

人工智能之深度学习

RNN--讲义

主讲人：Vincent Ying

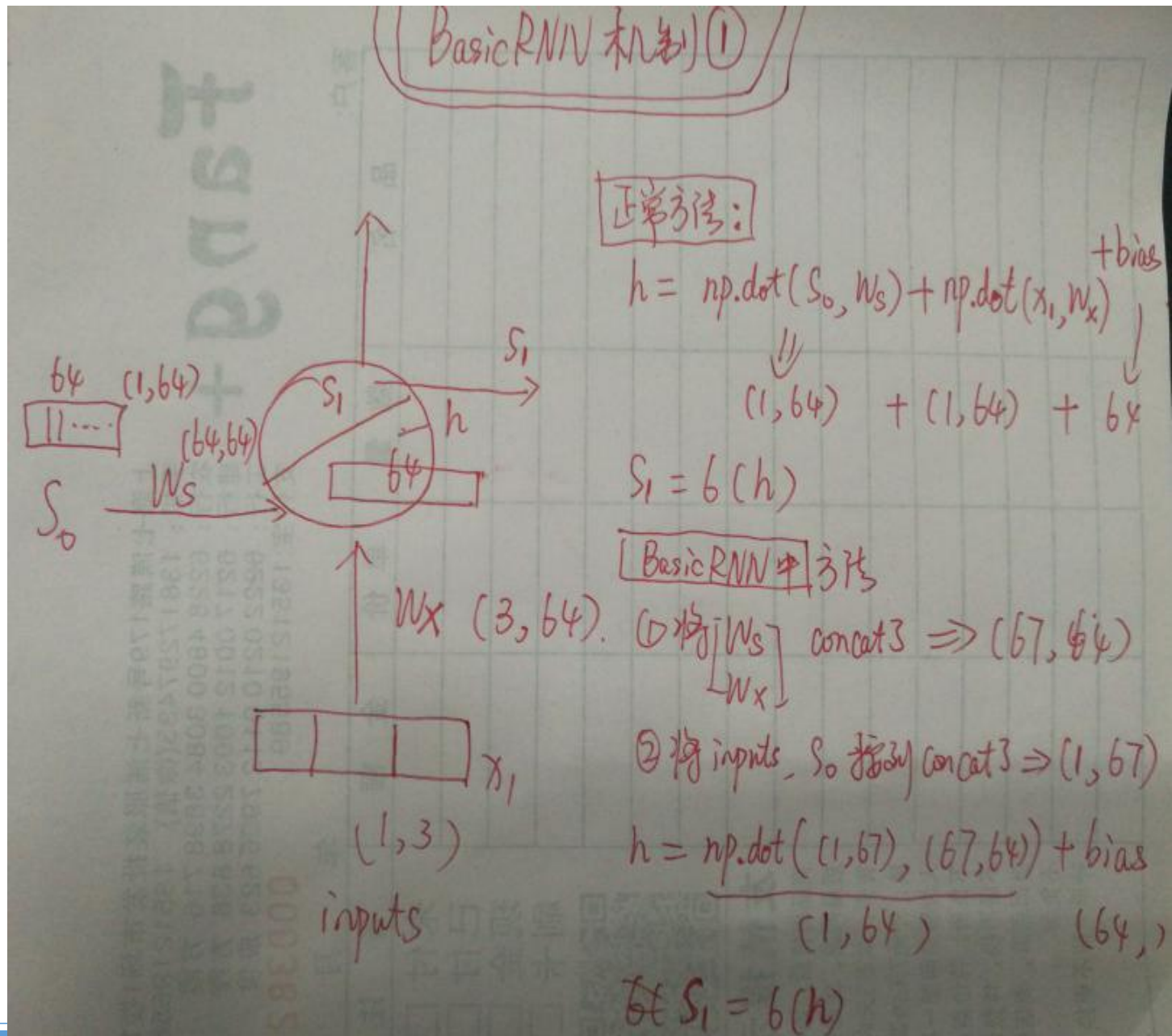
