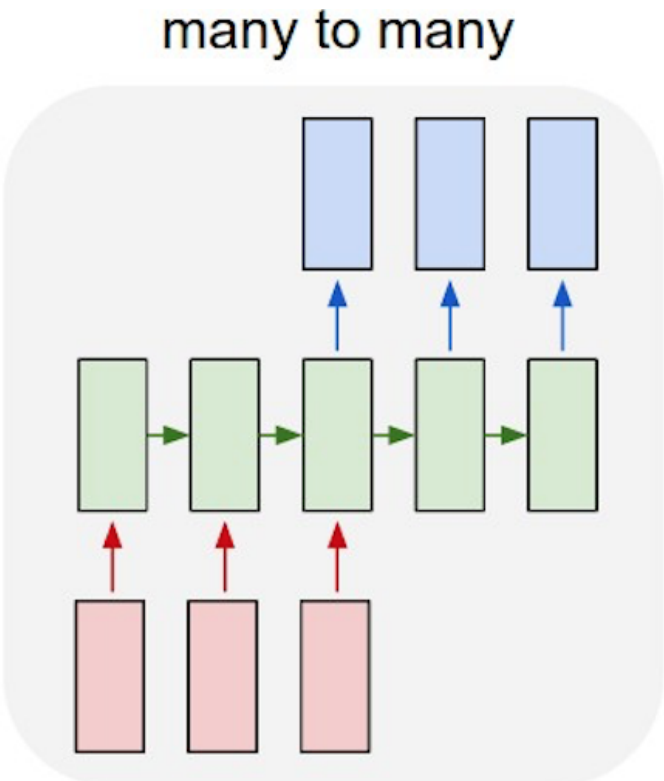
- определение тональности текста
- определение стиля изображения

RNN многие ко многим асинхронно, #1



для чего?

