

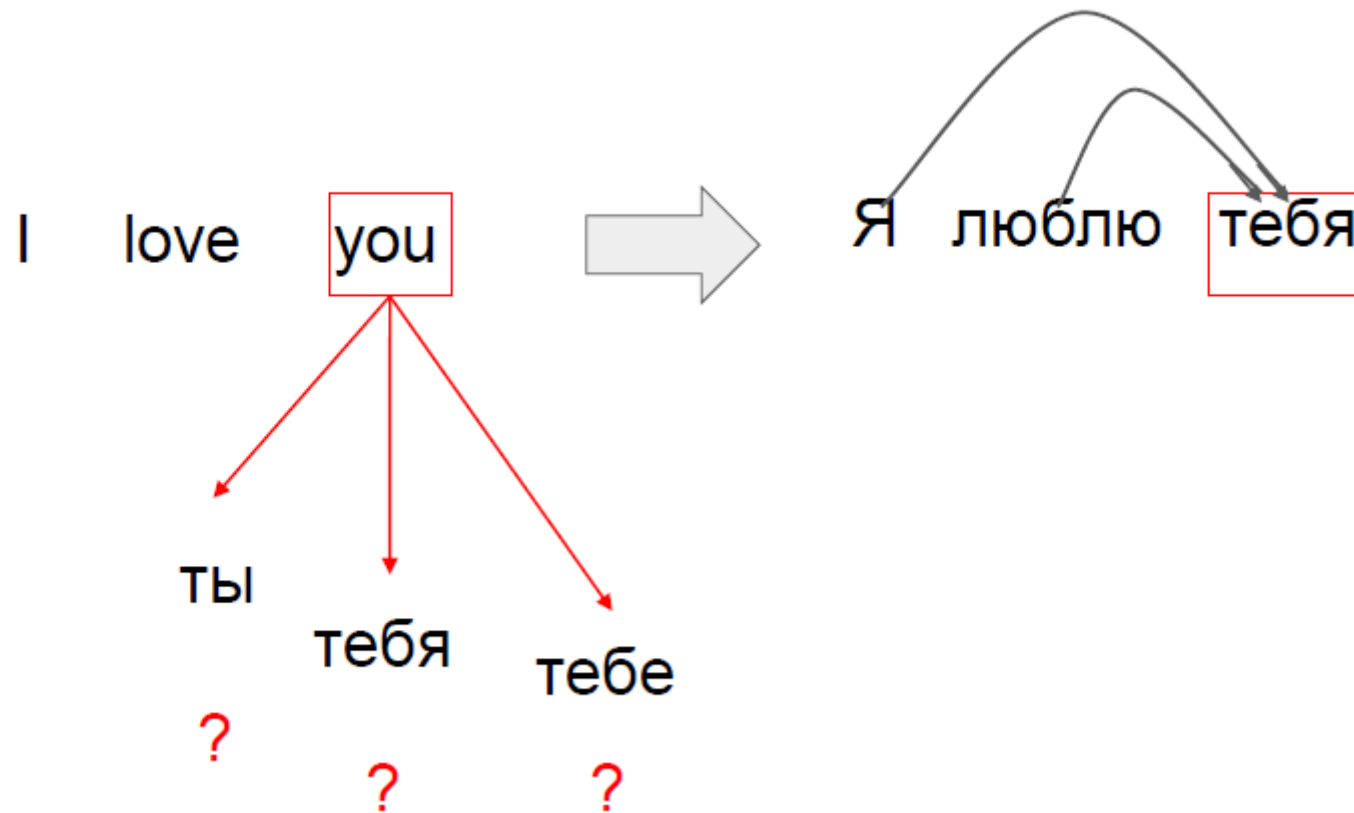
Рекуррентные нейронные сети (RNN)

- Рекуррентные сети - это семейство моделей для обработки **последовательных** данных.
- Используют концепцию совместного использования параметров, а также свёрточные сети.
- Могут обрабатывать **ввод различной длины**.