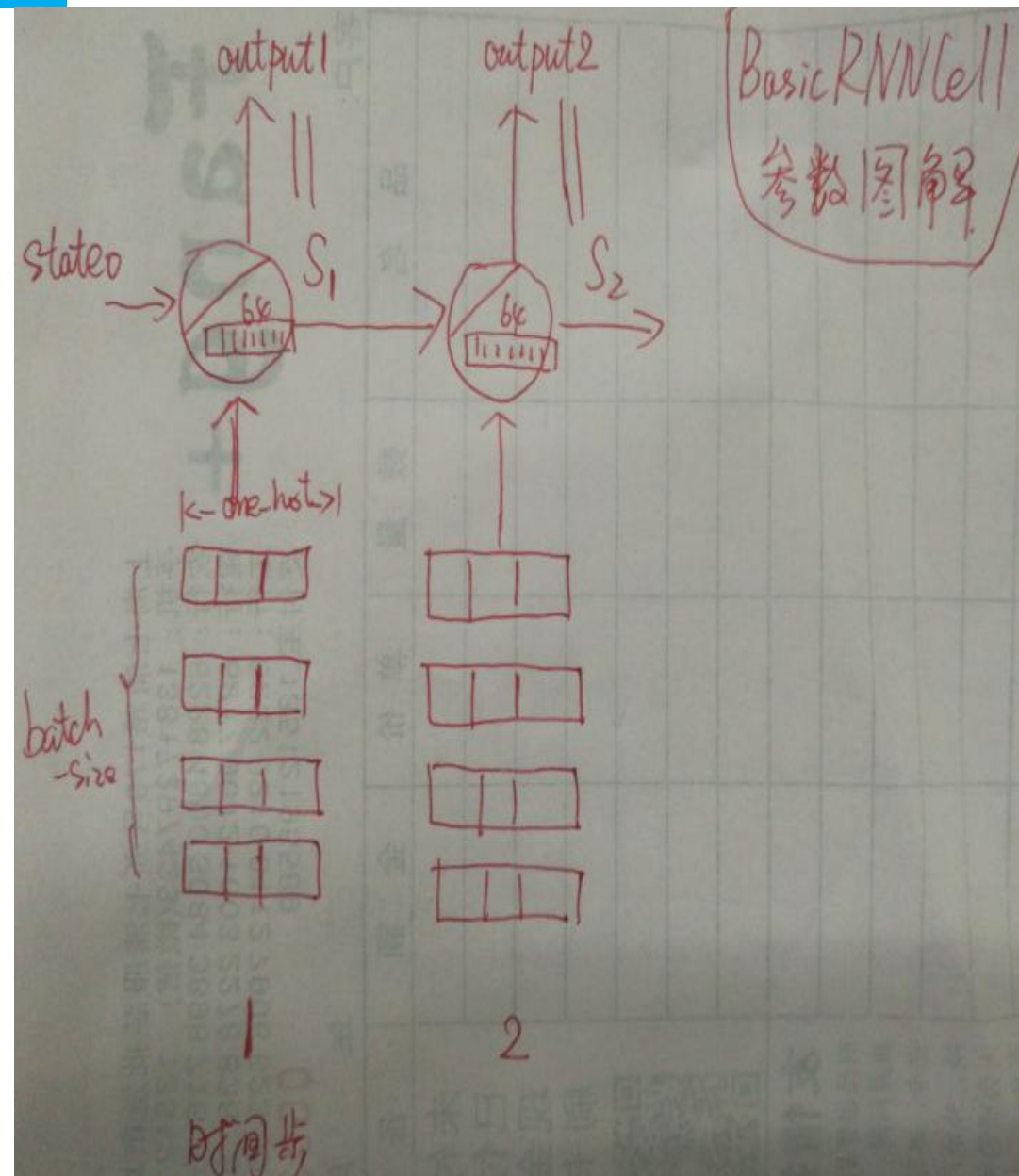
上海育创网络科技有限公司



