курс «Глубокое обучение»

Рекуррентные нейросети

Александр Дьяконов

(ВМК МГУ имени М.В. Ломоносова)

23 марта 2020 года

План

Рекуррентные нейросети: RNN

LSTM (Forget / Input / Output Gate, Cell update)

Gated Recurrent Unit (GRU)

Метод форсирования учителя (teacher forcing)

Scheduled sampling

Двунаправленные (Bidirectional) RNN

Глубокие (Deep) RNN

Глубокие двунаправленные / многонаправленные RNN

Рекурсивные (Recursive Neural Networks) НС

Exploding / Vanishing gradients

Особенности регуляризации в RNN: Dropout

Особенности регуляризации в RNN: Batchnorm

Интерпретация RNN

Image Captioning

Image Captioning with Attention

Про лучевой поиск

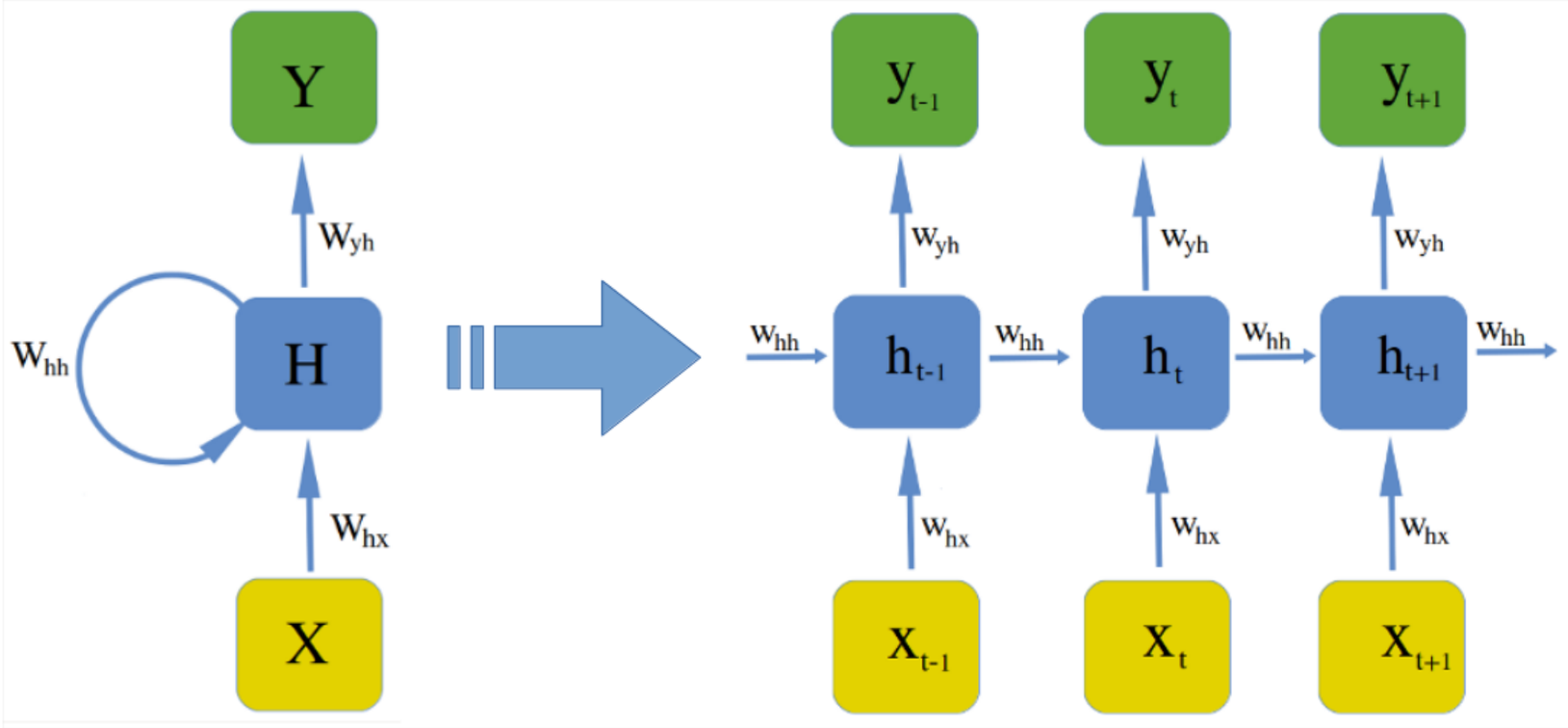
Пиксельные RNN

Рекуррентная нейросеть (RNN = Recurrent neural network)

– для обработки последовательностей

использование выхода (output) / скрытого состояния (hidden state)

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;)

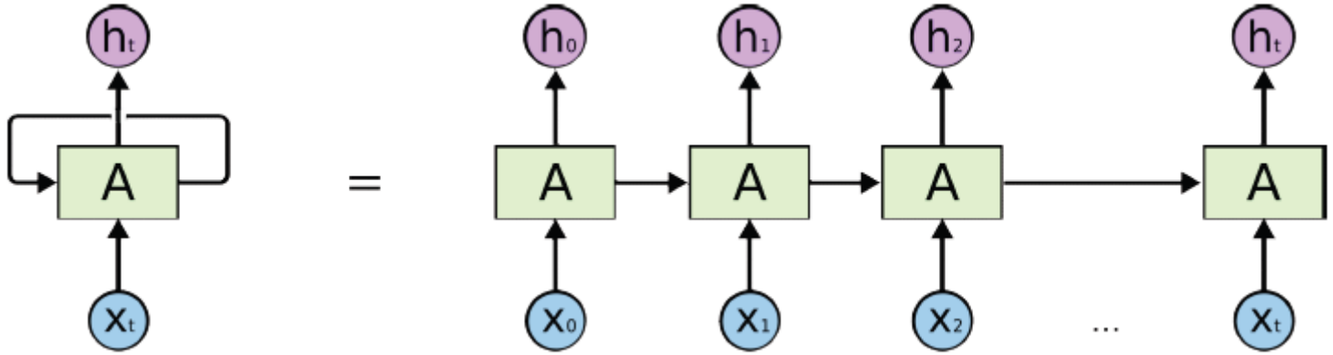
Матрицы весов одинаковые при обработке любого элемента последовательности (символ, слово, …)

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



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

$$\dots$$

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

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

тут для простоты
убрали свободный член

Это для однослойной сети!

линейный слой + нелинейность
без свободного члена

индексы могут быть другие

RNN: форма записи

$$h_t = \sigma(W_{hh} h_{t-1} + W_{xh} x_t) \quad \sim \quad h_t = \sigma\left(W \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix} \right) \equiv \sigma\left(W[h_{t-1}; x_t]\right)$$

это обычная однослойная сеть

Потенциальные проблемы –
забывание
должны помнить начало последовательности

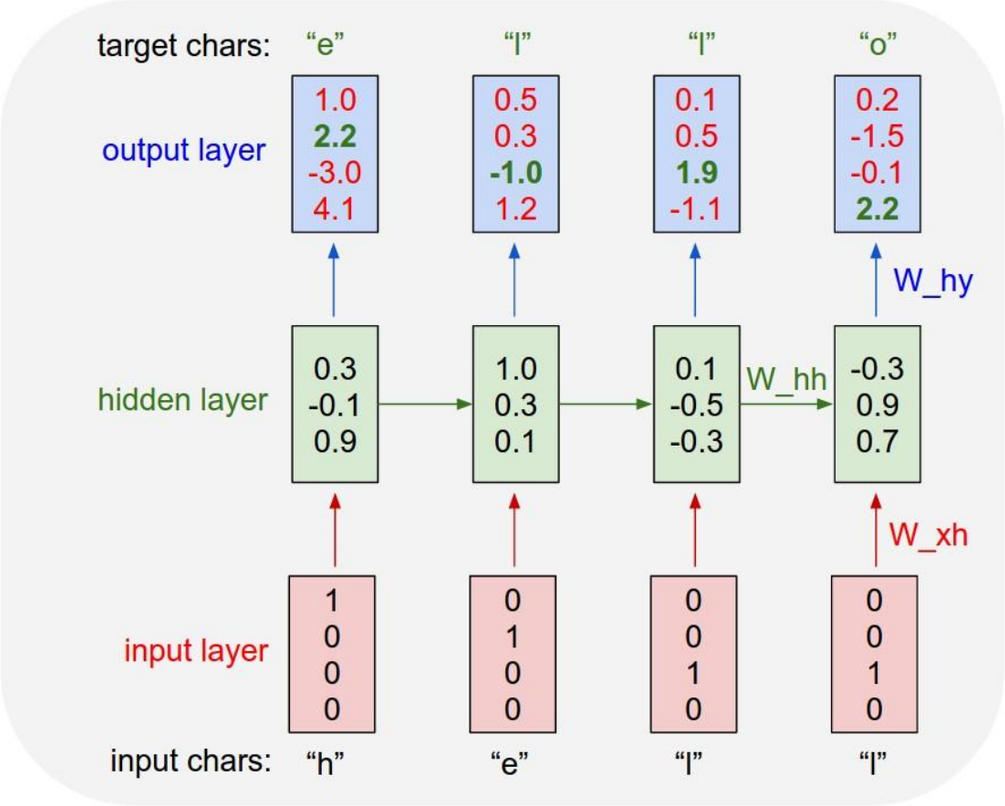
градиенты
дальше разберём

RNN: обучение

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

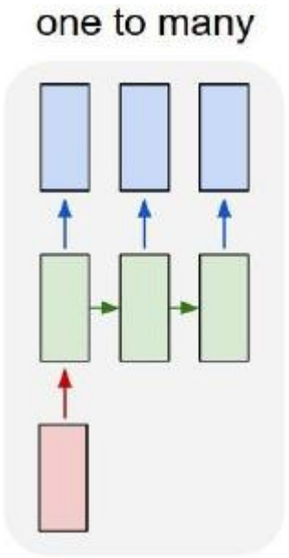
как будто мы «разворачиваем» рекуррентную сеть и даём на вход всю последовательность

Пример работы RNN

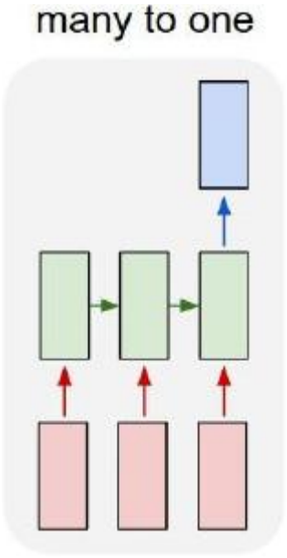


http://karpathy.github.io/2015/05/21/rnn-effectiveness/

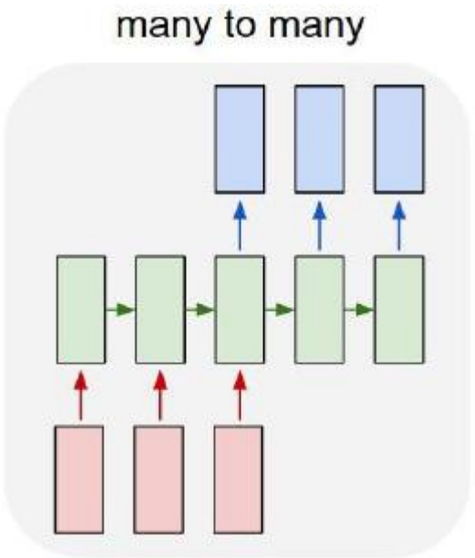
Применение RNN



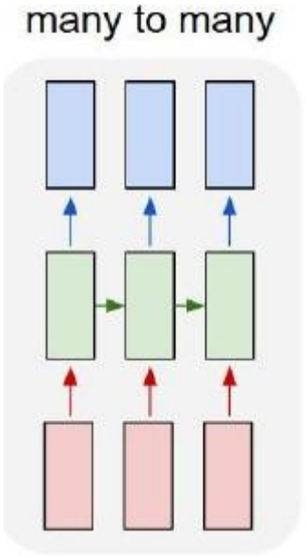
описание изображения



тема / настроение текста



машинный перевод

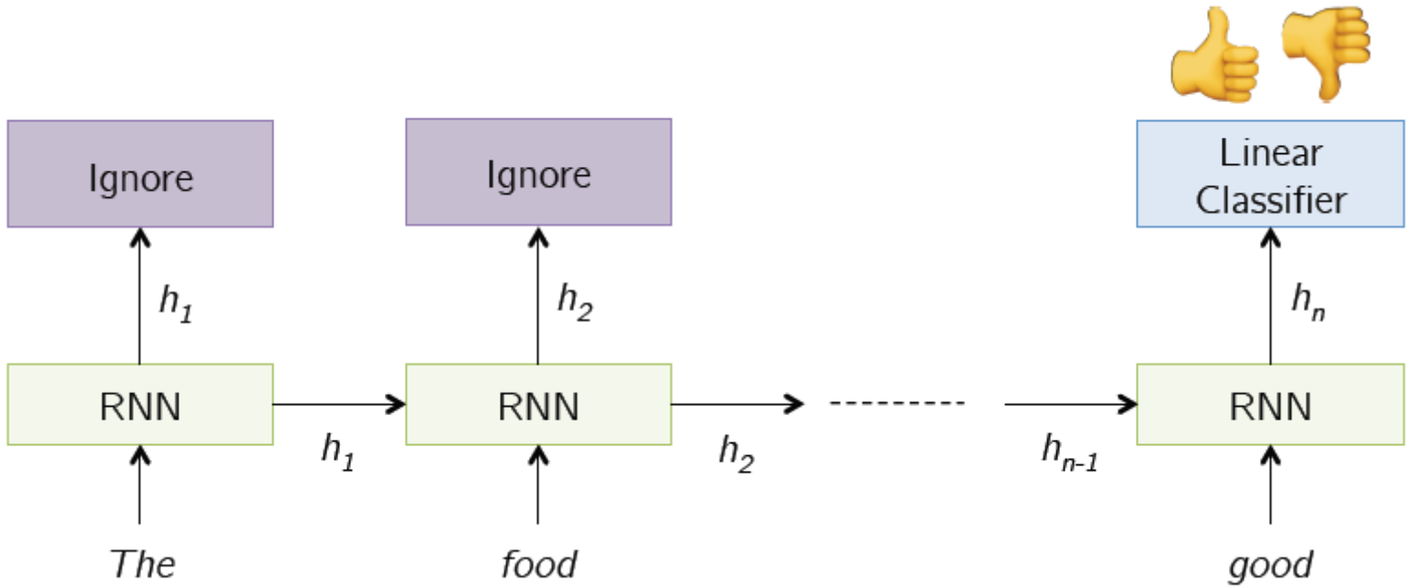


классификация фреймов видео

Можно по-разному собирать блоки –
для решения разных задач
http://karpathy.github.io/2015/05/21/rnn-effectiveness/