Приложения рекуррентных нейронных сетей

- Прогнозирование временных рядов
- Управление технологическими процессами
- Классификация текстов или их фрагментов
- Анализ тональности документа / предложений / слов
- Машинный перевод
- Распознавание речи
- Синтез речи
- Синтез ответов на вопросы, разговорный интеллект
- Генерация подписей к изображениям
- Генерация рукописного текста
- Интерпретация генома и другие задачи биоинформатике

Машинный перевод



RNN

x_t — входной вектор в момент t

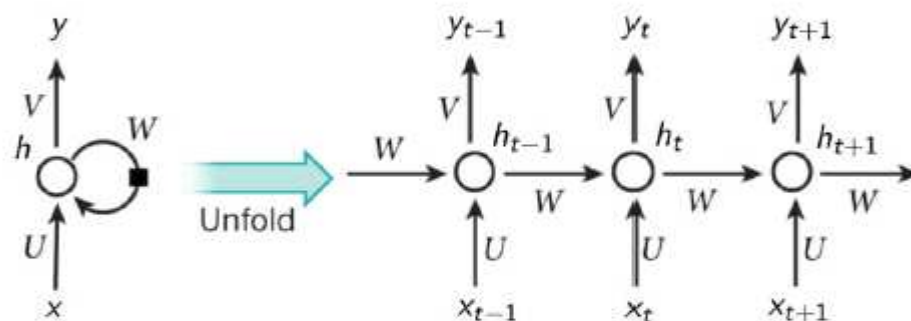
h_t — вектор скрытого состояния в момент t

y_t — выходной вектор (в некоторых приложениях $y_t \equiv h_t$)

Разворачивание (unfolding) рекуррентной сети

$$h_t = \sigma_h(Ux_t + Wh_{t-1})$$

$$y_t = \sigma_y(Vh_t)$$



Обучение рекуррентной сети:

$$\sum_{t=0}^T \mathcal{L}_t(U, V, W) \rightarrow \min_{U, V, W}$$

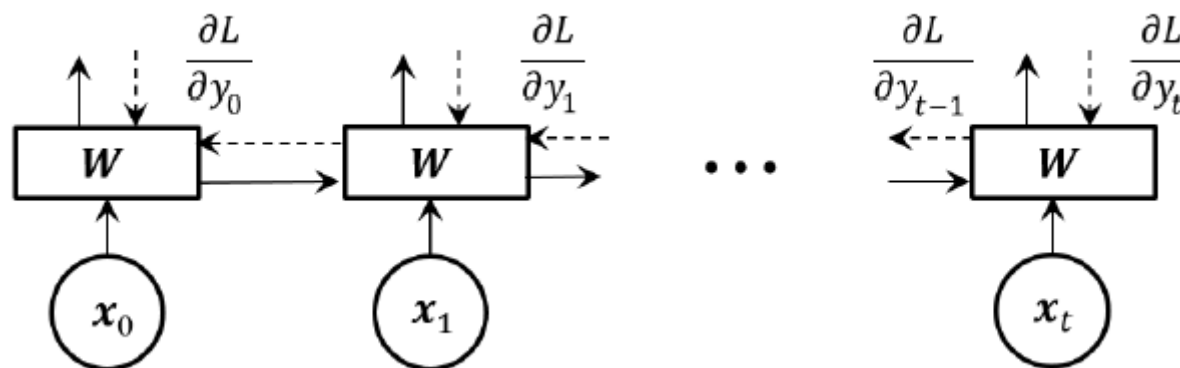
$\mathcal{L}_t(U, V, W) = \mathcal{L}(y_t(U, V, W))$ — потеря от предсказания y_t

Обучение RNN

Используется градиентный спуск + Backpropagation Through Time (BPTT)