课程内容

- 一、递归神经网络(RNN)
- 二、RNN反向传播
- 三、Word2vec
- 四、双向RNN
- 五、LSTM
- 六、RNN超参数

01 RNN的基本API



input_shape.
[Batch_size, n_step]

① 初始化:

Weights : $\begin{bmatrix} n_step + 1 & num_units \\ 3 & 4 \end{bmatrix}$, $\begin{bmatrix} num_units \\ 4 \end{bmatrix}$

Bias : $\begin{bmatrix} num_units \end{bmatrix}$

② call 方法:

$\text{matmul}(\text{concat}([inputs, state], 1), \text{Weights}) + \text{Bias}$

⚠️ tanh 激活函数 / 按列拼

shape = $\begin{bmatrix} batch_size, n_step + num_units \end{bmatrix}$ \otimes $\begin{bmatrix} num_units \\ 4 \end{bmatrix}$ Weights

\Downarrow

$\begin{bmatrix} batch_size, num_units \end{bmatrix} + \text{Bias}$

BasicRNN

LSTMcell 机制

Lstm Cell

Input_shape = [Batch_size, ~~num-unit~~]

① 初始化 W, b one-hot 与 embedded.

$W : [\text{one-hot} + \text{num-unit}, 4 \times \text{num-unit}]$

$bias : [4 \times \text{num-unit}]$

state 为 tuple, shape 为 $(\underbrace{[batch_size, num_unit]}_c, \underbrace{[batch_size, num_unit]}_h)$

② Call

$gate_input = tf.matmul(\underbrace{concat([inputs, h], 1)}_{\substack{[B, one-hot] \quad [B, num-unit]}}, W) + bias$

matmul 结果为: $[Batch_size, 4 \times num_unit]$

$i, f, o = tf.split(gate_input, 4, axis=1)$
分成 4 份 (向量)

$new_c = c \otimes \sigma(f + forget_bias_tensor) + \sigma(i) \otimes \tanh(j)$

$new_h = \tanh(new_c) \otimes \sigma(o)$

$new_state = (new_c, new_h)$

(static rnn 机制)

① 初始化 state

(这是一个 list)

② for time, input in enumerate(inputs):

reuse-variable() 重用参数

④ cell_cell = lambda: cell(input_, state)

↓

② 这是 LSTM 实例, 有 cell 方法, 做的是 LSTM 正向传播返回 (new-h, new-state)

(out_put, state) = _rnn_step(返回)

③

outputs.append(out_put)

步马聚

→ 计算动态 RNN 小批量的一个

最终返回 (outputs, state)