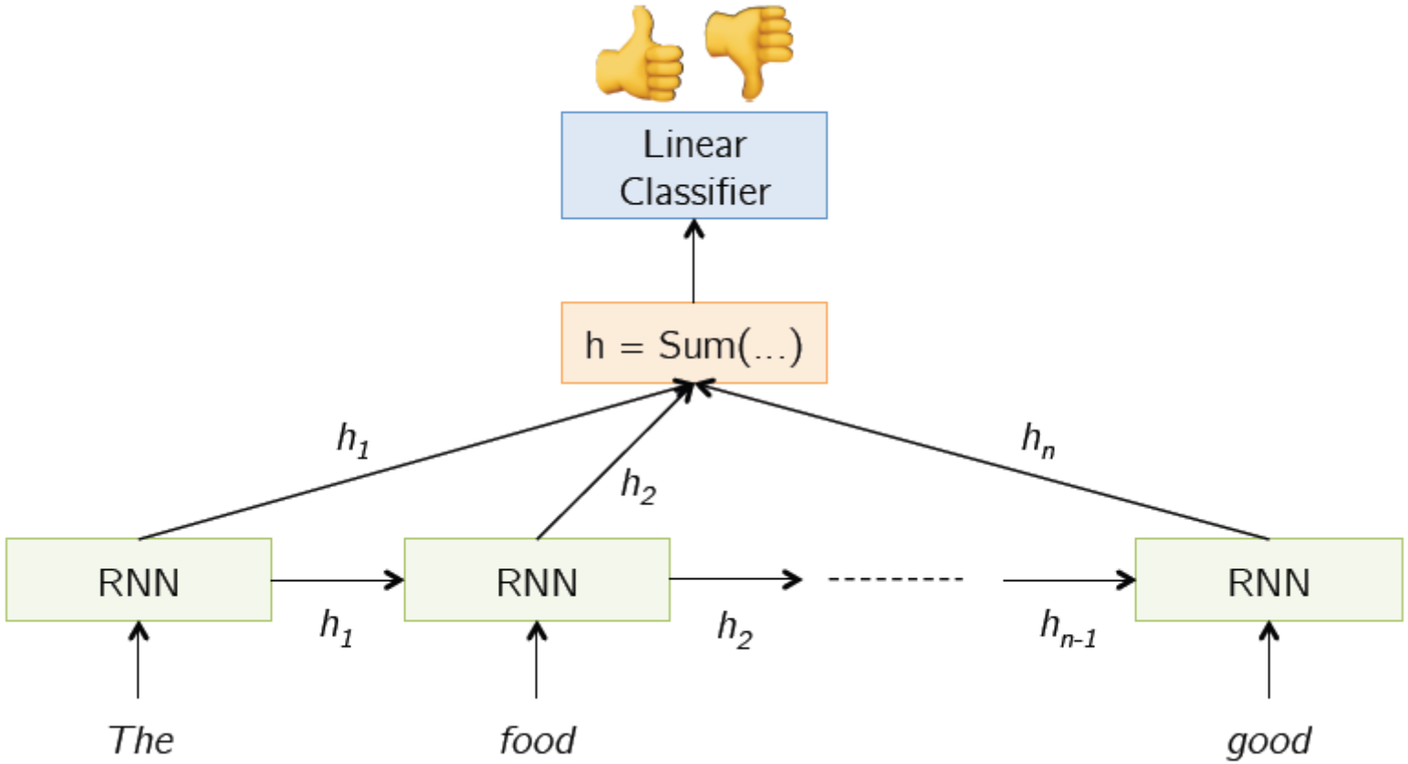
RNN: как решать задачи классификации
пример: тональность сообщения

Первый способ



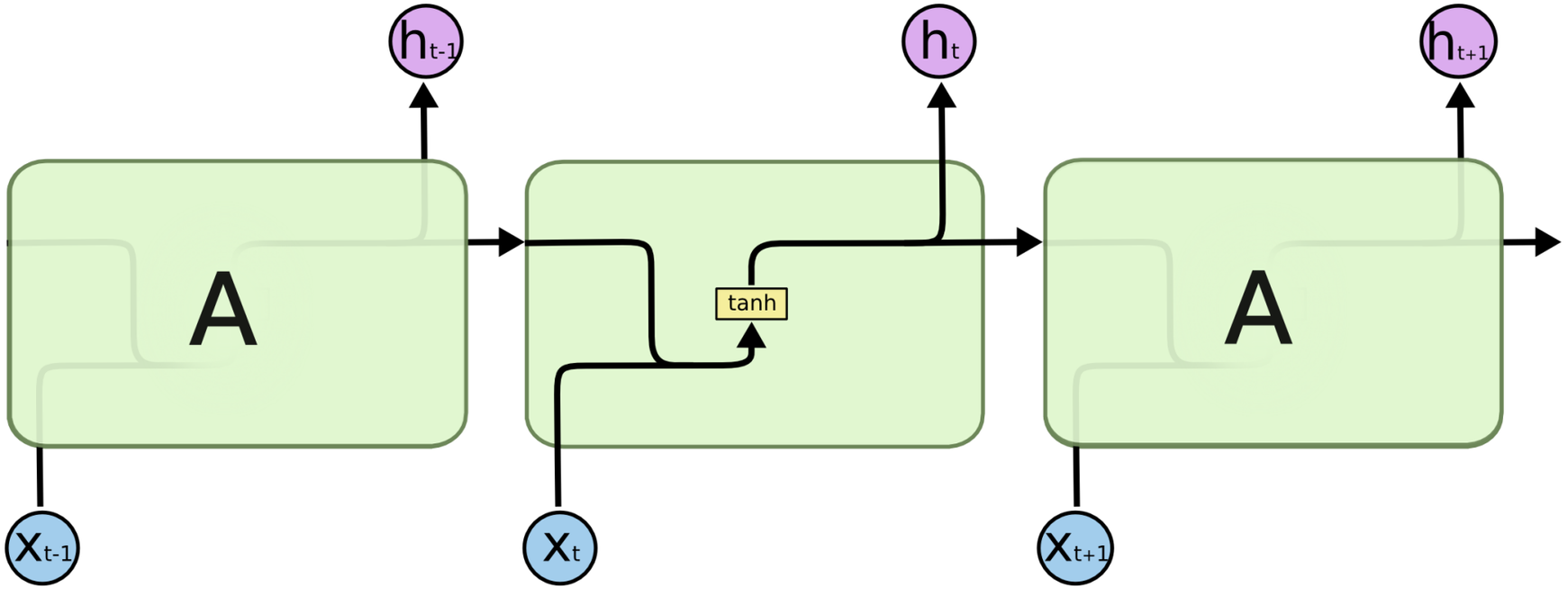
RNN: как решать задачи классификации
пример: тональность сообщения

Второй способ



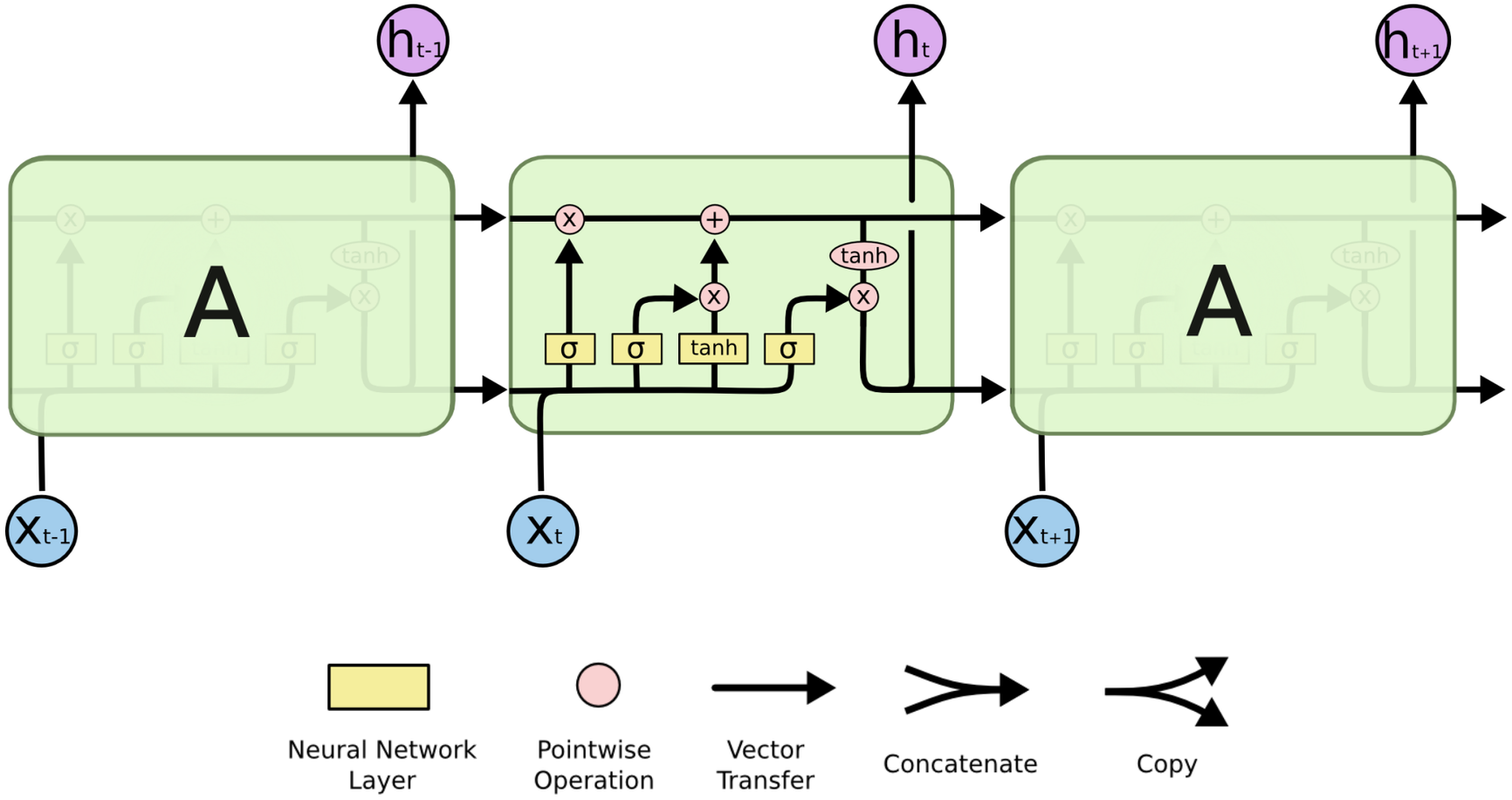
http://slazebni.cs.illinois.edu/spring17/

Стандартная RNN



LSTM (Long Short Term Memory)

здесь другой базовый блок:

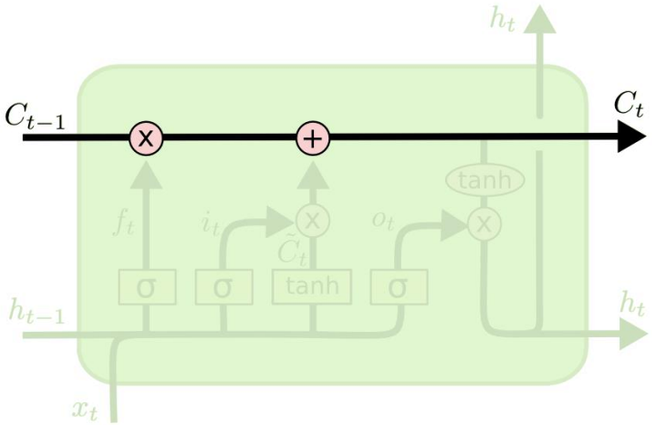


[Hochreiter&Schmidhuber, 1997]