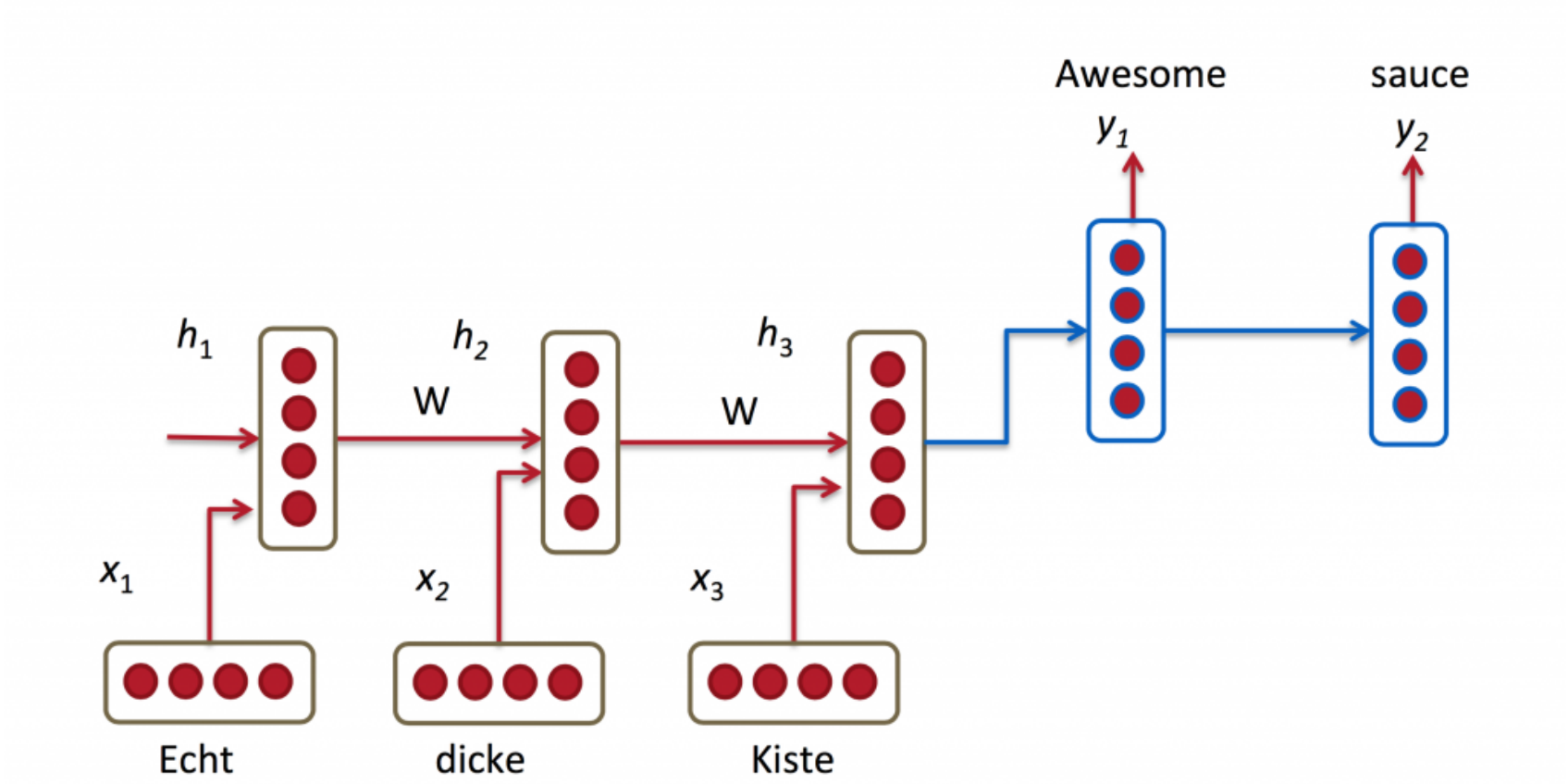
RNN многие ко многим асинхронно, #2



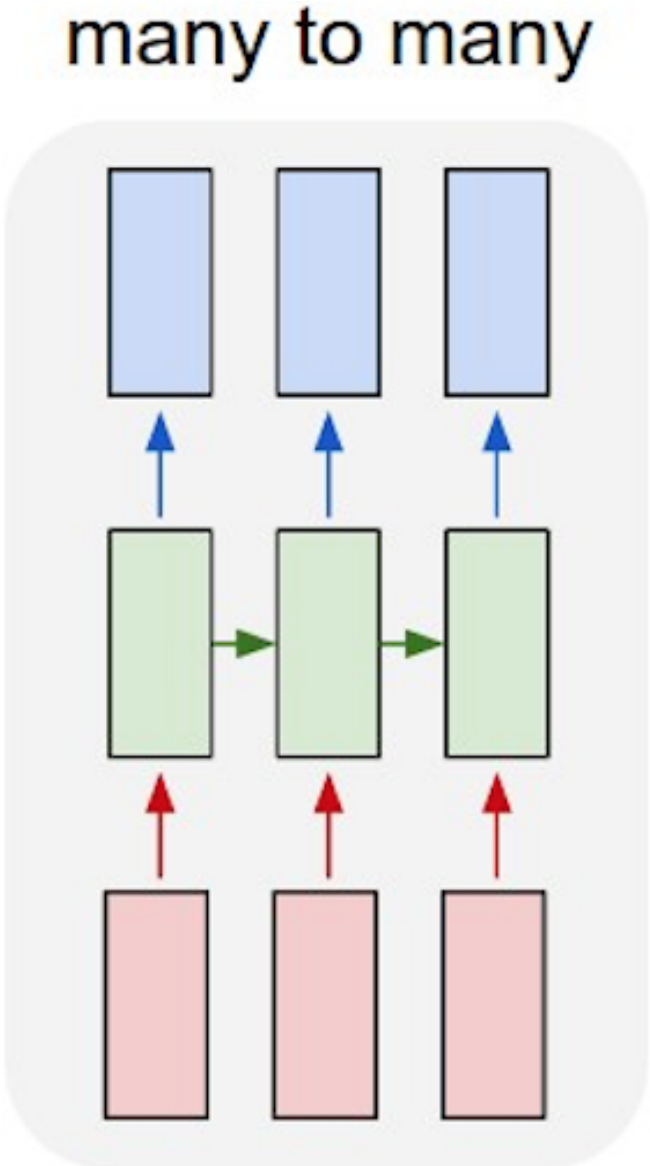
последовательность в последовательность

- перевод текста (*зачем асинхронность?*)