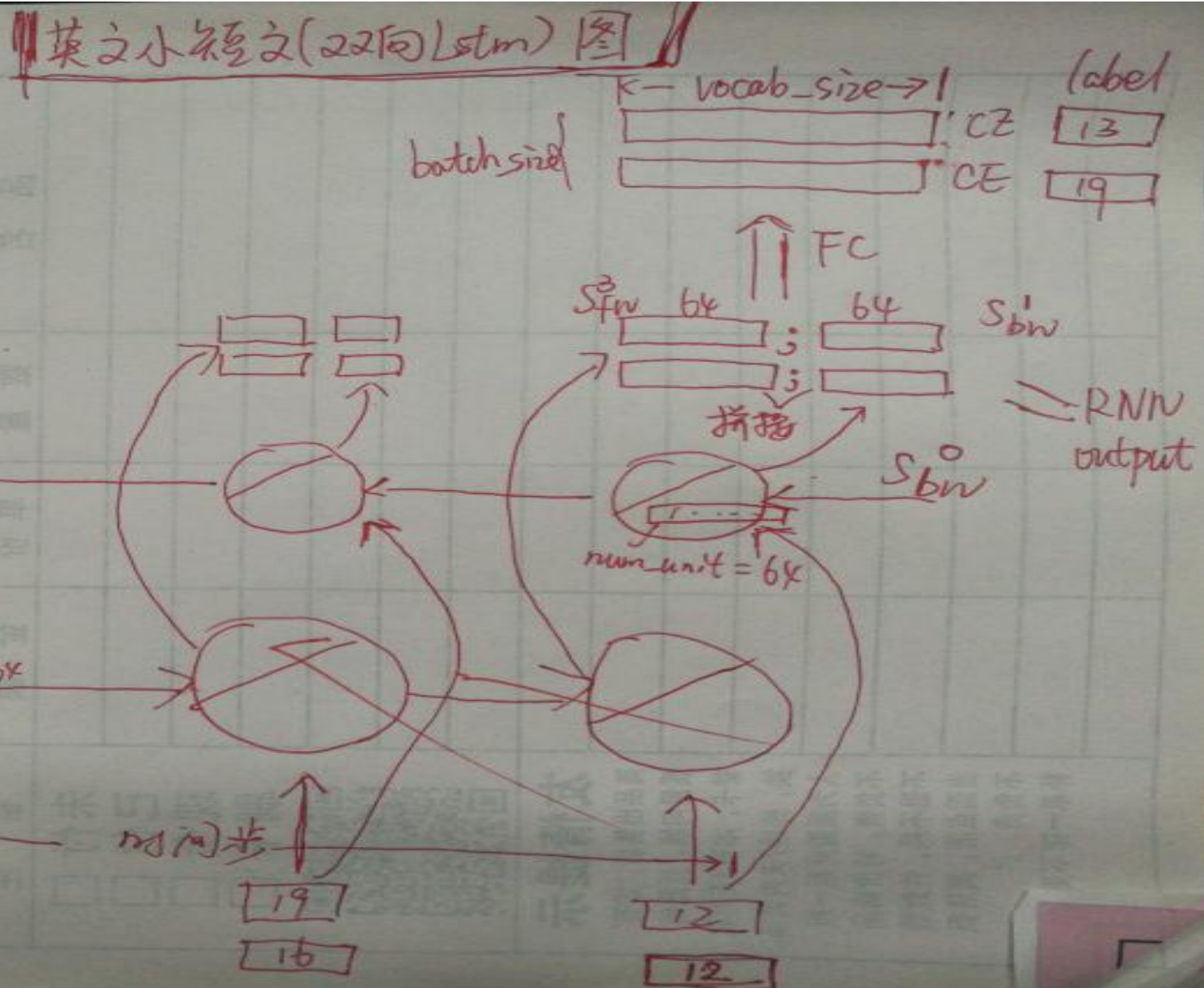
↓

list of outputs

= final state

英文小短文预测 (双向LSTM)

英文小短文预测 (双向LSTM) 模型图



static_bidirectional_rnn 机制

① 调用 `static_rnn()` 获得正向输出 (`output_fw`, `output_state_fw`)

② 对输入 `inputs` 逆排序, 调用 `static_rnn()` 获得逆向输出。
是 - `input list` 逆:

③ 将 `output_fw` 和 `output_bw` 拼接 \Rightarrow `flat_output_fw` 和 `flat_output_bw`

`tuple([concat([fw, bw], 1) for fw, bw in zip(flat_output_fw, flat_output_bw)])`
||
`outputs`
↓
拼接

返回 (`outputs`, `output_state_fw`, `output_state_bw`)