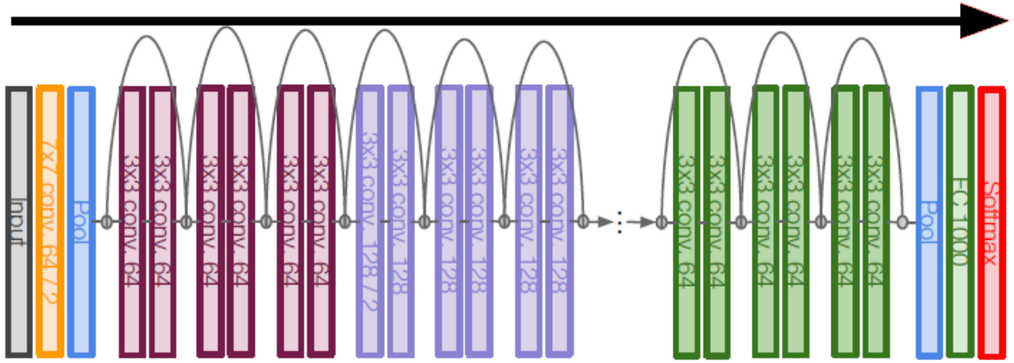
Ключевая идея LSTM

«состояние ячейки/блока» – проходит через все блоки

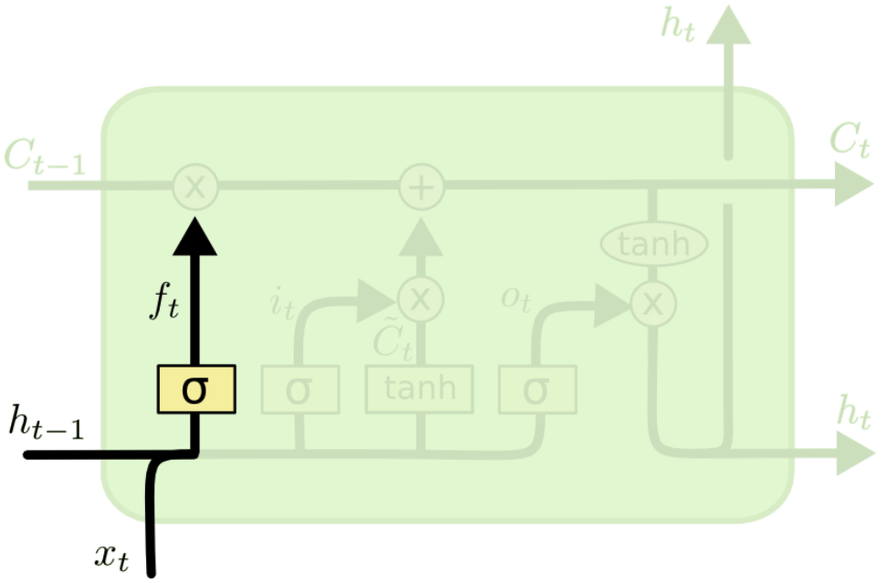
«Peephole connection»



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



Забывающий гейт (Forget Gate)



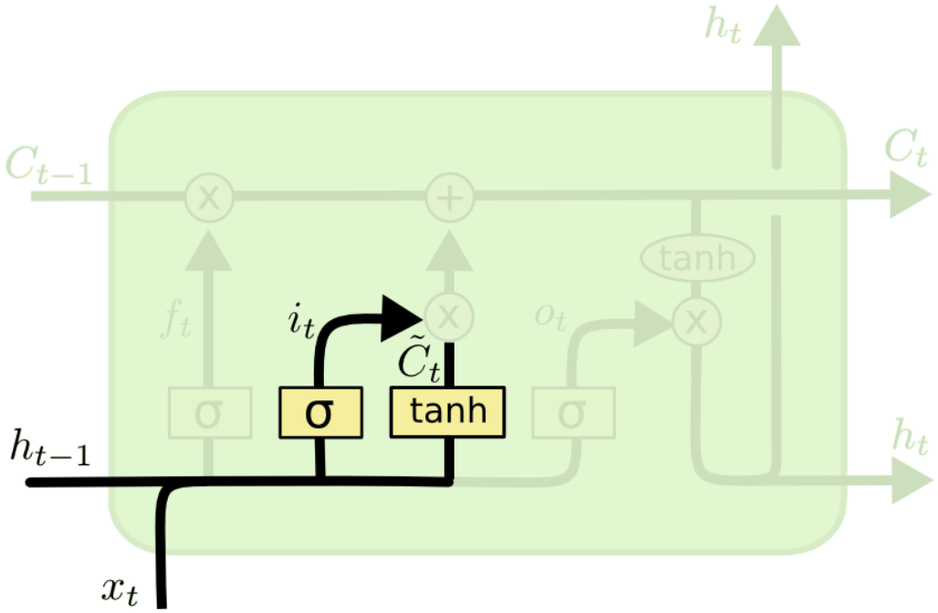
$$f_t = \sigma(W_f[h_{t-1}; x_t] + b_f)$$

если = 1 – передаём полностью состояние блока

если = 0 – то забываем предыдущее состояние

строго равенства не будут выполняться

Входной гейт (Input Gate)



входной гейт:

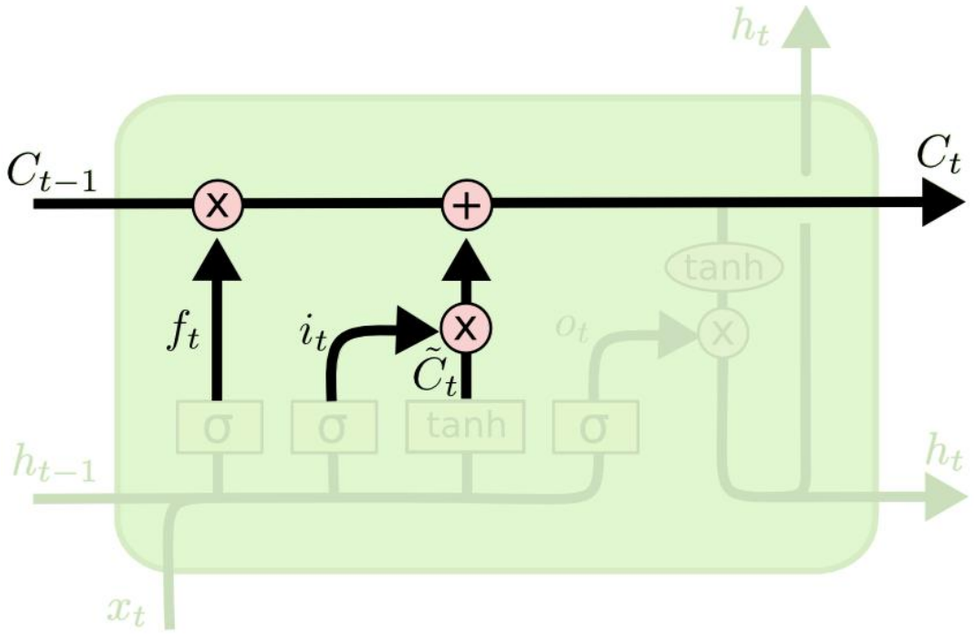
$$i_t = \sigma(W_i[h_{t-1}; x_t] + b_i)$$

текущее состояние:

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

Какую новую информацию учитываем в состоянии…

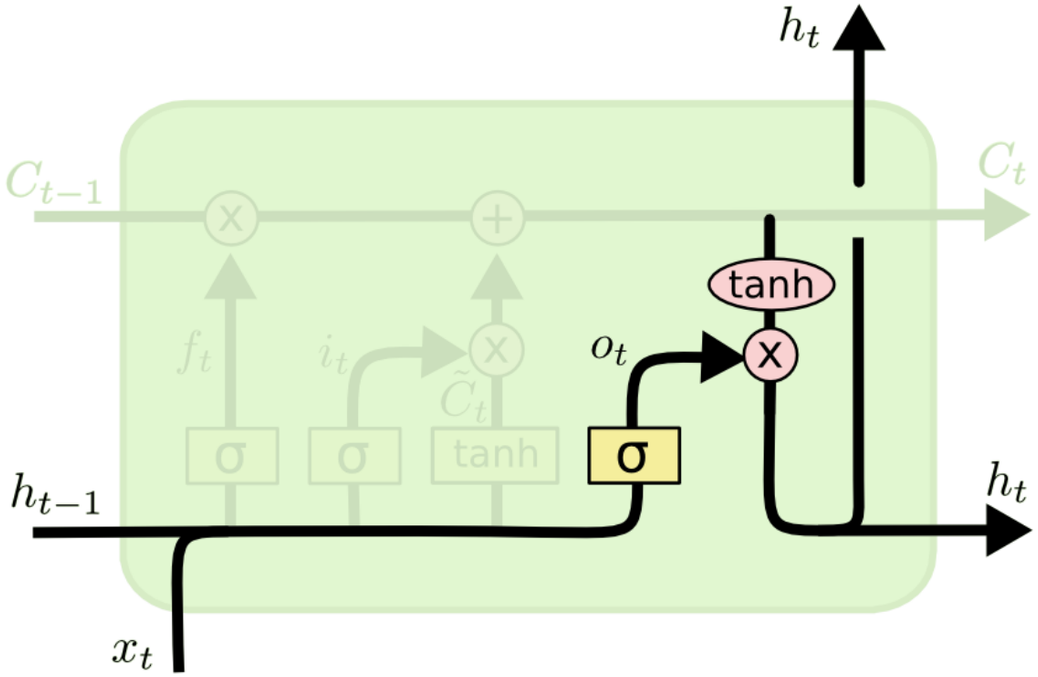
Обновление состояния (Cell update)



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

новое состояние = (старое состояние | гейт) + (посчитанное состояние | гейт)

Выходной гейт (Output Gate)



выходной гейт:

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

скрытое состояние:

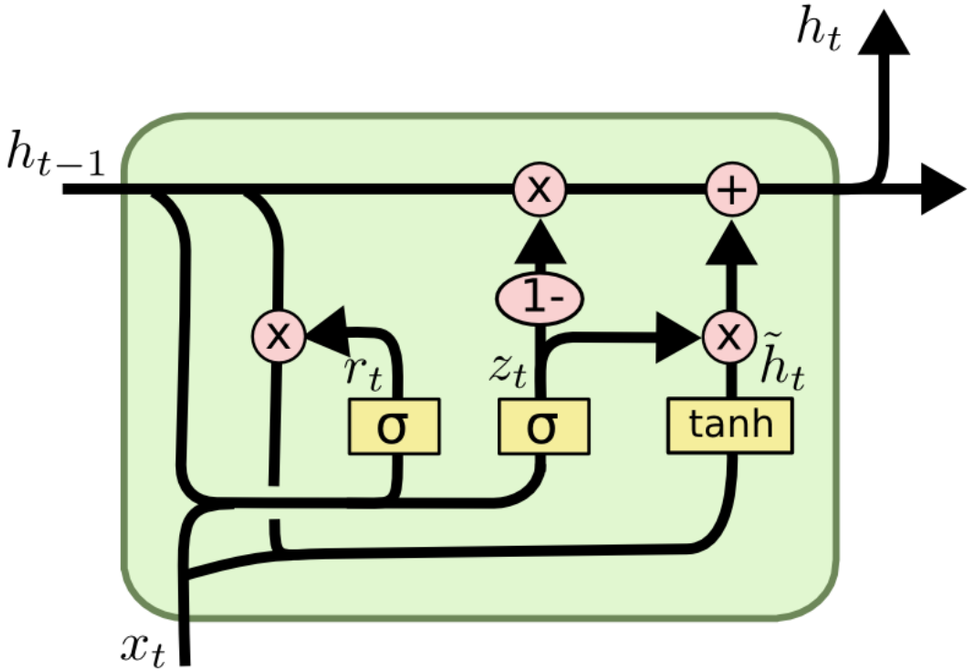
$$h_t = o_t \tanh(C_t)$$

LSTM with peephole connections

$$f_t = \sigma(W_f[h_{t-1}; x_t; C_{t-1}] + b_f)$$
$$i_t = \sigma(W_i[h_{t-1}; x_t; C_{t-1}] + b_i)$$
$$o_t = \sigma(W_o[h_{t-1}, x_t; C_t] + b_o)$$
$$\tilde{C}_t = \tanh(W_C[h_{t-1}; x_t] + b_C)$$
$$C_t = f_t C_{t-1} + i_t \tilde{C}_t$$
$$h_t = o_t \tanh(C_t)$$

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?