RNN многие ко многим асинхронно, #3

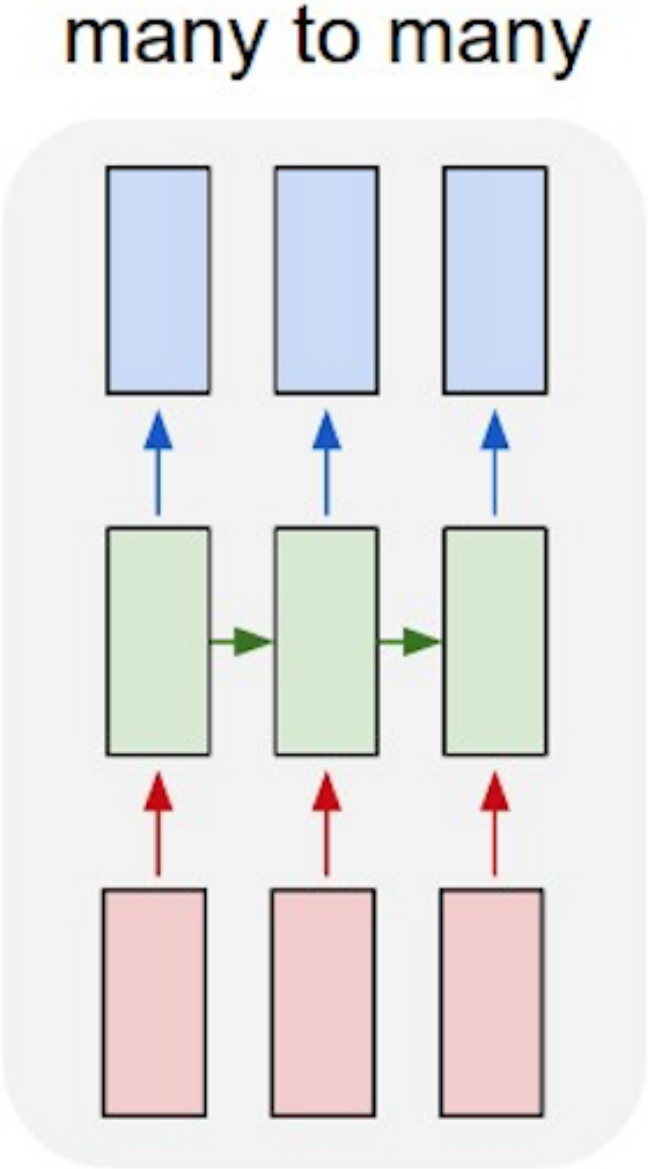


RNN многие ко многим синхронно, #1



для чего?

RNN многие ко многим синхронно, #2



последовательность в последовательность

- описать каждый кадр видео

Смотрим результат генерации текстов тут
`http://karpathy.github.io/2015/05/21/rnn-effectiveness/`