MultiRNN(Lstm, GRU)

① state-size

tuple(cell.state_size for cell in self._cells)

$(\underbrace{(c_{num_unit}, h_{num_unit})}_{Lstm}, \underbrace{(h_{num_unit})}_{GRU})$

② zero-state (调用的是 RnnCell 方法, 继承而来)

③ call(self, inputs, state):

cur_inp = inputs

for i, cell in enumerate(self._cells):

if self._state_is_tuple:

cur_state = state[i]

cur_inp, new_state = cell(cur_inp, cur_state)

new_states.append(new_state)

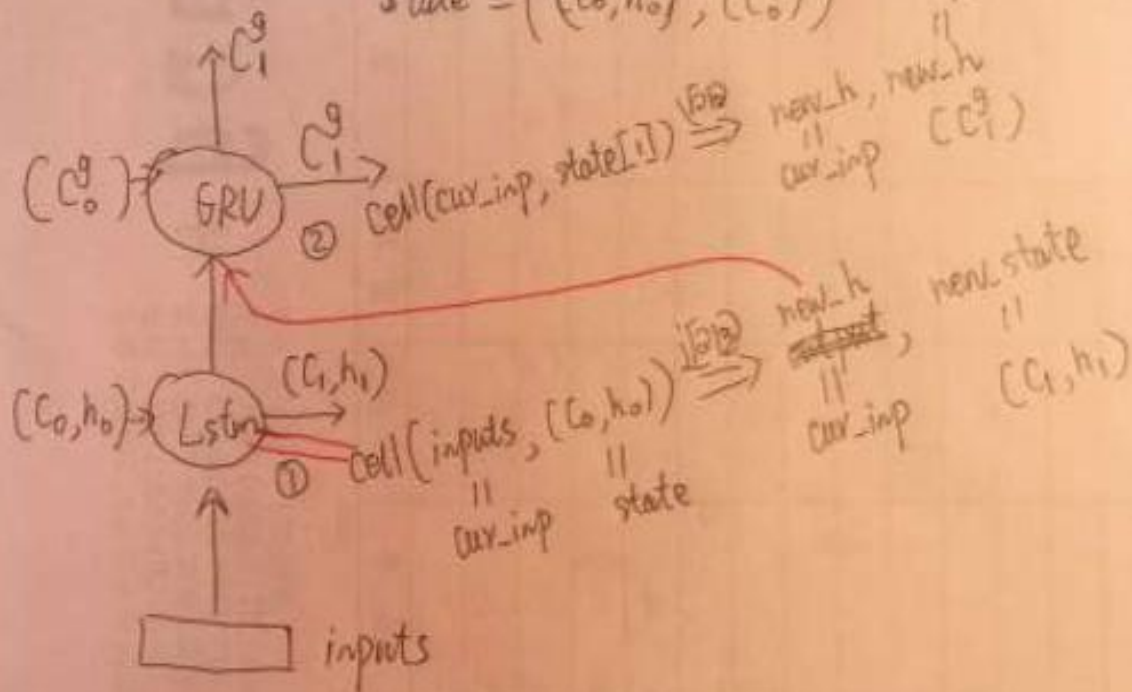
new_states = (tuple(new_states))