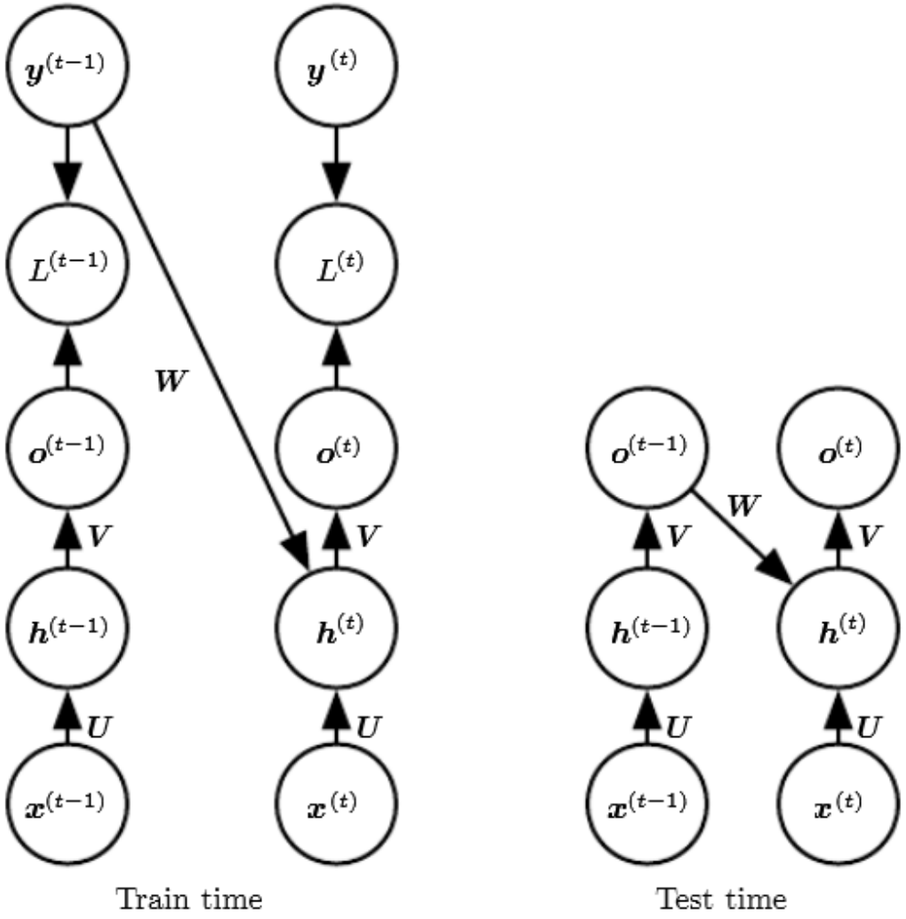
Есть обзоры

«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



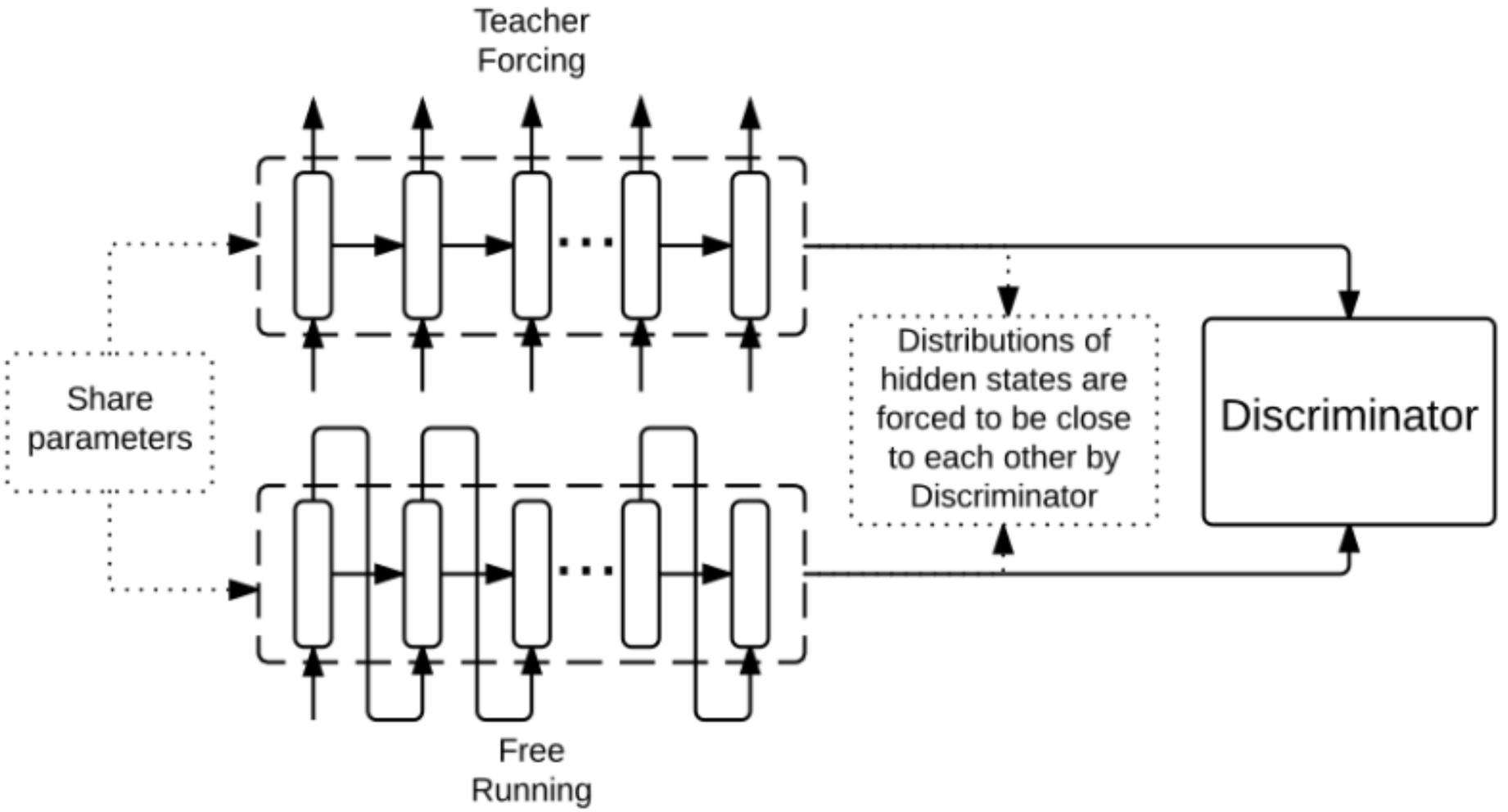
Только если связь типа «выход-модель» (не передаётся скрытая переменная)

+ можно не делать BPTT

– то что видит при тестировании и обучении может отличаться

+ можно использовать для предтренировки

Приёмы: метод форсирования учителя (teacher forcing)

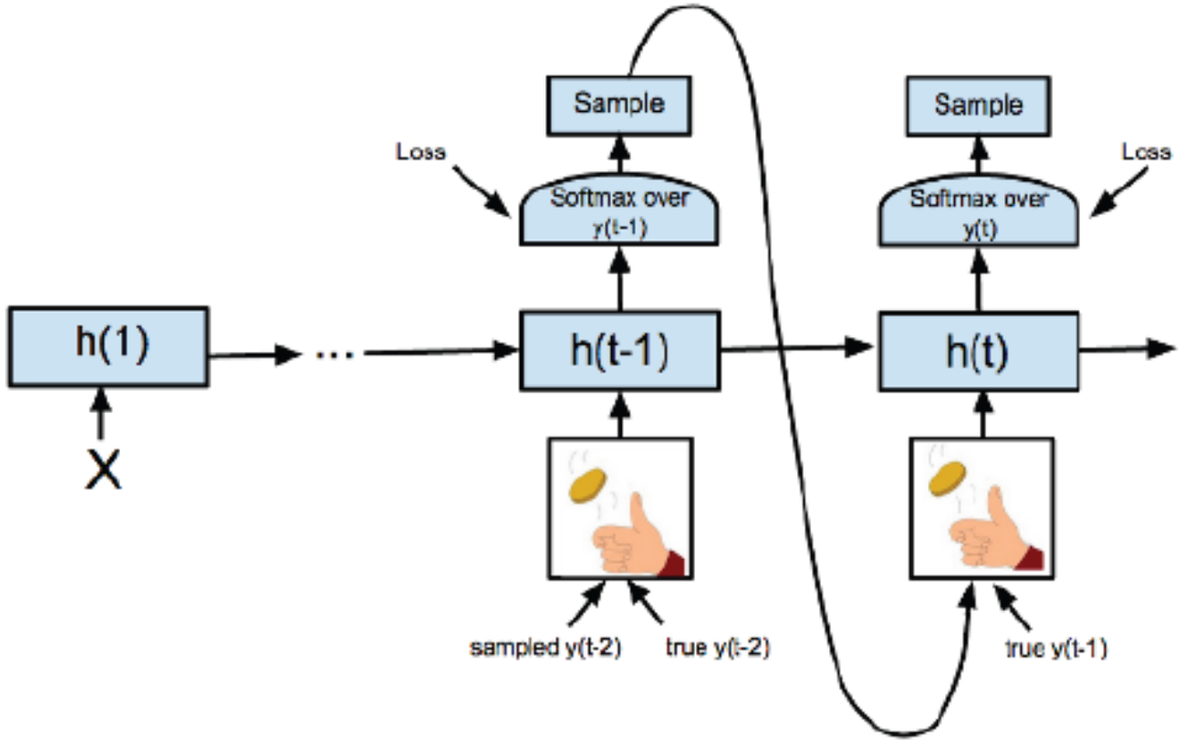


можно хитрее: одновременно истинная метка и сгенерированная

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**

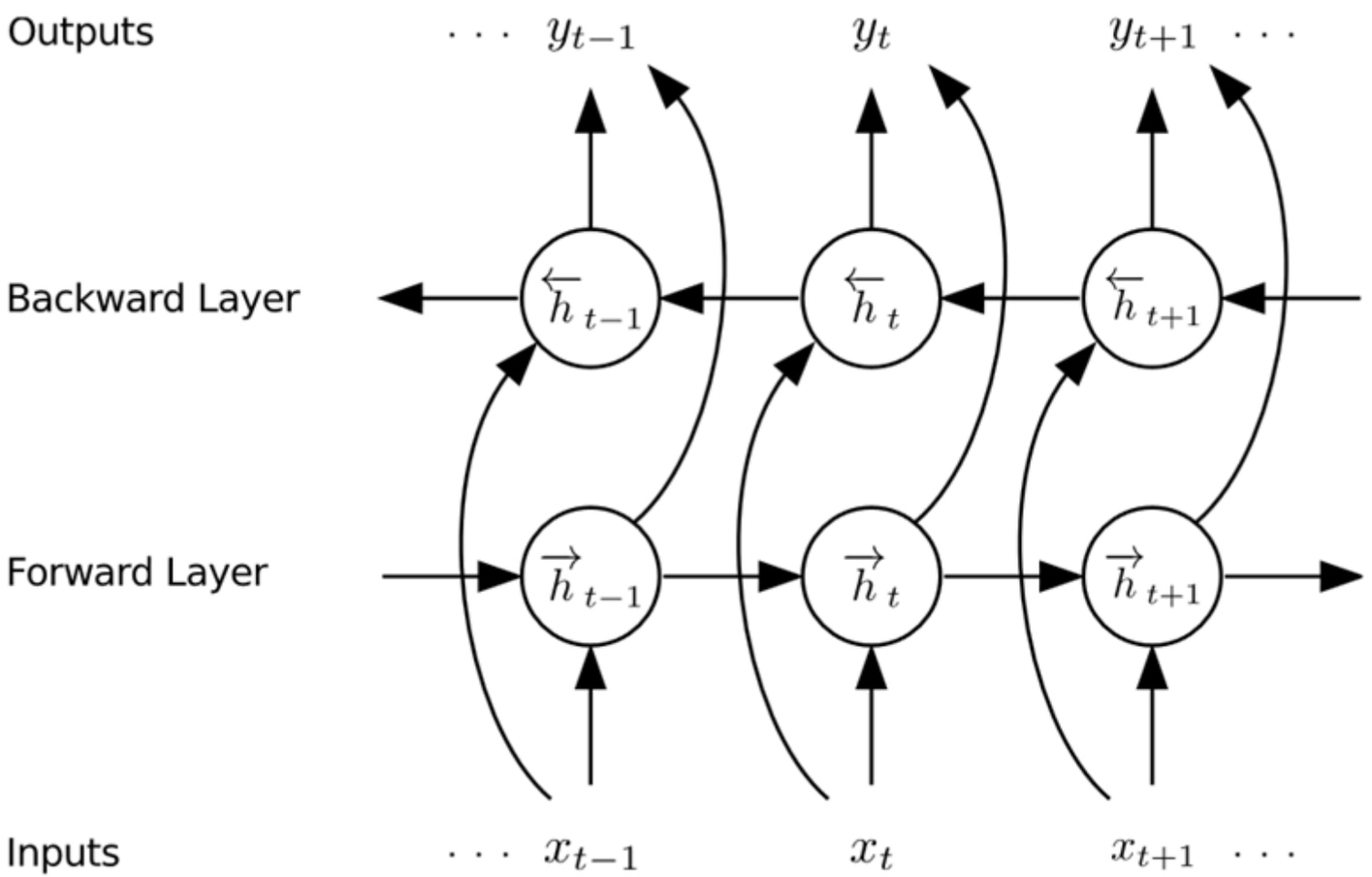
годится и для вывода последовательности чисел

Приёмы: минибатчи

последовательности в батче выравнивают по длине (дополняют пустыми символами)