- Ошибка сети есть сумма ее ошибок для всех t
- Общий градиент по переменной есть сумма ее градиентов для всех t

$$\frac{\partial L_y}{\partial W} = \frac{\partial L_y}{\partial y_t} \frac{\partial y_t}{\partial h_t} \frac{\partial h_t}{\partial W}$$



Найдем производную функции потерь L_y

Т.к. h_t зависит от W, h_{t-1}, \dots, h_0 , то это производная сложной функции нескольких переменных:

$$\frac{\partial L_y}{\partial W} = \frac{\partial L_y}{\partial y_t} \frac{\partial y_t}{\partial h_t} \left(\frac{\partial h_t}{\partial W} + \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial W} + \frac{\partial h_t}{\partial h_{t-2}} \frac{\partial h_{t-2}}{\partial W} + \dots + \frac{\partial h_t}{\partial h_0} \frac{\partial h_0}{\partial W} \right)$$

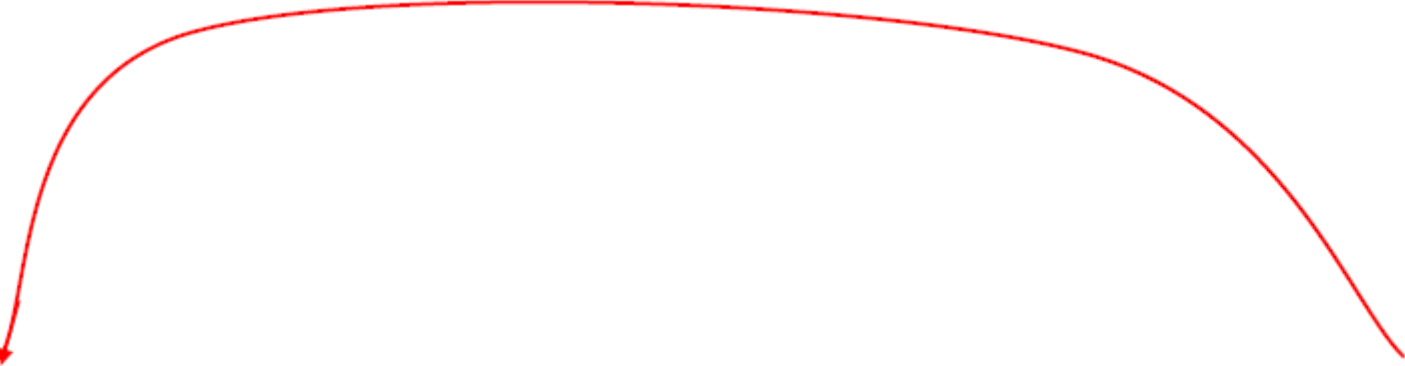
Производную $\frac{\partial h_t}{\partial h_k}$ можно представить в виде:

$$\frac{\partial h_t}{\partial h_k} = \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial h_{t-2}} \frac{\partial h_{t-2}}{\partial h_{t-3}} \dots \frac{\partial h_{k+1}}{\partial h_k} = \prod_{j=k+1}^t \frac{\partial h_j}{\partial h_{j-1}}$$

Тогда

$$\frac{\partial L_y}{\partial W} = \frac{\partial L_y}{\partial y_t} \frac{\partial y_t}{\partial h_t} \sum_{k=0}^t \left(\prod_{j=k+1}^t \frac{\partial h_j}{\partial h_{j-1}} \right) \frac{\partial h_k}{\partial W}$$