Deep RNN

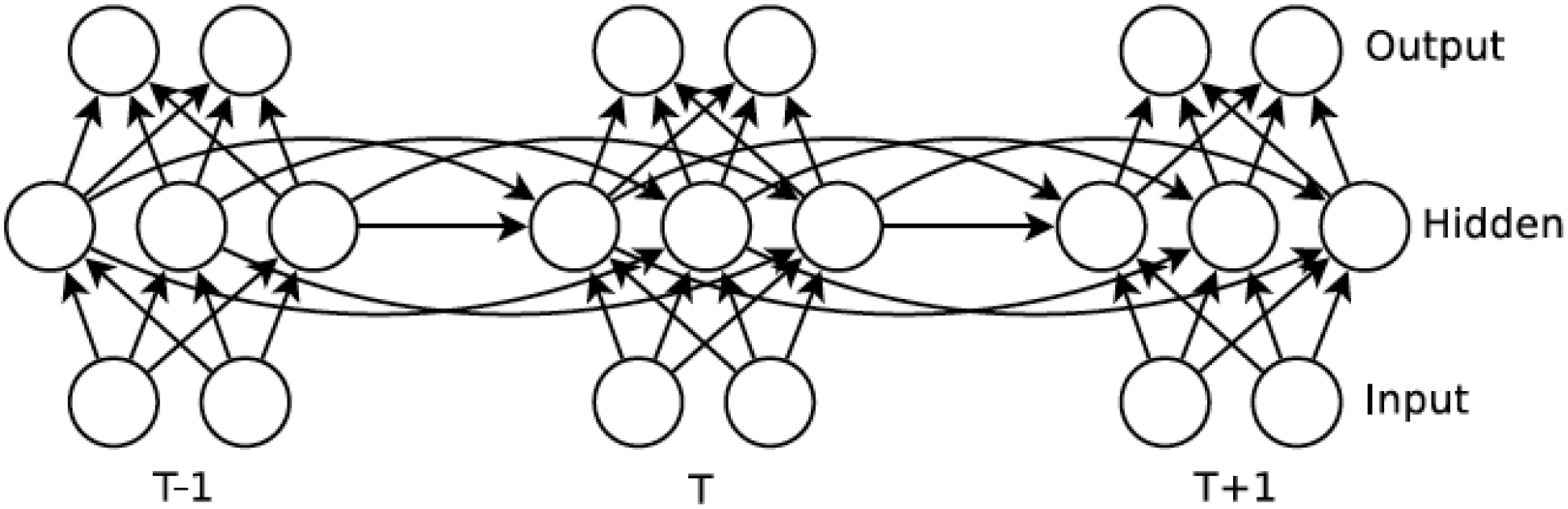


Рис.: Generating Text with Recurrent Neural Networks[9]

[9] http://www.cs.toronto.edu/~ilya/pubs/2011/LANG-RNN.pdf

LSTM, #1

Память в RNN:

- long-term memory: параметры/веса сети в процессе обучения медленно изменяются, кодируя общие знания о предметной области
- short-term memory: проявляется в процессе прохода сигнала по рекуррентным слоям
- Long Short-Term Memory[10] промежуточный способ памяти, проявляется в специальной конструкции нейронов с памятью

Так же, LSTM - это способ борьбы с ростом и затуханием градиентов.

[10] LONG SHORT-TERM MEMORY, Hochreiter & Schmidhuber
`http://deeplearning.cs.cmu.edu/pdfs/Hochreiter97_lstm.pdf`