Лучше брать последовательности примерно равной длины

Двунаправленные (Bidirectional) RNN

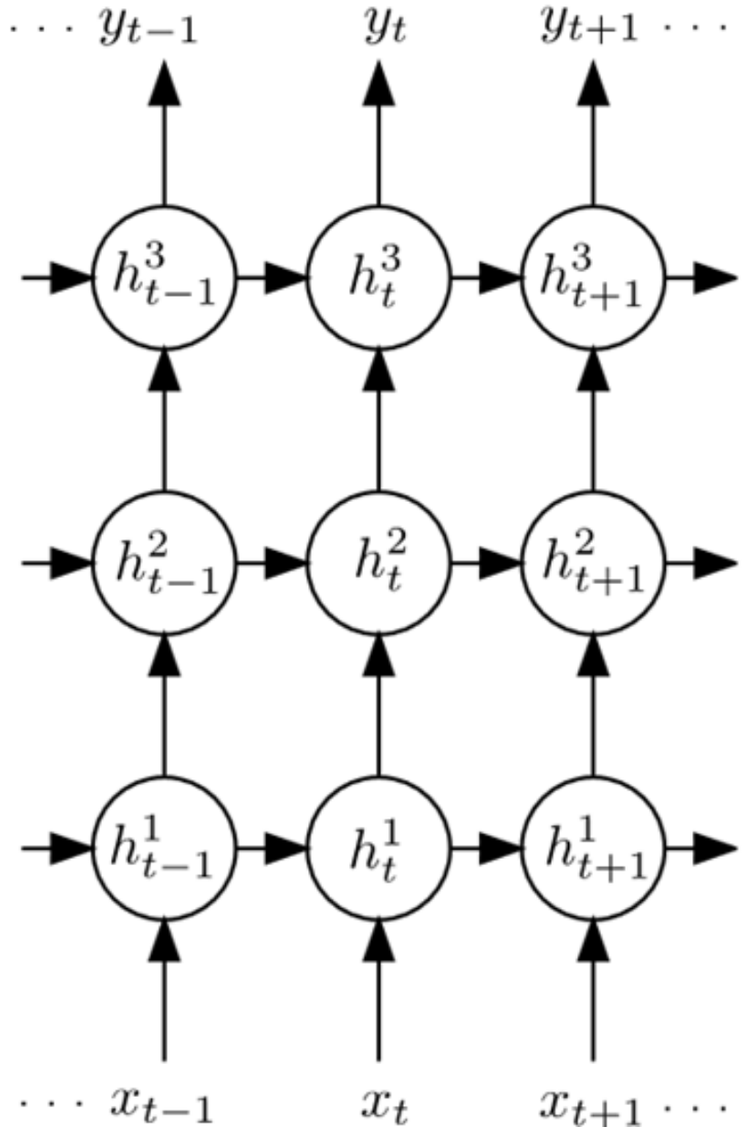


распознавание рукописных текстов

распознавание речи

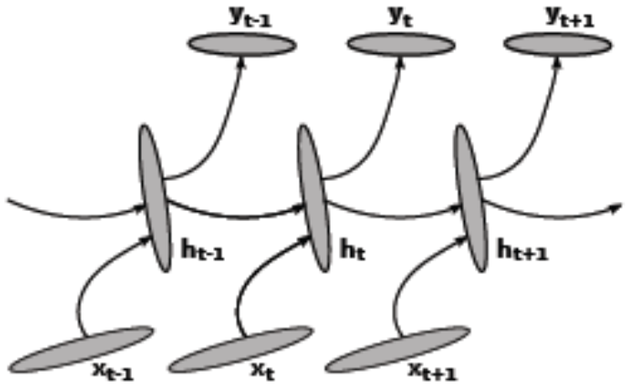
биоинформатика

Глубокие (Deep) RNN / Multi-layer RNN / stacked RNN

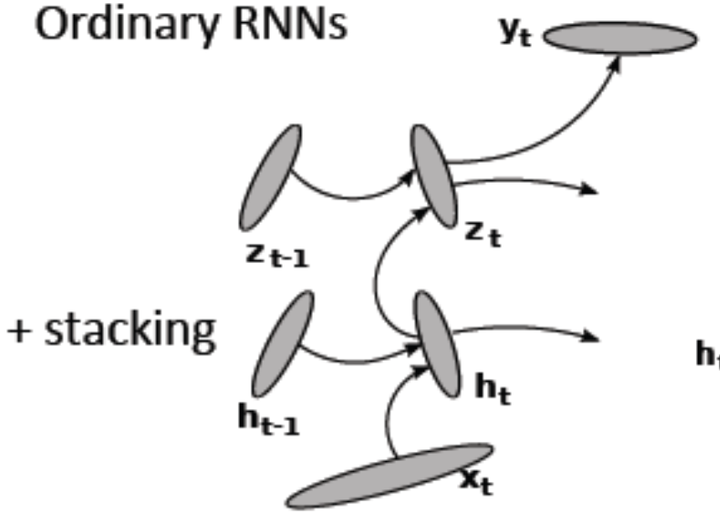


Как строить глубокие RNN

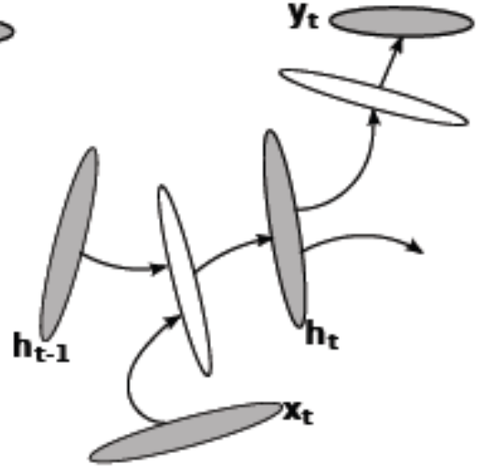
(пример – вариантов много!)

