Обзор LSTM, GRU и Recurrent Sigmoid Piecewise

Общие сведения

При работе с LSTM, GRU и другими рекуррентными нейронами/сетями часто используются следущие "сущности":

- вектор входов x_t - определяет входы нейрона/сети на момент времени t; в задаче прогнозирования это обычно значение(я) одного либо нескольких временных рядов в момент(ы) времени t $(, t-1, t-2, ...)$
- вектор выходов h_t - определяет выход нейрона/сети на момент временеи t; в задаче прогнозирования это обычно прогноз значения основного временнного ряда в момент времени $t+i$ либо промежуточный вектор, который используется для получения окончательного прогноза
- вектор "контекста" c_t - контекст можно описать как "информацию о прошлом, закодированную в вектор фиксированной размерности"

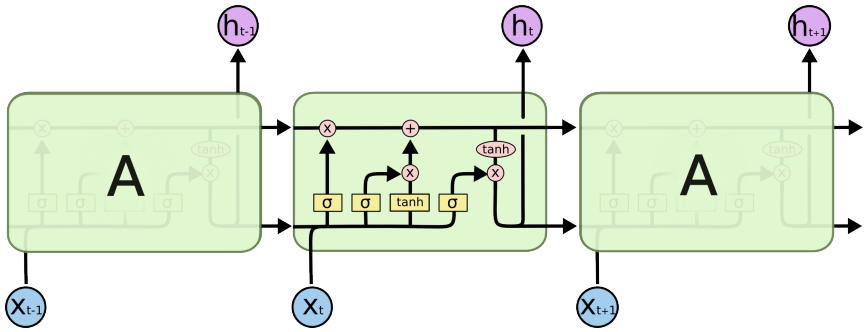
В каждый момент времени t нейрон/сеть принимает вектор входов x_t, векторы контекста и выходов с **предыдущего** "шага" c_{t-1}, h_{t-1} и выдает **новые** векторы контекста и выходов c_t, h_t. Таким образом, вектор контекста c_t содержит некоторую информацию про все входы $x_1, ..., x_t$ так как является результатом рекуррентных вызовов нейрона/сети:

$$c_t = f(x_t, c_{t-1}, h_{t-1}); c_{t-1} = f(x_{t-1}, c_{t-2}, h_{t-2}); ...c_1 = f(x_1, c_0, h_0)$$

где c_0, h_0 - начальные векторы контекста и выходов, обычно задаются как некоторые фиксированные векторы, например нулевые векторы. При этом, в большинстве случаев происходит "затухание" информации - то есть чем больше разница $t-i, i < t$ - тем меньше информации про вход x_i содержится в контексте c_t.

LSTM

"Классический" LSTM-нейрон имеет следующую структуру:



Полное математическое описание классического LSTM нейрона:

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

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

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

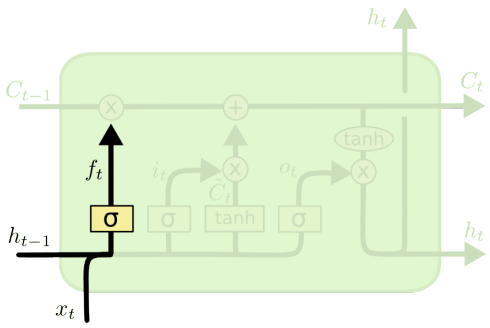
$$C_t = f_t * C_{t-1} + i_t * \tilde{C_t}$$

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

$$h_t = o_t * tanh(c_t)$$

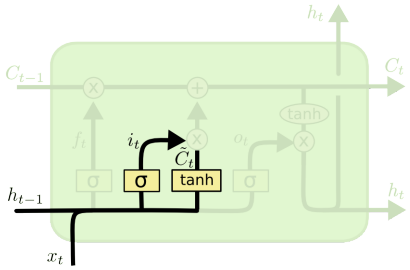
Основные блоки LSTM нейрона:

- Шлюз "забывания" (forget gate) $f_t(h_{t-1}, x_t; W_f, b_f)$ - принимает на вход вектор входов x_t, предыдущий выходной вектор h_{t-1} и выдает вектор f_t с той же размерностью, что и у вектора контекста, и значениями в интервале $(0, 1)$. Выход этого блока используется для "взвешивания"значений предыдущего вектора контекста c_{t-1}, где значения, умноженные на вес, близкий к 0, “забываются“.



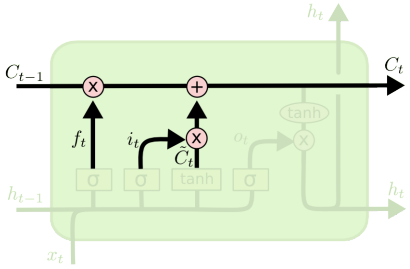
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

- Шлюз "обновления" (update gate) $i_t(h_{t-1}, x_t; W_i, b_i)$ - аналогичен шлюзу забывания, но веса этого шлюза используются для взвешивания значений вектора-кандидата для нового контекста.
- Блок подсчета вектора-кандидата для нового контекста $\tilde{C}_t(h_{t-1}, x_t; W_C, b_C)$ - простой слой из $tanh$ нейронов.