LSTM, #2

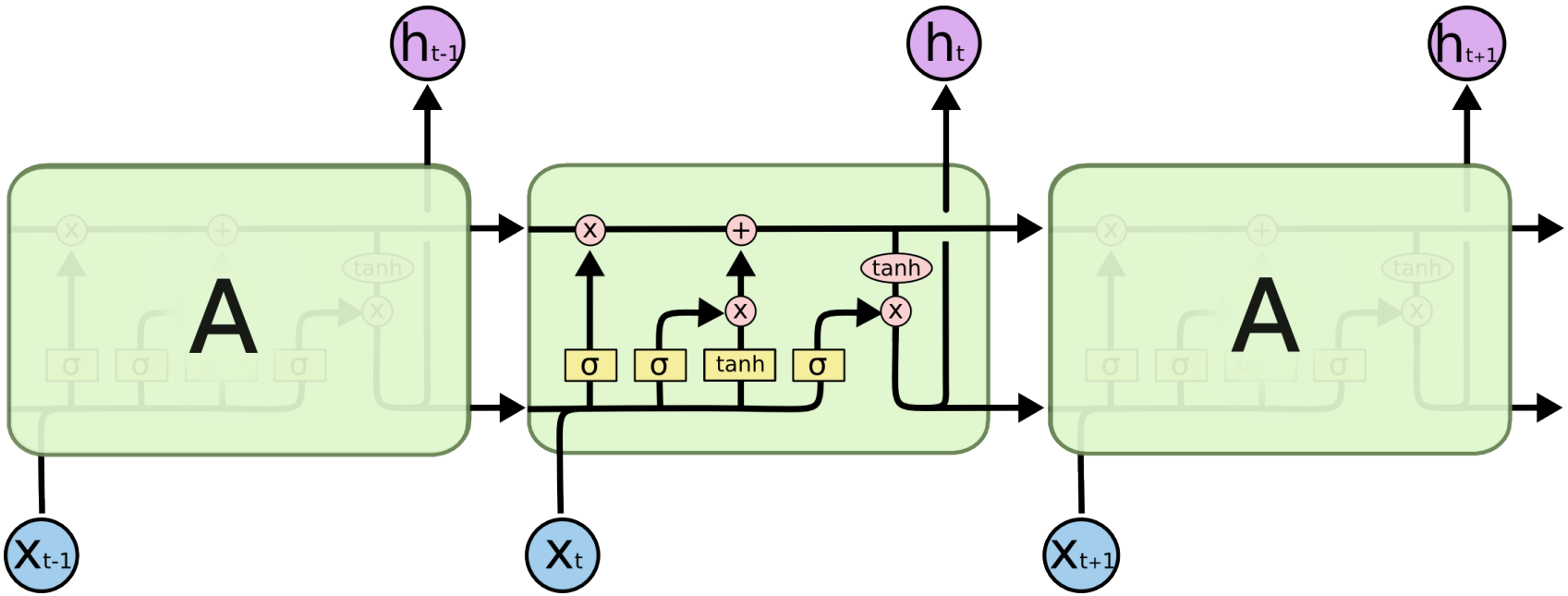
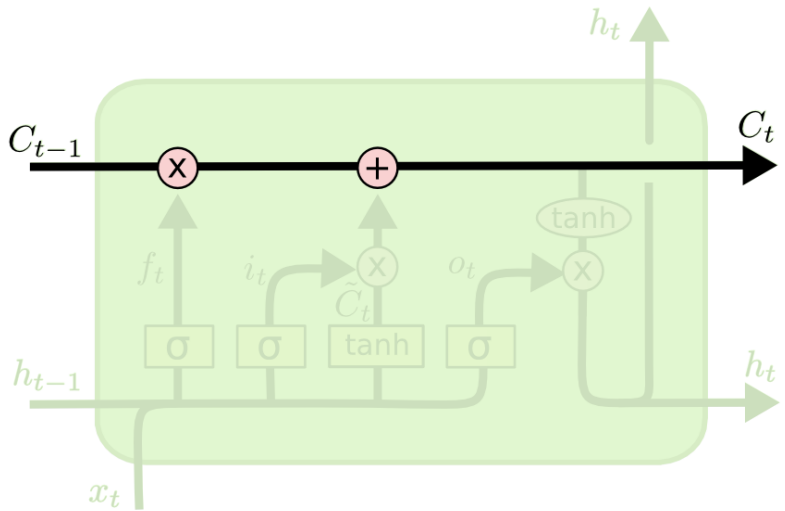


Рис.: Оригинальная модель memory unit версии 1997 года[11]; Так же как и в простых реккурентных сетях, у нас есть скрытый слой с циклической связью.

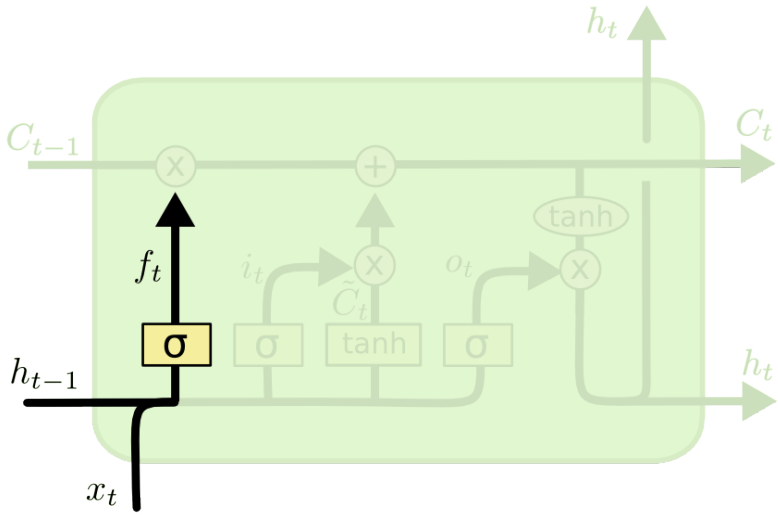
[11] http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM, #3. "Конвейер"состояний



- Важная состовляющая LSTM - слой состояния сети C_t. LSTM может как добавлять новую информацию к состояниям, так и стирать старую.