return cur_inp, new_states

= cells = (Lstm, GRU)

MultiRNN(Lstm + GRU)

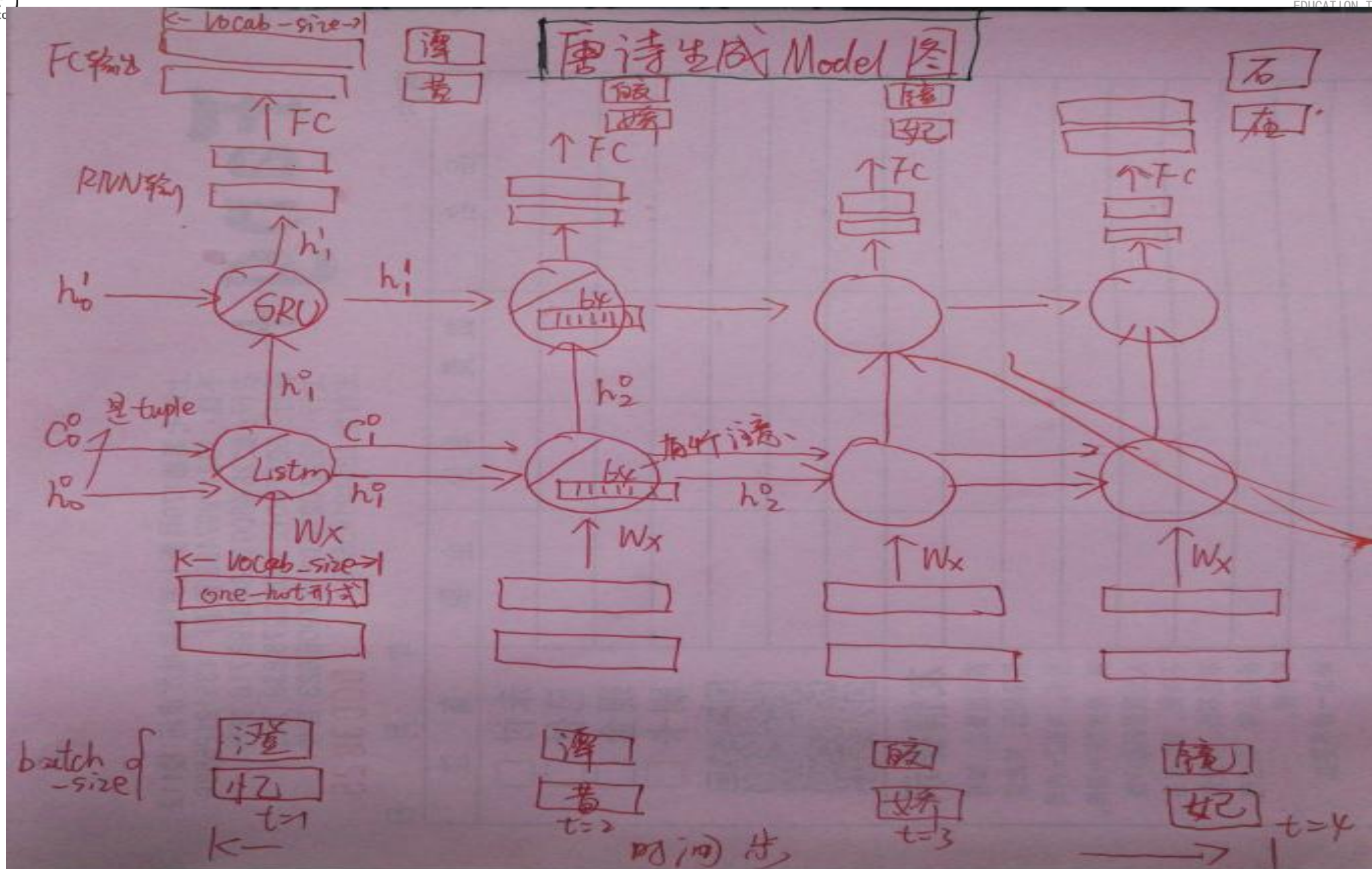
state = $((c_0, h_0), (c_0^g))$



③ states.append(new_state) $\Rightarrow [(c_1, h_1), c_1^g]$

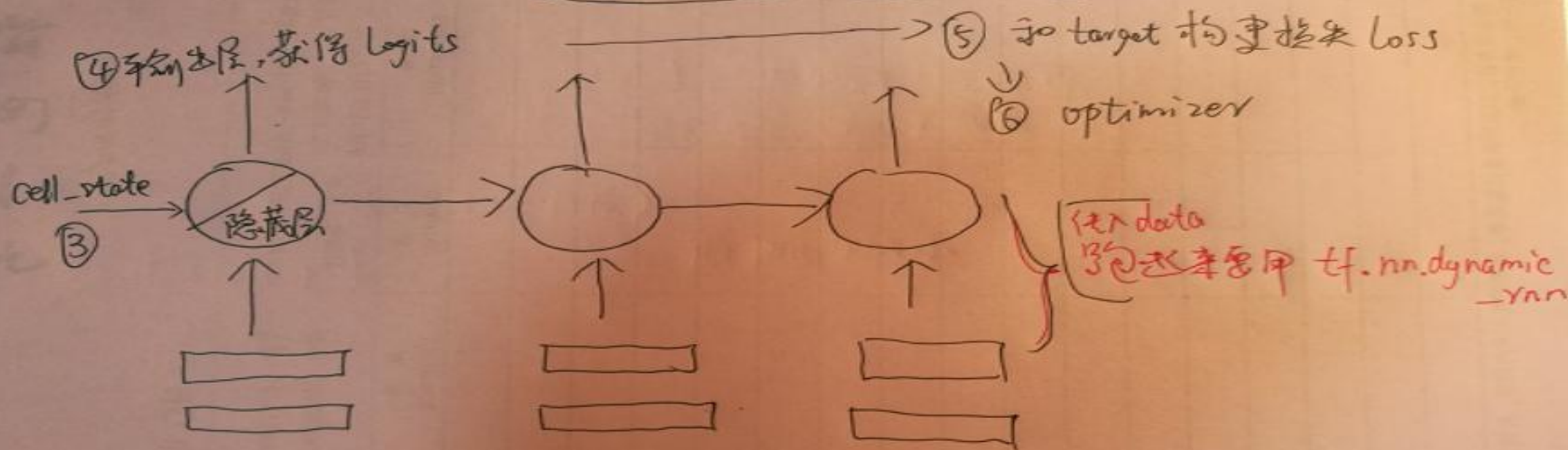
④ (tuple(states)) $\Rightarrow ((c_1, h_1), c_1^g)$

唐诗生成--模型图



01 字符级RNN项目—序列处理

01 RNN 模型结构图



② input 占位符 + target 占位符

① 数据预处理

④ 字符 RNN 序列, batch 获取

[1, 2, 3, 4, 5, 6, ..., 15] drop掉

batch=2 单个序列长度=3

① $\frac{\text{len(arr)}}{\text{batch} \times \text{n-steps}} \Rightarrow \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 8 & 9 & 10 & 11 & 12 & 13 & 14 \end{bmatrix}$ 取 batch x n-steps 丢弃

② $x_0 = \begin{bmatrix} 1 & 2 & 3 \\ 8 & 9 & 10 \end{bmatrix}$ $y_0 = \begin{bmatrix} 2 & 3 & 4 \\ 9 & 10 & 11 \end{bmatrix}$ 在它的基上挪一位