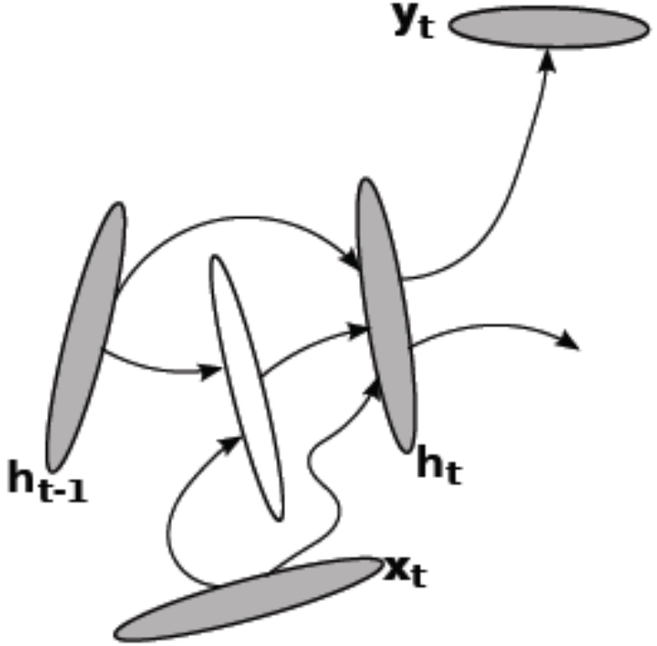


Ordinary RNNs



+ deep hid-to-out
+ deep hid-to-hid
+deep in-to-hid





+ skip connections for creating shorter paths

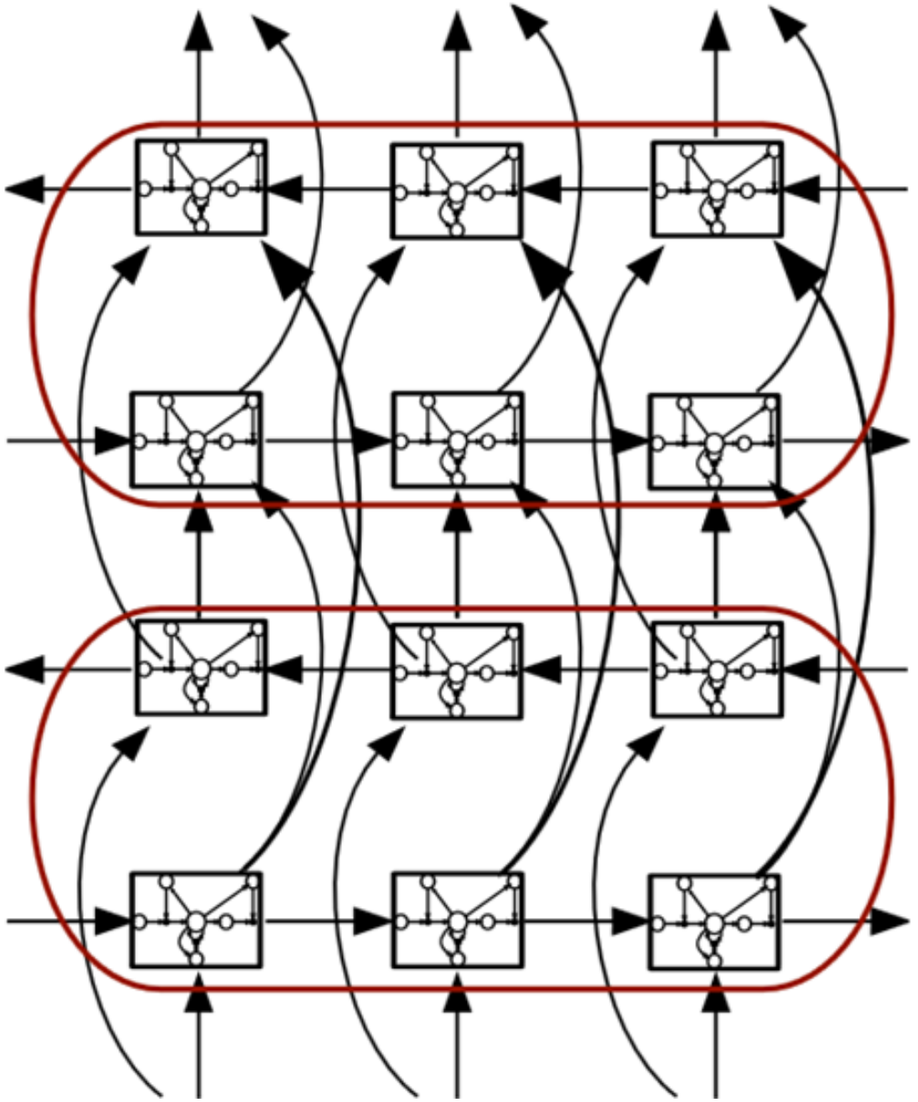
8

Глубокие RNN

Обычно не слишком глубокие (по сравнению с CNN)
~ 4 слоя

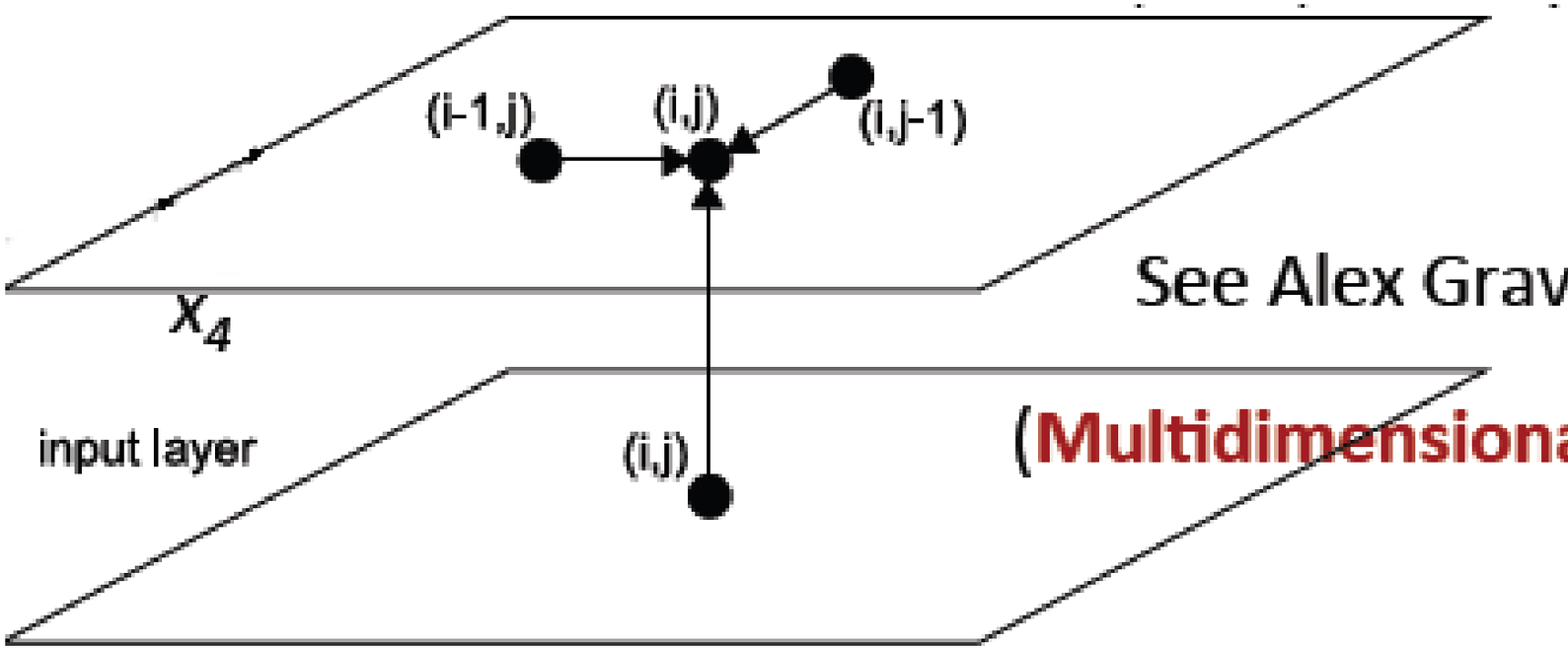
Хотя дальше трансформеры ~12 слоёв

Глубокие двунаправленные (Bidirectional) RNN



но нужна вся входная последовательность! не всегда есть…

Многонаправленные RNN / Многомерные RNN

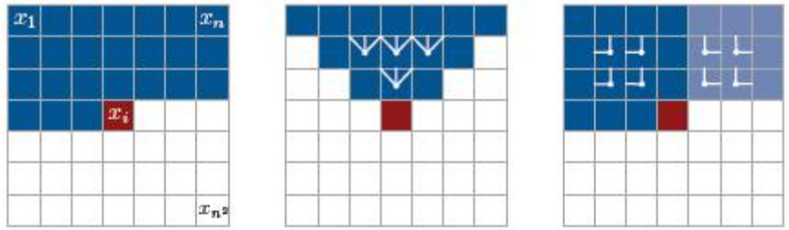


See Alex Graves's work, e.g., 2012

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

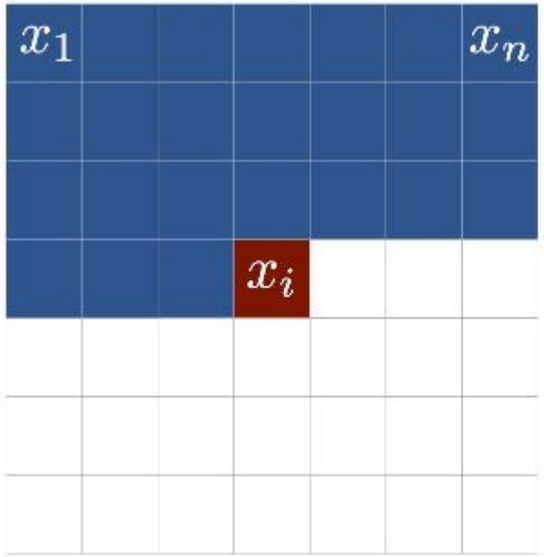
Пиксельные RNN

хорошо учат текстуру



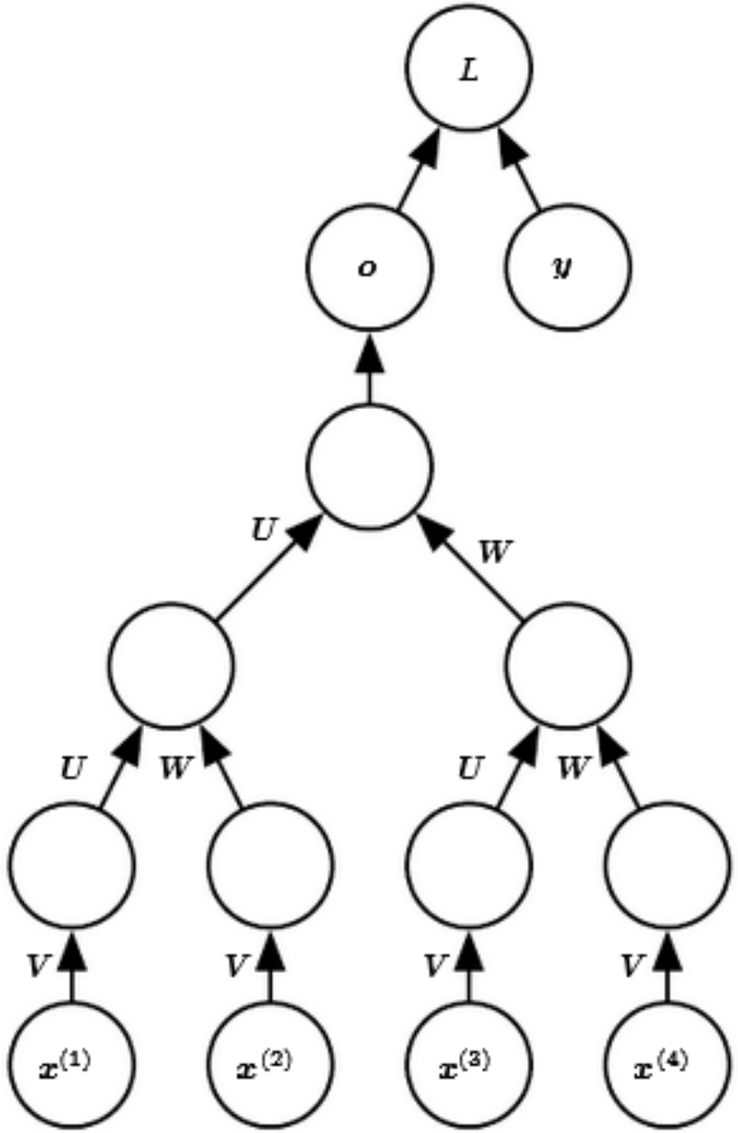
van den Oord (DeepMind) et al ICML 2016, best paper

Пиксельные RNN



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

Рекурсивные (Recursive Neural Networks) НС

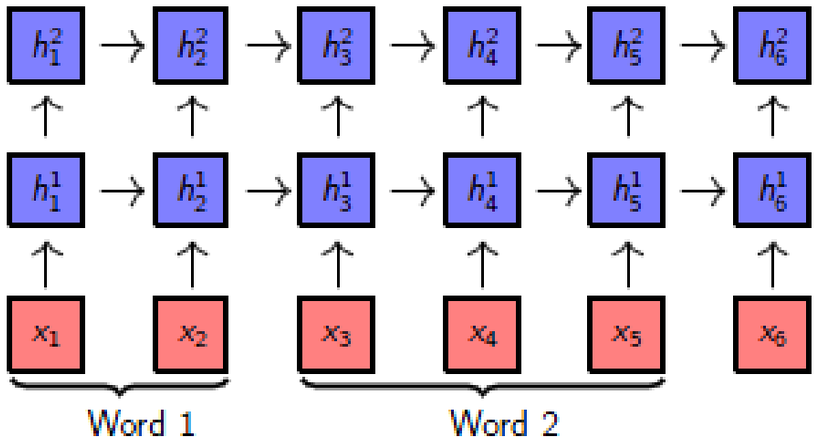


Hierarchical Multiscale Recurrent Neural Networks

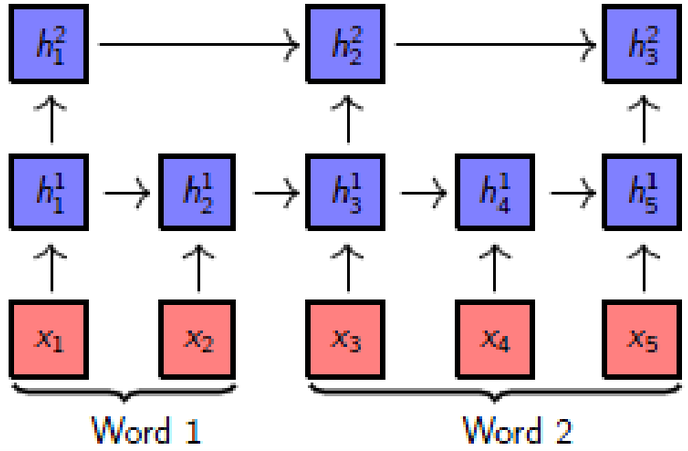
У текстов структура на разных масштабах:

буквы → слова → фразы → предложения → абзацы

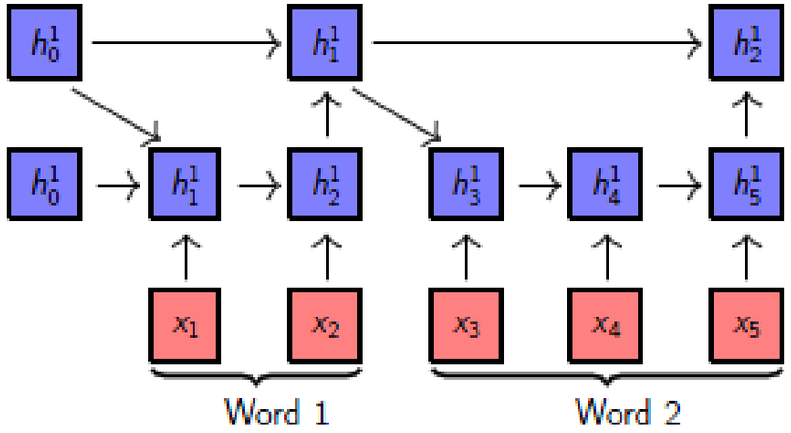
Stacked RNN



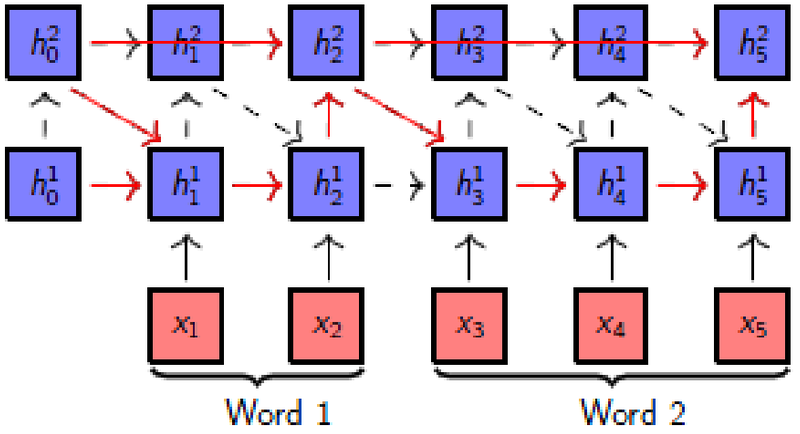
Clockwork RNN



Boundary-aware RNN



Hierarchical Multiscale RNN



Chung J., Ahn S., Bengio Y. «Hierarchical Multiscale Recurrent Neural Networks», 2017 // https://arxiv.org/abs/1609.01704

Hierarchical Multiscale Recurrent Neural Networks

+ вычислительная эффективность (верхние слои проще)

+ меньше изменений $\Rightarrow$ лучше распространение информации

– сеть теперь не дифференцируема

- можно использовать Хэвисайда во время прямого распространения и игнорировать порог во время обратного
- можно склон делать всё более крутым

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…

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

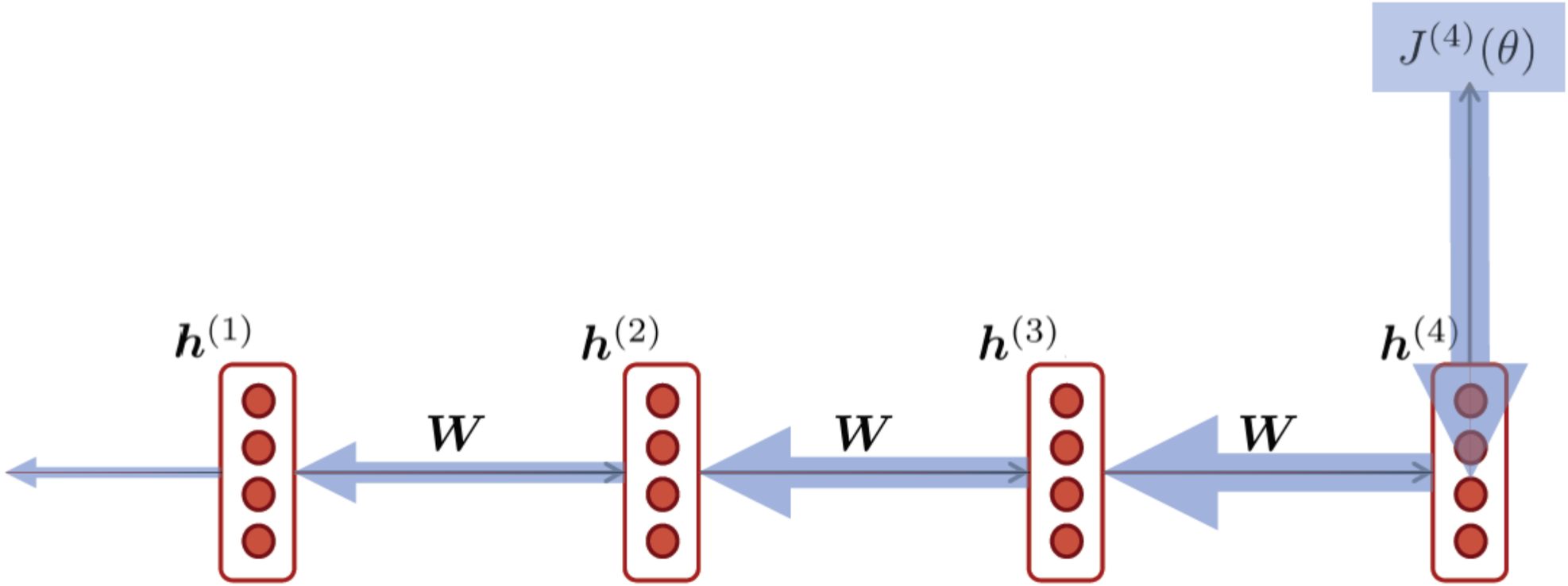
$$\frac{\partial h_t}{\partial h_k} = \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial h_{t-2}} \dots \frac{\partial h_{k+1}}{\partial h_k} = \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} =$$