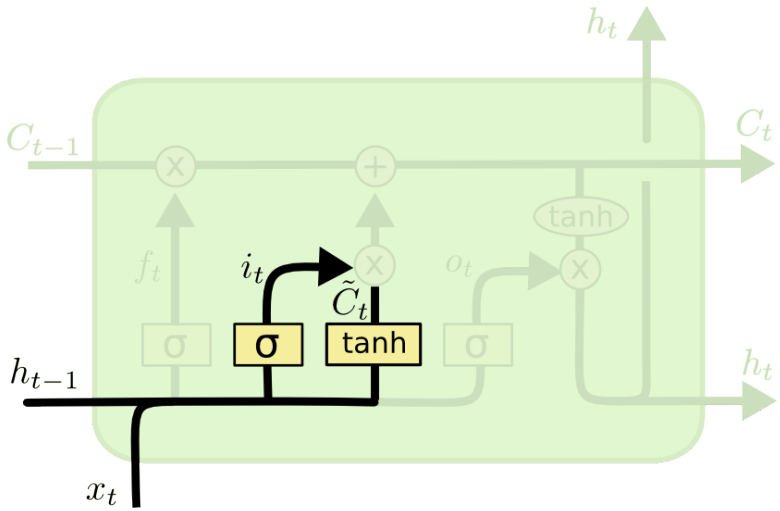
LSTM, #4. Забывающий слой



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

- Forget gate layer f_t с помощью сигмодидальной функции смотрит на значения x_t и h_t и выдает для каждого числа в C_{t-1} число от 0 (полностью забыть) до 1 (полностью сохранить).

LSTM, #5

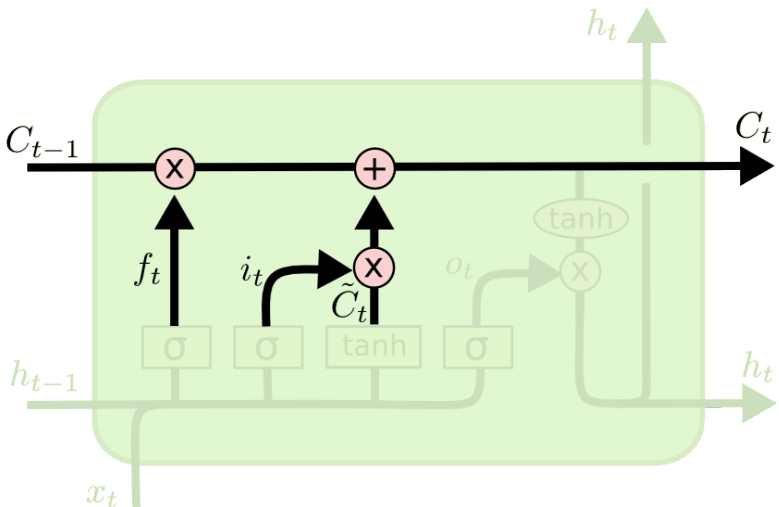


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

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

- Операция i_t играет роль входного слоя ("input gate layer"), решая какие значения обновлять. Как и в забывающем слое при $i_t = 0$ ничего не передается, при $i_t = 1$ передается все.
- Далее с помощью tanh вычисляются значения-"кандидаты"новых состояний.