Проблема обучения RNN

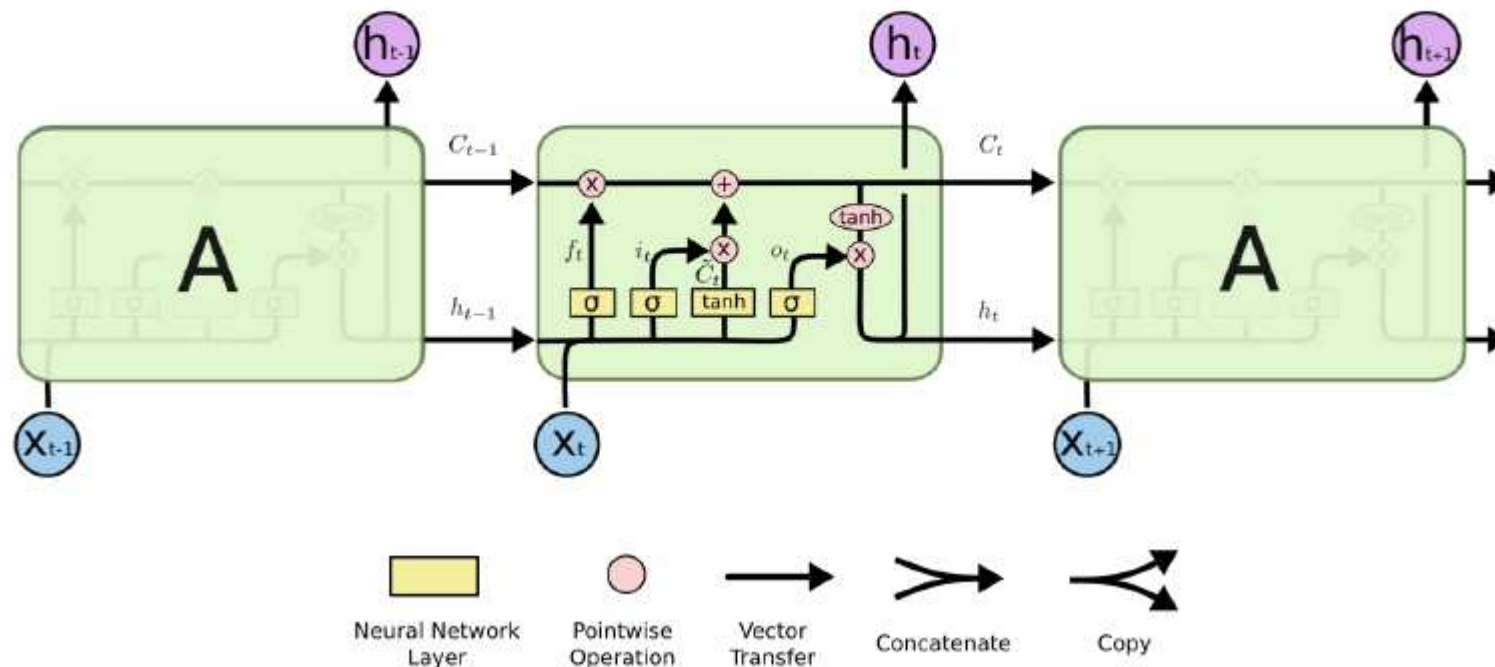


Человек, на голове которого была шляпа с павлиньим пером, зашел в бар

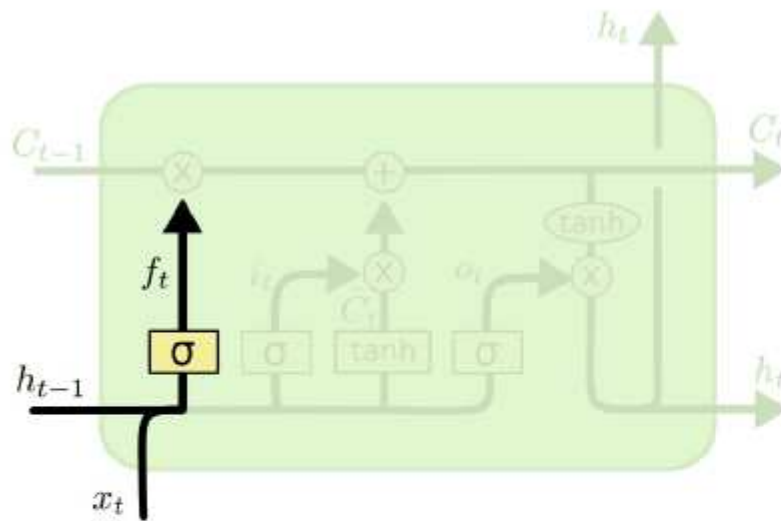
Long short-term memory (LSTM)

Мотивация LSTM: сеть должна долго помнить контекст, какой именно — сеть должна выучить сама.