– произведение Якобианов

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

диагонализация вектора

Самая главная проблема RNN – Exploding / Vanishing gradients



$$\frac{\partial J^{(4)}}{\partial h^{(1)}} = \frac{\partial h^{(2)}}{\partial h^{(1)}} \times \frac{\partial h^{(3)}}{\partial h^{(2)}} \times \frac{\partial h^{(4)}}{\partial h^{(3)}} \times \frac{\partial J^{(4)}}{\partial h^{(4)}}$$

http://web.stanford.edu/class/cs224n/

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**
- **Обрезка градиентов (Clipping gradients)**
- **Метод форсирования учителя (Teacher Forcing)**
- **Ограничение шагов обратного распространения (Truncated Backpropagation Through Time)**
- **Эхо-сети (Echo State Networks)**

знаем…

было…

было…

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

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

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

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

заодно – распространение долговременных зависимостей

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

сложная формула;)

- **Инициализация**

Ex: у орт. матриц все с.з. = 1

Geoffrey et al «Improving Performance of Recurrent Neural Network with ReLU nonlinearity»

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

Особенности регуляризации в RNN: Dropout

Только на нерекуррентности

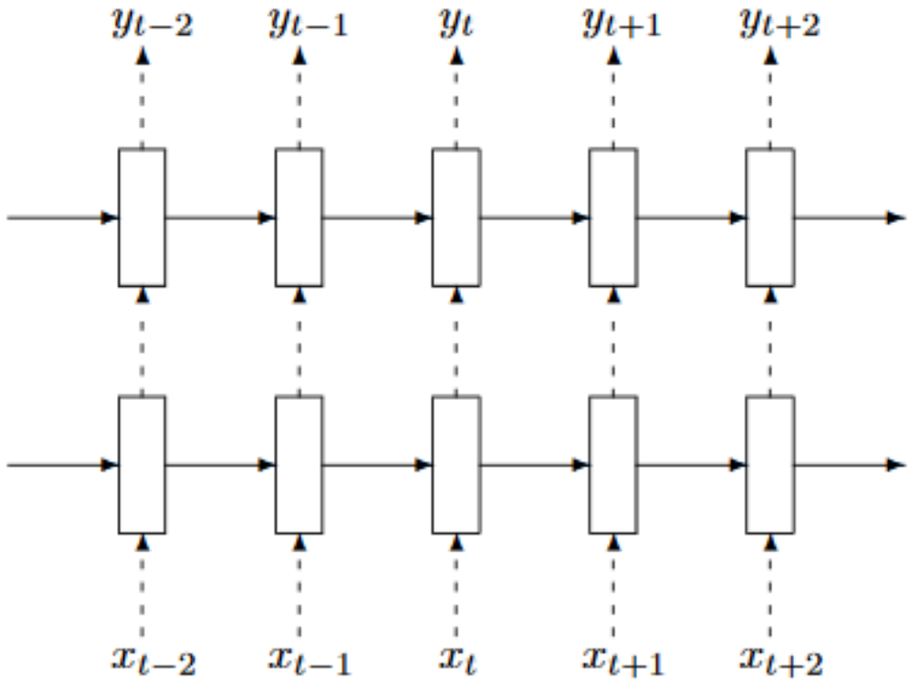


Figure 2: Regularized multilayer RNN. The dashed arrows indicate connections where dropout is applied, and the solid lines indicate connections where dropout is not applied.

Zaremba et al. 2014. «Recurrent neural network regularization» // https://arxiv.org/abs/1409.2329

Особенности регуляризации в RNN: Dropout

При адаптации состояния

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

несильное искажение состояния – трогаем только добавку

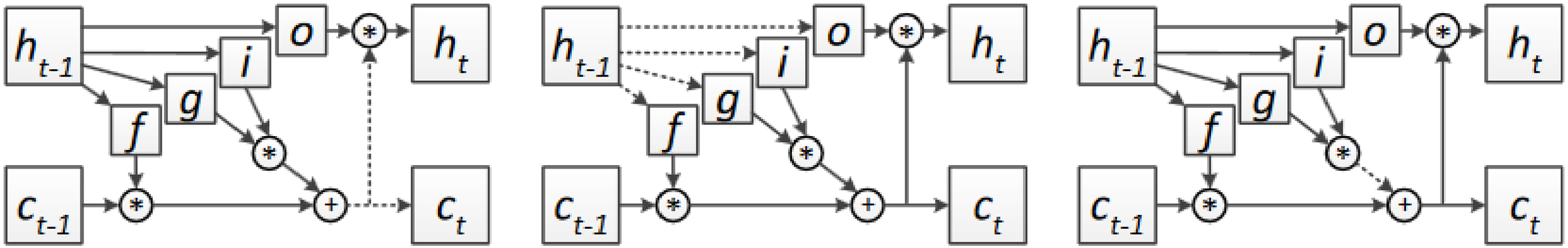


Figure 1: Illustration of the three types of dropout in recurrent connections of LSTM networks. Dashed arrows refer to dropped connections. Input connections are omitted for clarity.

хорошо своя маска для каждого шага

Semenuita et al. «Recurrent dropout without memory loss» // https://arxiv.org/abs/1603.05118

Gal 2015. «A theoretically grounded application of dropout in recurrent neural networks»

Особенности регуляризации в RNN: Batchnorm

Естественный вариант

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

но лучше

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

по аналогии с dropout – только к вертикальным связям, а не горизонтальным

- **нормировка может быть по батчам и по символам**
- **на отдельных шагах своя статистика (но проблемы с разной длиной)**

Особенности регуляризации в RNN: Batchnorm

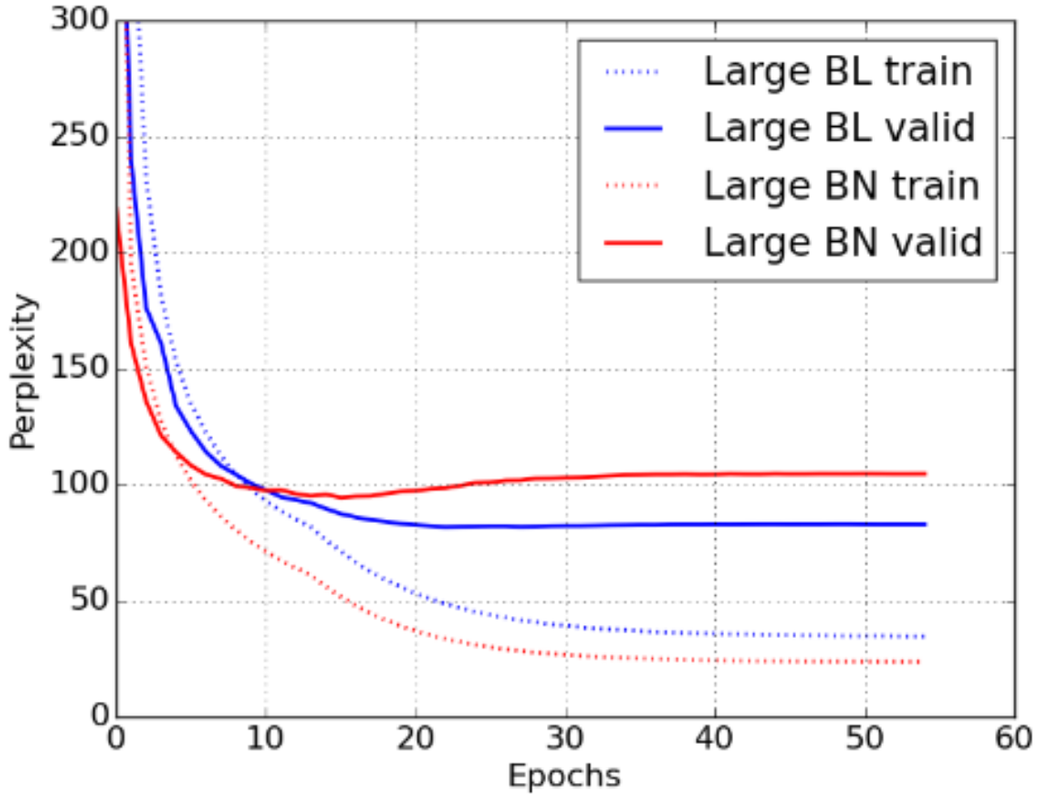


Figure 2: Large LSTM on Penn Treebank for the baseline (blue) and the batch normalized (red) networks. The dotted lines are the training curves and the solid lines are the validation curves.

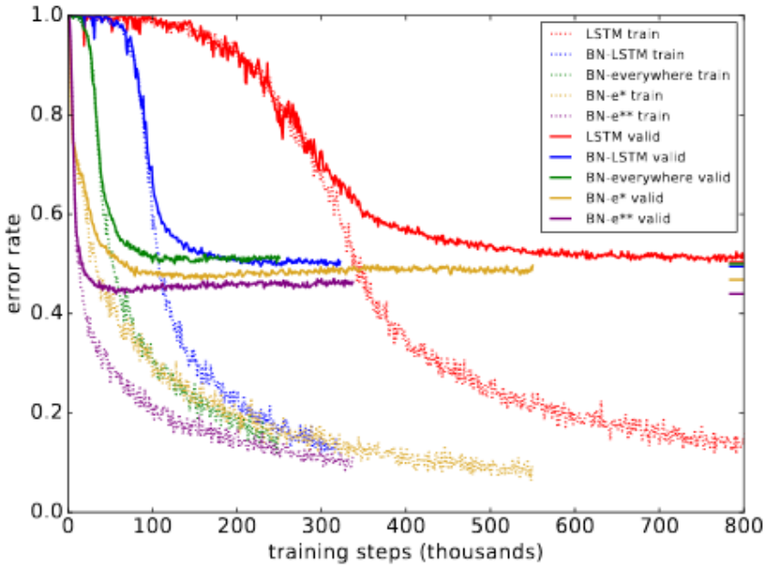
Laurent et al. «Batch normalized recurrent neural networks» // https://arxiv.org/abs/1510.01378

Особенности регуляризации в RNN: Batchnorm

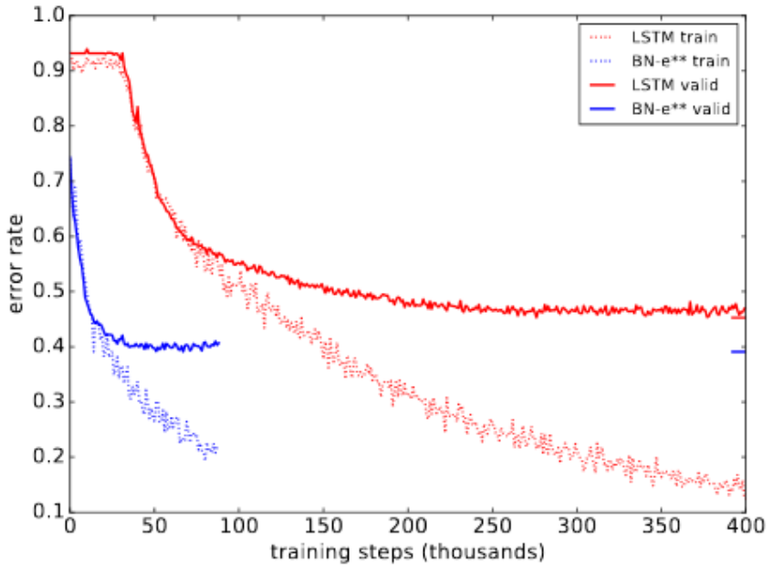
$$\begin{pmatrix} \tilde{\mathbf{f}}_t \\ \tilde{\mathbf{i}}_t \\ \tilde{\mathbf{o}}_t \\ \tilde{\mathbf{g}}_t \end{pmatrix} = \mathrm{BN}(\mathbf{W}_h \mathbf{h}_{t-1}; \gamma_h, \beta_h) + \mathrm{BN}(\mathbf{W}_x \mathbf{x}_t; \gamma_x, \beta_x) + \mathbf{b}$$

$$\mathbf{c}_t = \sigma(\tilde{\mathbf{f}}_t) \odot \mathbf{c}_{t-1} + \sigma(\tilde{\mathbf{i}}_t) \odot \tanh(\tilde{\mathbf{g}}_t)$$

$$\mathbf{h}_t = \sigma(\tilde{\mathbf{o}}_t) \odot \tanh(\mathrm{BN}(\mathbf{c}_t; \gamma_c, \beta_c))$$



(a) Error rate on the validation set for the Attentive Reader models on a variant of the CNN QA task (Hermann et al., 2015). As detailed in Appendix C, the theoretical lower bound on the error rate on this task is 43%.



(b) Error rate on the validation set on the full CNN QA task from Hermann et al. (2015).