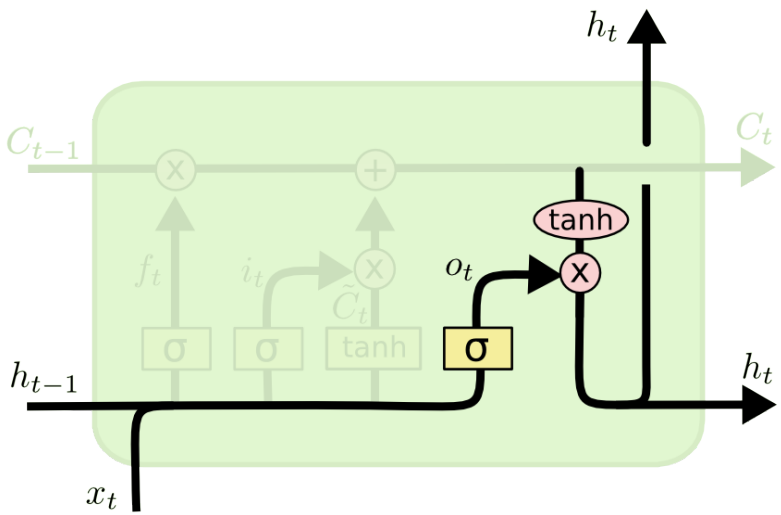
LSTM, #6



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

- Обновляем вектор состояний

LSTM, #7. Слой выхода

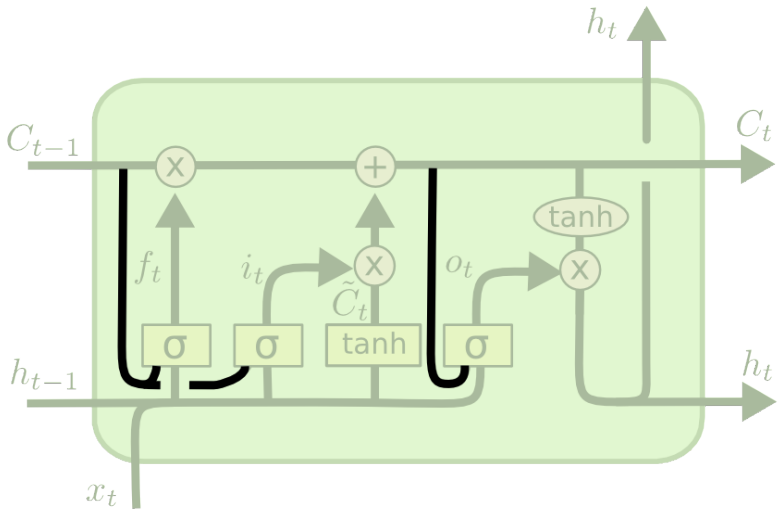


$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

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

- Наконец, решаем, что нам нужно вывести на данном шаге.

LSTM, #8. Варианты



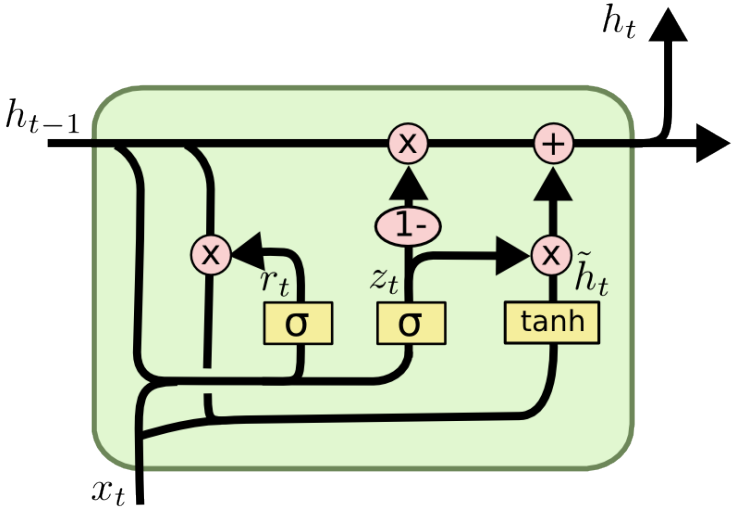
$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_i\right)$$
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] \; + \; b_o\right)$$

Рис.: “Peephole connections” [12]

- Даем возможность всем слоям смотреть на вектор состояний.

[12] ftp://ftp.idsia.ch/pub/juergen/TimeCount-IJCNN2000.pdf

LSTM, #8. Варианты



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

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

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

Рис.: Gated Recurrent Unit [13]

- Объединяем забывающий и входной слои в "слой обновления". Объединяем состояния и скрытый слой.