Вводится C_t — вектор состояния сети в момент t .



LSTM



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

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

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

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

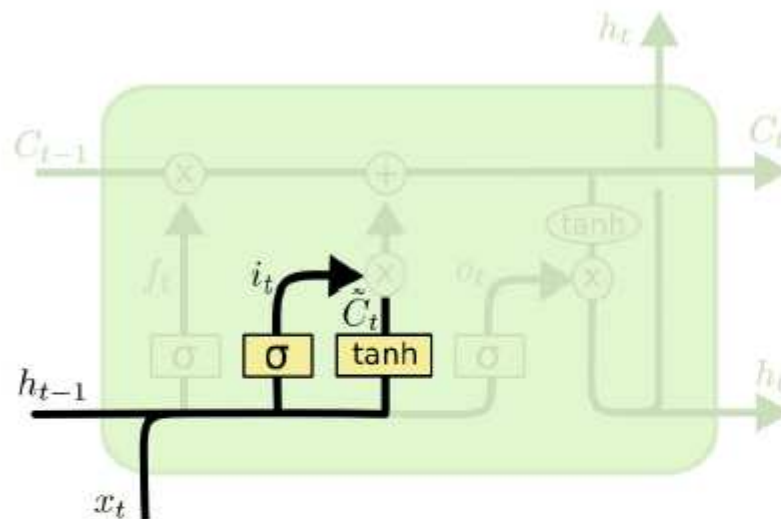
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

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

Фильтр забывания (forget gate) с параметрами W_f , b_f решает, какие координаты вектора состояния C_{t-1} надо запомнить.

\odot — операция покомпонентного перемножения векторов,
 $[h_{t-1}, x_t]$ — конкатенация векторов,
 σ — сигмоидная функция.

LSTM



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

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

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

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

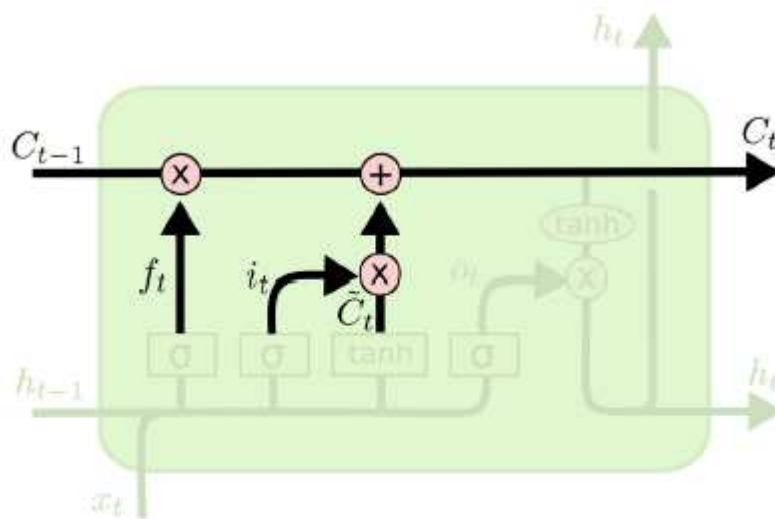
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \odot \text{th}(C_t)$$

Фильтр входных данных (input gate) с параметрами W_i , b_i решает, какие координаты вектора состояния надо обновить.

Модель нового состояния с параметрами W_C , b_C формирует вектор \tilde{C}_t значений-кандидатов нового состояния.