Cooijmans et al. «Recurrent Batch Normalization» // https://arxiv.org/abs/1603.09025

MI (Multiplicative Integration)

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

(произведение адамарово), вместо

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

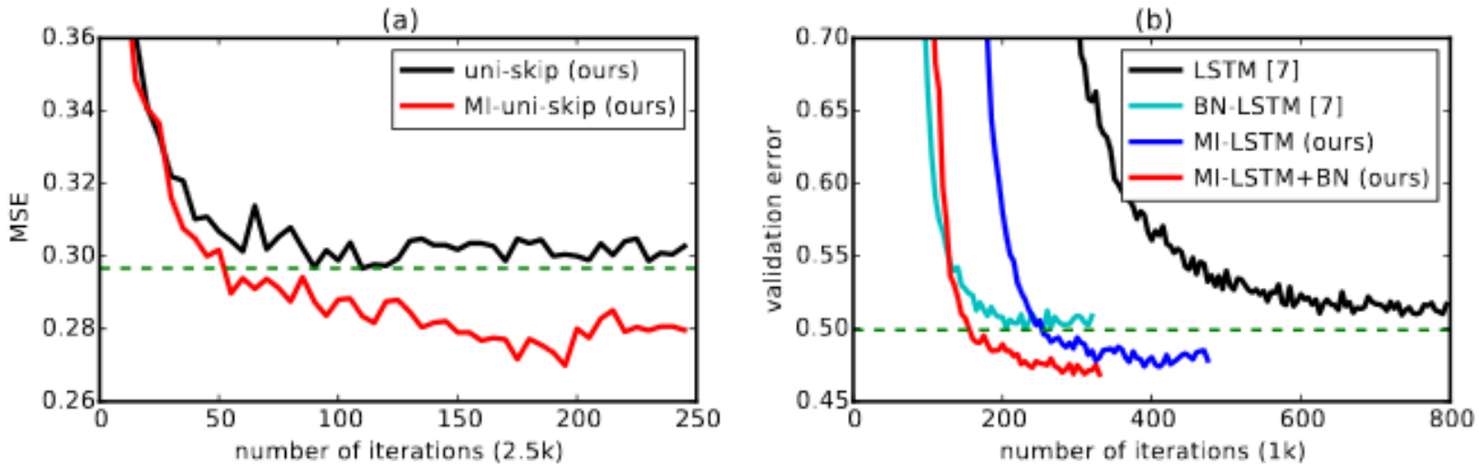
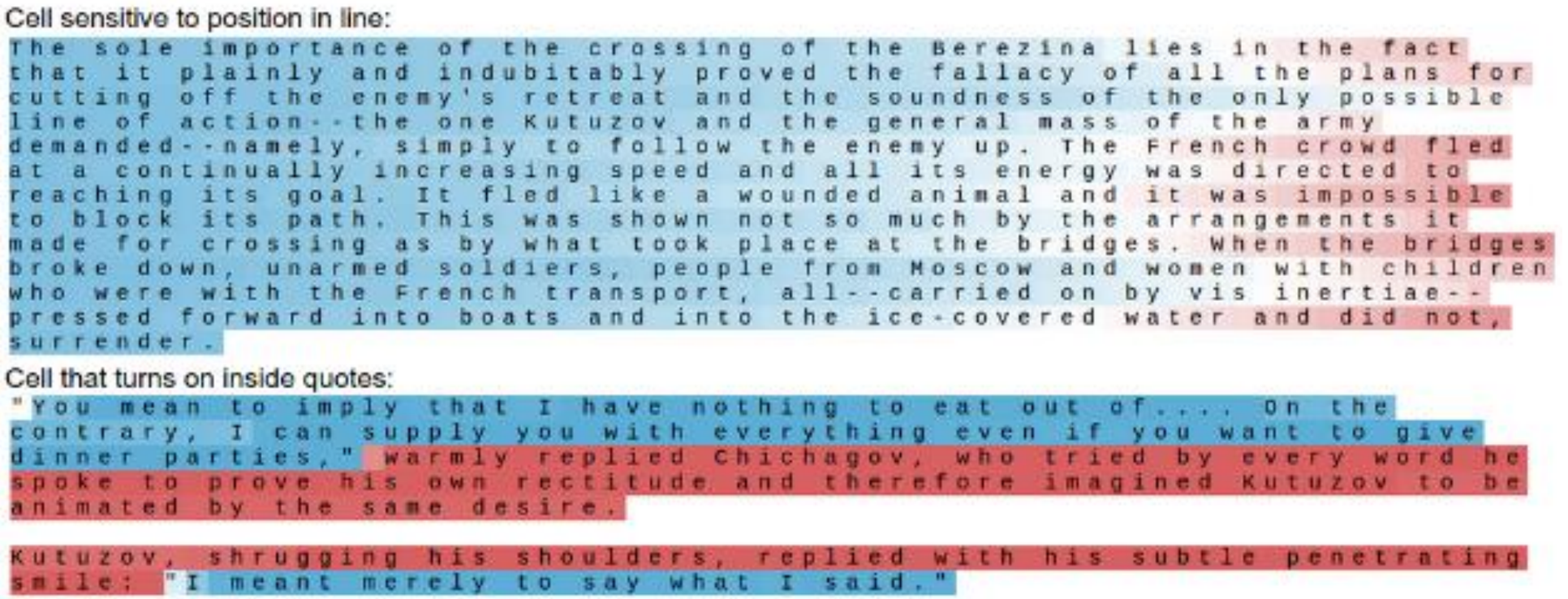


Figure 2: (a) MSE curves of uni-skip (ours) and MI-uni-skip (ours) on semantic relatedness task on SICK dataset. MI-uni-skip significantly outperforms baseline uni-skip. (b) Validation error curves on attentive reader models. There is a clear margin between models with and without MI.

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

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

RNN: советы

используйте GRU или LSTM

Специальные ортогональные инициализации рекуррентных матриц

forget gate ~ 1 (запоминать)

Adam

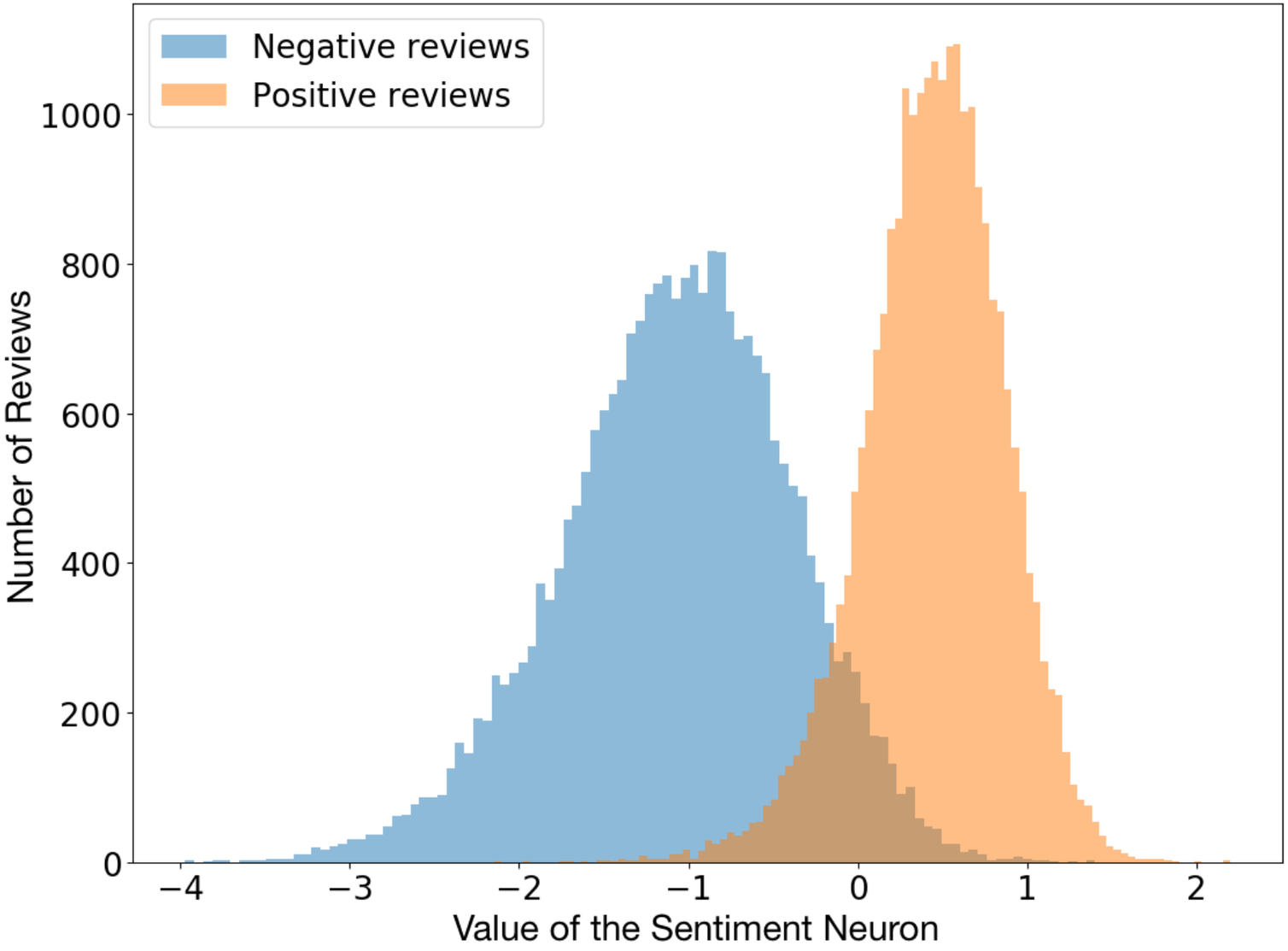
Clip ~ 1

начинайте с простых моделей (переобучитесь)

потом усложнение + регуляризация

смотрите на статистику по данным / по ответам модели

Интерпретация LSTM: Sentiment neuron



Интерпретация LSTM: Sentiment neuron

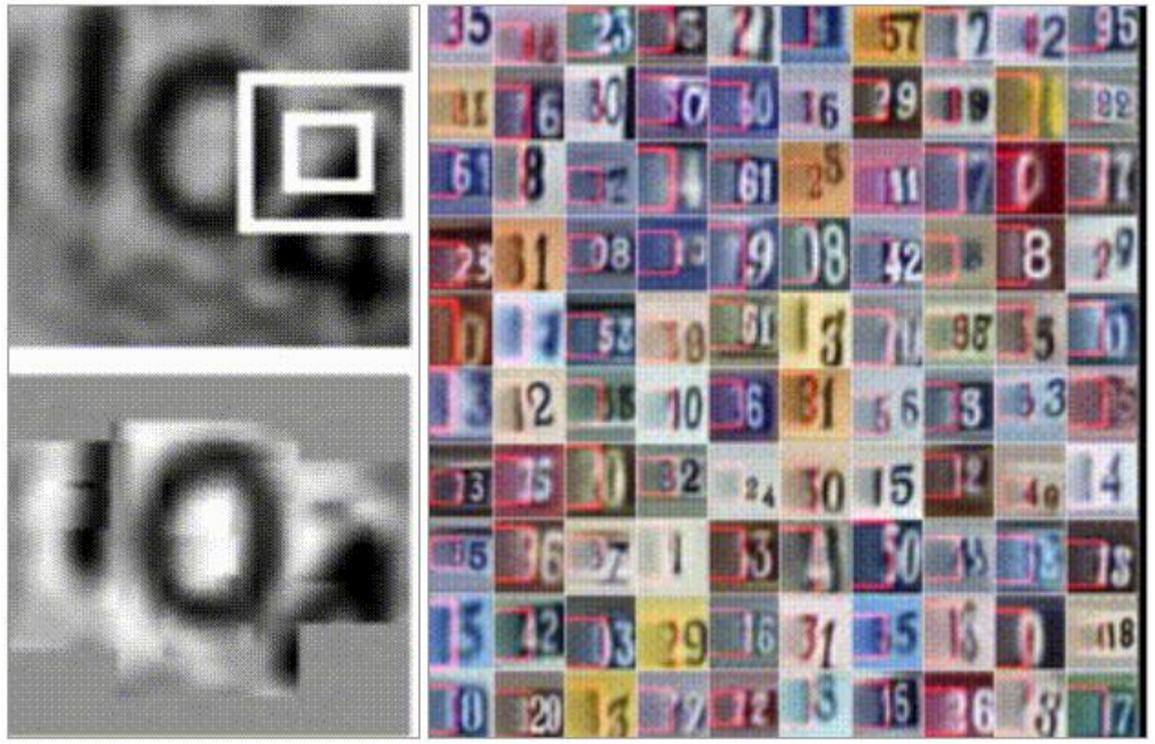
модель предсказывает следующий символ – возник нейрон, отвечающий за сентимент
если его значение фиксировать, то можно генерировать тексты с разным сентиментом

SENTIMENT FIXED TO POSITIVE	SENTIMENT FIXED TO NEGATIVE
Just what I was looking for. Nice fitted pants, exactly matched seam to color contrast with other pants I own. Highly recommended and also very happy!	The package received was blank and has no barcode. A waste of time and money.
This product does what it is supposed to. I always keep three of these in my kitchen just in case ever I need a replacement cord.	Great little item. Hard to put on the crib without some kind of embellishment. My guess is just like the screw kind of attachment I had.
Best hammock ever! Stays in place and holds it's shape. Comfy (I love the deep neon pictures on it), and looks so cute.	They didn't fit either. Straight high sticks at the end. On par with other buds I have. Lesson learned to avoid.
Dixie is getting her Doolittle newsletter we'll see another new one coming out next year. Great stuff. And, here's the contents - information that we hardly know about or forget.	great product but no seller. couldn't ascertain a cause. Broken product. I am a prolific consumer of this company all the time.
I love this weapons look . Like I said beautiful !!! I recommend it to all. Would suggest this to many roleplayers, And I stronge to get them for every one I know. A must watch for any man who love Chess!	Like the cover, Fits good. . However, an annoying rear piece like garbage should be out of this one. I bought this hoping it would help with a huge pull down my back & the black just doesn't stay. Scrap off everytime I use it.... Very disappointed.

https://openai.com/blog/unsupervised-sentiment-neuron/

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

Итог

Рекуррентность в DL – ещё один пример разделения весов

Много проблем с памятью, градиентом и обучением

Есть много архитектур

GRU считается быстрее, LSTM мощнее (но это условно)

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/