[13] http://arxiv.org/pdf/1406.1078v3.pdf

LSTM, #10

- визуализация работы LSTM - `http://karpathy.github.io/2015/05/21/rnn-effectiveness/` или из доп. материалов

RNN with memory

- Facebook: Memory Networks[14]
- Google: Neural Turing Machine[15]

Bilbo travelled to the cave. Gollum dropped the ring there. Bilbo took the ring. Bilbo went back to the Shire. Bilbo left the ring there. Frodo got the ring. Frodo journeyed to Mount-Doom. Frodo dropped the ring there. Sauron died. Frodo went back to the Shire. Bilbo travelled to the Grey-havens. The End.

- Where is the ring? A: Mount-Doom
- Where is Bilbo now? A: Grey-havens
- Where is Frodo now? A: Shire

- Google: Differentiable neural computers [16]

[14] http://arxiv.org/pdf/1410.3916v10.pdf
[15] http://arxiv.org/pdf/1410.5401v2.pdf
[16] https://deepmind.com/blog/differentiable-neural-computers/

BRNN

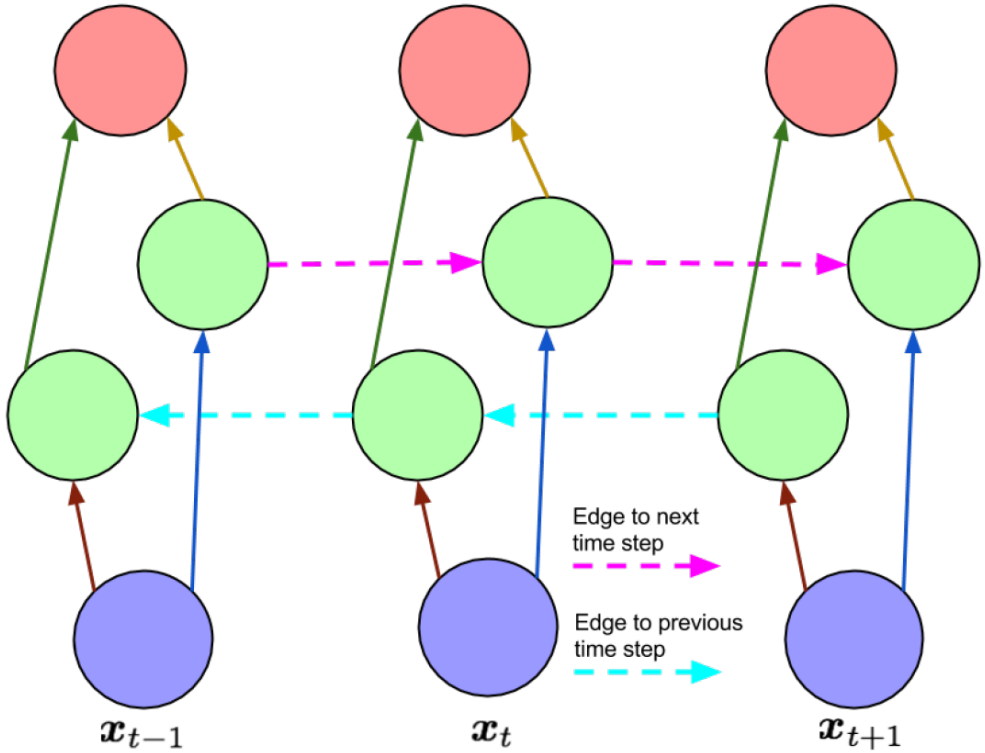


Рис.: Recurrent layer [17]

- $h^{(t)} = \sigma\left(W_{hx}x^{(t)} + W_{hh}h^{(t-1)} + b_h\right)$
- $z^{(t)} = \sigma\left(W_{zx}x^{(t)} + W_{zz}z^{(t-1)} + b_z\right)$
- $\hat{y}^{(t)} = \text{softmax}\left(W_{yh}h^{(t)} + W_{yz}z^{(t)} + b_y\right)$

[17] Bidirectional recurrent neural networks
`http://www.di.ufpe.br/~fnj/RNA/bibliografia/BRNN.pdf`

Вопросы