LSTM



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

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

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

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

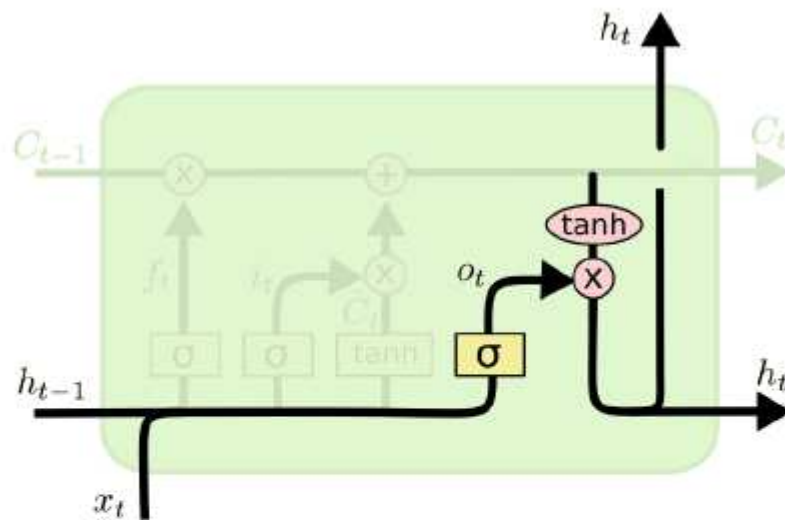
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \odot \text{th}(C_t)$$

Новое состояние C_t формируется как смесь старого состояния C_{t-1} с фильтром f_t и вектора значений-кандидатов \tilde{C}_t с фильтром i_t .

Настраиваемых параметров нет.

LSTM

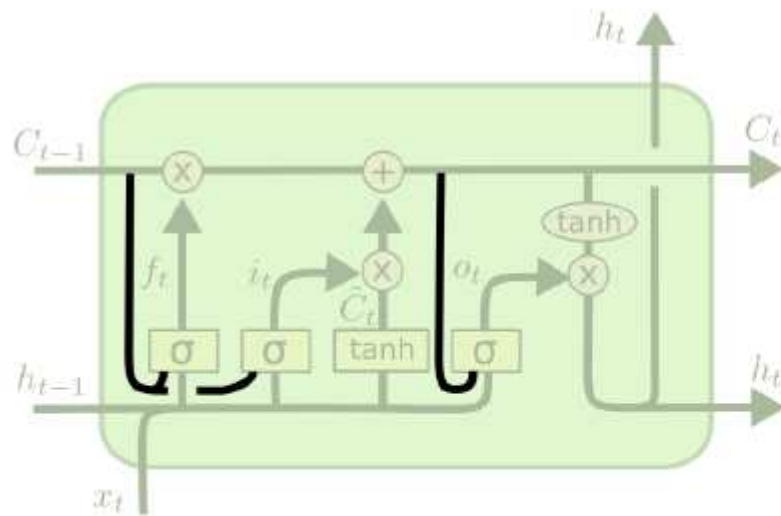


$$\begin{aligned}f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\\tilde{C}_t &= \text{th}(W_C \cdot [h_{t-1}, x_t] + b_C) \\C_t &= f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \\o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\h_t &= o_t \odot \text{th}(C_t)\end{aligned}$$

Фильтр выходных данных (output gate) с параметрами W_o , b_o решает, какие координаты вектора состояния C_t надо выдать.

Выходной сигнал h_t формируется из вектора состояния C_t с помощью нелинейного преобразования th и фильтра o_t .