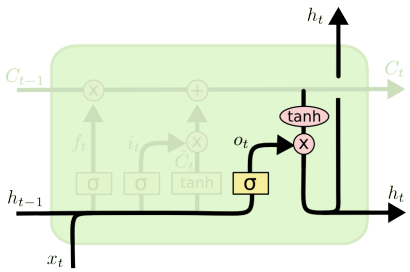
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

- Блок подсчета нового контекста $C_t(f_t, i_t, C_{t-1}, \tilde{C}_t)$ - простой блок, который взвешивает и суммирует значения предыдущего контекста C_{t-1} и кандидата $\tilde{C}_t$ для получения текущего вектора контекста c_t.



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

- Выходной шлюз (output gate) $o_t(h_{t-1}, x_t; W_o, b_o)$ - аналогичен шлюзу забывания, но используется для определения весов выходного вектора h_t.
- Блок подсчета выходного вектора $h_t(o_t, C_t)$ - простой блок, подсчитывающий значения выходного вектора h_t путем взвешивания трансформированных значений текущего контекста C_t.



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

Рассмотрим упрощенный пример использования LSTM нейрона для прогнозирования:

- Входная последовательность временного ряда: $[1, 2, 3, 4, 5]$.
- Размерность векторов контекста и выходов LSTM нейрона: 3, и окончательный значение прогноза будет получаться путем применения некоторой функции $g(h_t)$.
- Размерность вектора входов: 2, таким образом нейрон будет вызываться на следующих входных векторах: $x_1 = [1, 2], x_2 = [2, 3], x_3 = [3, 4], x_4 = [4, 5]$, размерность вектора $[h_{t-1}, x_t]$ будет равна 5=3 + 2, размерность матриц W_f, W_i, W_C, W_o будет равна 3×5, размерность векторов b_f, b_i, b_C, b_o будет равна 3.
- Задается начальный выходной вектор $h_0 = [0, 0, 0]$ и начальный вектор контекста. $c_0 = [0, 0, 0]$
- Первый "вызов" LSTM нейрона:
  - Рассчитываются значения шлюза забывания $f_1(h_0, x_1; W_f, b_f)$, пускай $f_1 = [1, 1, 1]$.
  - Рассчитываются значения шлюза обновления $i_1(h_0, x_1; W_i, b_i)$, пускай $i_1 = [0.2, 0.1, 0.9]$.
  - Рассчитываются значения вектора-кандидата нового контекста $\tilde{C_1}(h_0, x_1; W_C, b_C)$, пускай $\tilde{C_1} = [0.3, 0.1, 0.2]$.
  - Рассчитывается новый вектор контекста C_1 как $C_1 = f_1 * C_0 + i_1 * \tilde{C_1}$:

    $$C_1 = [1, 1, 1]*[0, 0, 0]+[0.2, 0.1, 0.9]*[0.3, 0.1, 0.2] = [0.06, 0.01, 0.18]$$

  - Рассчитываются значения выходного шлюза $o_1(h_0, x_1; W_o, b_o)$ пускай $o_1 = [0.9, 0.5, 0.5]$
  - Рассчитывается выходной вектор $h_1(C_1; o_1)$:

    $$h_1 = [0.9, 0.5, 0.5] * tanh([0.06, 0.01, 0.18]) = [0.054, 0.005, 0.089]$$

  - Рассчитывается финальное значение прогноза $y_1 = g(h_1)$, пускай $y_1 = 2.9$, при обучении это значение будет использоваться в функции ошибки для рассчета градиентов и обновления весов нейрона.
- Второй вызов LSTM нейрона:
  - Рассчитываются новые значения шлюза забывания $f_2(h_1, x_2; W_f, b_f)$, при этом используются второй входной вектор $x_2 = [2, 3]$ и предыдущий выходной вектор $h_1 = [0.054, 0.005, 0.089]$, пускай $f_1 = [0.9, 0.1, 1.0]$.

- Рассчитываются новые значения шлюза обновления $i_2(h_1, x_2; W_i, b_i)$, пускай $i_2 = [0.2, 0.9, 0.2]$.
- Рассчитываются новые значения вектора-кандидата нового контекста $\tilde{C}_2(h_1, x_2; W_C, b_C)$, пускай $\tilde{C}_2 = [-0.6, 0.8, -0.3]$.
- Рассчитывается новый вектор контекста C_2 как $C_2 = f_2 * C_1 + i_2 * \tilde{C}_2$, где $C_1 = [0.06, 0.01, 0.18]$ это предыдущий вектор контекста полученный на первом вызове:

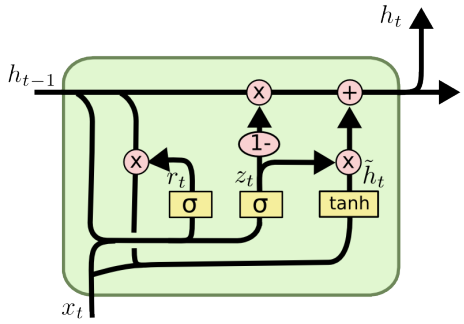
  $C_2 = [0.9, 0.1, 1.0] * [0.06, 0.01, 0.18] + [0.2, 0.9, 0.2] * [-0.6, 0.8, -0.3] = [-0.066, 0.721, 0.12]$
- Рассчитываются значения выходного шлюза $o_2(h_1, x_2; W_o, b_o)$ пускай $o_2 = [0.1, 0.4, 0.8]$
- Рассчитывается выходной вектор $h_2(C_2; o_1)$:

  $h_2 = [0.1, 0.4, 0.8] * tanh([-0.066, 0.721, 0.12]) = [-0.0066, 0.2470, 0.0955]$
- Рассчитывается финальное значение прогноза $y_2 = g(h_2)$, пускай $y_2 = 4.1$.

- Третий и четвертый вызовы нейрона происходят аналогично второму вызову, но с использованием входов x_3, x_4, контекстов c_2, c_3 и выходов h_2, h_3 соответственно.
- При обучении нейрона, полученные финальные прогнозы $y_1, ..., y_4$ сравниваются с ожидаемыми, и суммарная ошибка используется для подсчета градиентов и обновления весов.

GRU

GRU нейрон это, по сути, упрощенная версия LSTM нейрона:



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

В данном нейроне вектор выходов h_t так же "выполняет" роль вектора контекста, и используются следующие блоки:

- Блок обновления $z_t(x_t, h_{t-1}; W_z)$, рассчитывающий веса в диапазоне $(0, 1)$, которые применяются для рассчета нового вектора выходов (и, одновременно, контекста) h_t исходя из вектора-кандидата $\tilde{h}_t$ и предыдущего вектора h_{t-1}

- Блок "релевантности" $r_t(x_t, h_{t-1}; W_r)$, рассчитывающий веса в диапазоне $(0, 1)$, которые определяют "релевантность"/"важность" значений предыдущего выходного вектора h_{t-1} при рассчете вектора-кандидата для нового выходного вектора $\tilde{h}_t$
- Блок рассчета вектора-кандидата новых выходов $\tilde{h}_t(x_t, h_{t-1}, r_t; W)$
- Блок рассчета нового вектора выходов $h_t(h_{t-1}, \tilde{h}_t, z_t)$ как взвешенной суммы соответствующих значений из предыдущего вектора h_{t-1} и нового вектора-кандидата $\tilde{h}_t$, где веса для значений под индексом i выбираются как $1 - z_t[i]$ и $z_t[i]$ соответственно

Recurrent Sigmoid Piecewise (RSP)

Sigmoid Piecewise (SP) нейрон имеет следующую математическую модель:

$$SP(x; w_+, w_-, s, k) = \frac{w_+ \cdot x}{1 + e^{-k(s \cdot x)}} + \frac{w_- \cdot x}{1 + e^{k(s \cdot x)}}$$

Используя обозначение сигмоидального нейрона:

$$\sigma(x; s) = \frac{1}{1 + e^{s \cdot x}}$$

и $k = 1$ получаем:

$$SP(x; w_+, w_-, s) = \sigma(x; s)(w_+ \cdot x) + \sigma(x; -s)(w_- \cdot x)$$

Используя равенство $\sigma(x; -s) = 1 - \sigma(x; s)$:

$$SP(x; w_+, w_-, s) = (1 - \sigma(x; s))(w_- \cdot x) + \sigma(x; s)(w_+ \cdot x)$$

Если вместо одного SP нейрона описывается слой из N нейронов, то вместо векторов w_+, w_-, s будут использоваться матрицы W_+, W_-, S:

$$SP(x; W_+, W_-, S) = (1 - \sigma(x; S)) * (W_- \cdot x) + \sigma(x; S) * (W_+ \cdot x)$$

Введя обозначения $z = \sigma(x; S)$, $a = W_- \cdot x$ и $b = W_+ \cdot x$ получаем:

$$SP(x) = (1 - z) * a + z * b$$

Что очень похоже на блок рассчета нового вектора выходов в нейроне GRU:

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

Таким образом, слегка изменив SP нейрон, можно получить его рекурретную версию, Recurrent Sigmoid Piecewise (RSP) нейрон, который принимает на вход вектор $p_t = [h_{t-1}, x_t]$ и выдает h_t:

$$h_t = RSP(p_t; W_+, W_-, S) = (1 - \sigma(p_t; S)) * (W_- \cdot p_t) + \sigma(p_t; S) * (W_+ \cdot p_t)$$

Либо же, по аналогии с LSTM/GRU нейронами, мат. модель RSP нейрона можно записать в несколько этапов/блоков:

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

$$q_t = W_- \cdot [h_{t-1}, x_t]$$

$$\tilde{h}_t = W_+ \cdot [h_{t-1}, x_t]$$

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

В такой записи сходство RSP и GRU нейронов особенно заметно.