Модификация LSTM



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

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

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

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

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

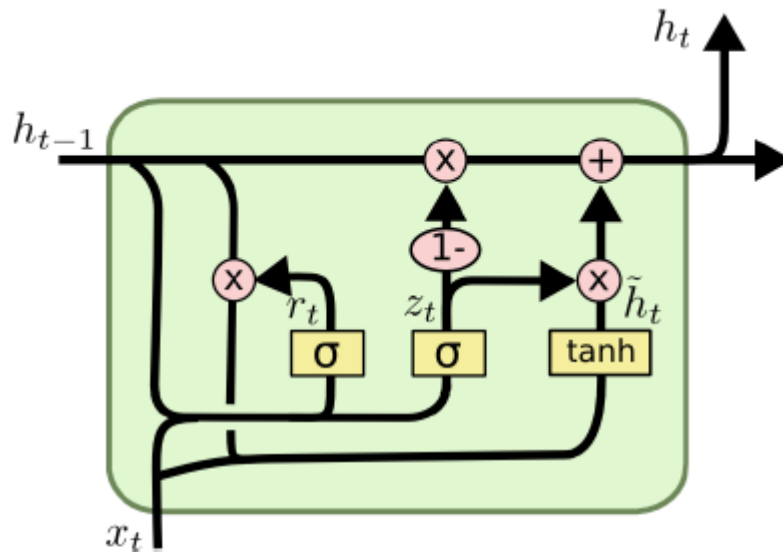
$$h_t = o_t \odot \text{th}(C_t)$$

Все фильтры «подглядывают» вектор состояния C_{t-1} или C_t .

Увеличивается число параметров модели.

Замочную скважину можно использовать не для всех фильтров.

Gated Recurrent Unit



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

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

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

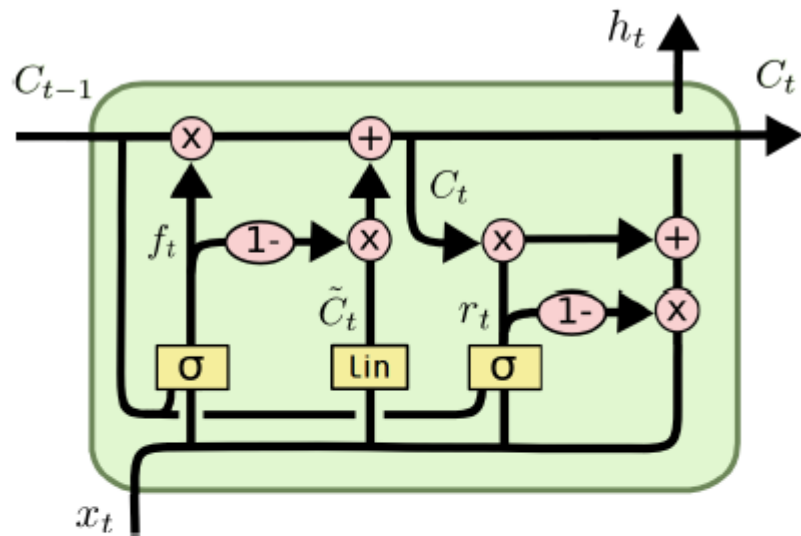
$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

Используется только состояние h_t , вектор C_t не вводится.

Фильтр обновления (update gate) вместо входного и забывающего.

Фильтр перезагрузки (reset gate) решает, какую часть памяти нужно перенести дальше с прошлого шага.

Simple Recurrent Unit



$$f_t = \sigma(W_f x_t + v_f \odot C_{t-1} + b_f)$$

$$\tilde{C}_t = W_C x_t$$

$$C_t = f_t \odot C_{t-1} + (1 - f_t) \odot \tilde{C}_t$$

$$r_t = \sigma(W_r x_t + v_r \odot C_{t-1} + b_r)$$

$$h_t = r_t \odot C_t + (1 - r_t) \odot x_t$$

С предыдущего шага передаётся только вектор C_{t-1} .

Два фильтра: забывания (forget gate) и перезагрузки (reset gate).

Сквозные связи (skip connections): x_t передаётся на все слои.

Облегчённая рекуррентность: $v_f \odot C_{t-1}$ вместо $W_f C_{t-1}$,
позволяет вычислять координаты векторов параллельно.