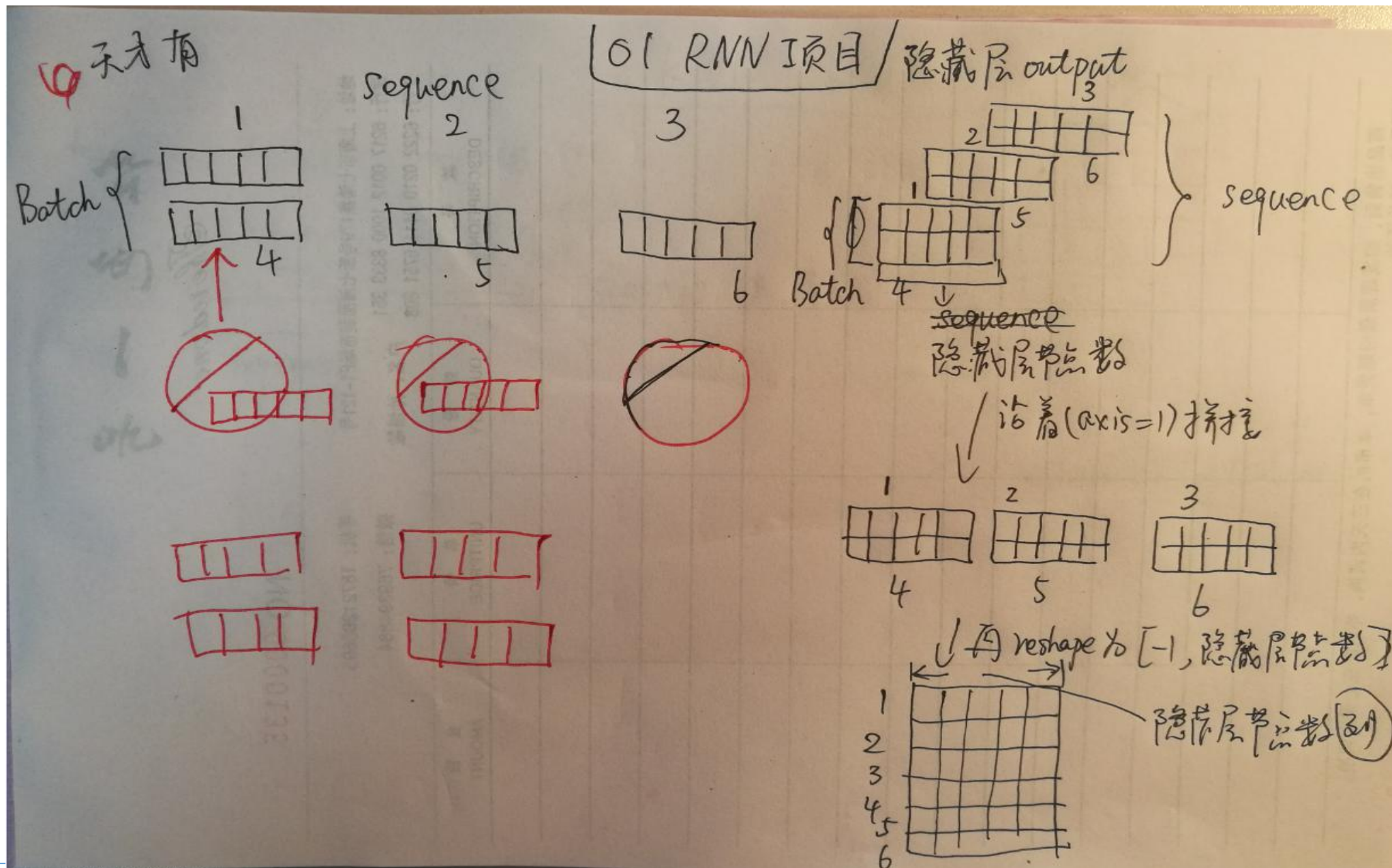
③ 但是最后 19 batch, y 会有问题

$x_{\text{final}} = \begin{bmatrix} 4 & 5 & 6 \\ 11 & 12 & 13 \end{bmatrix}$ $y\text{-temp} = \begin{bmatrix} 5 & 6 \\ 12 & 13 \end{bmatrix}$

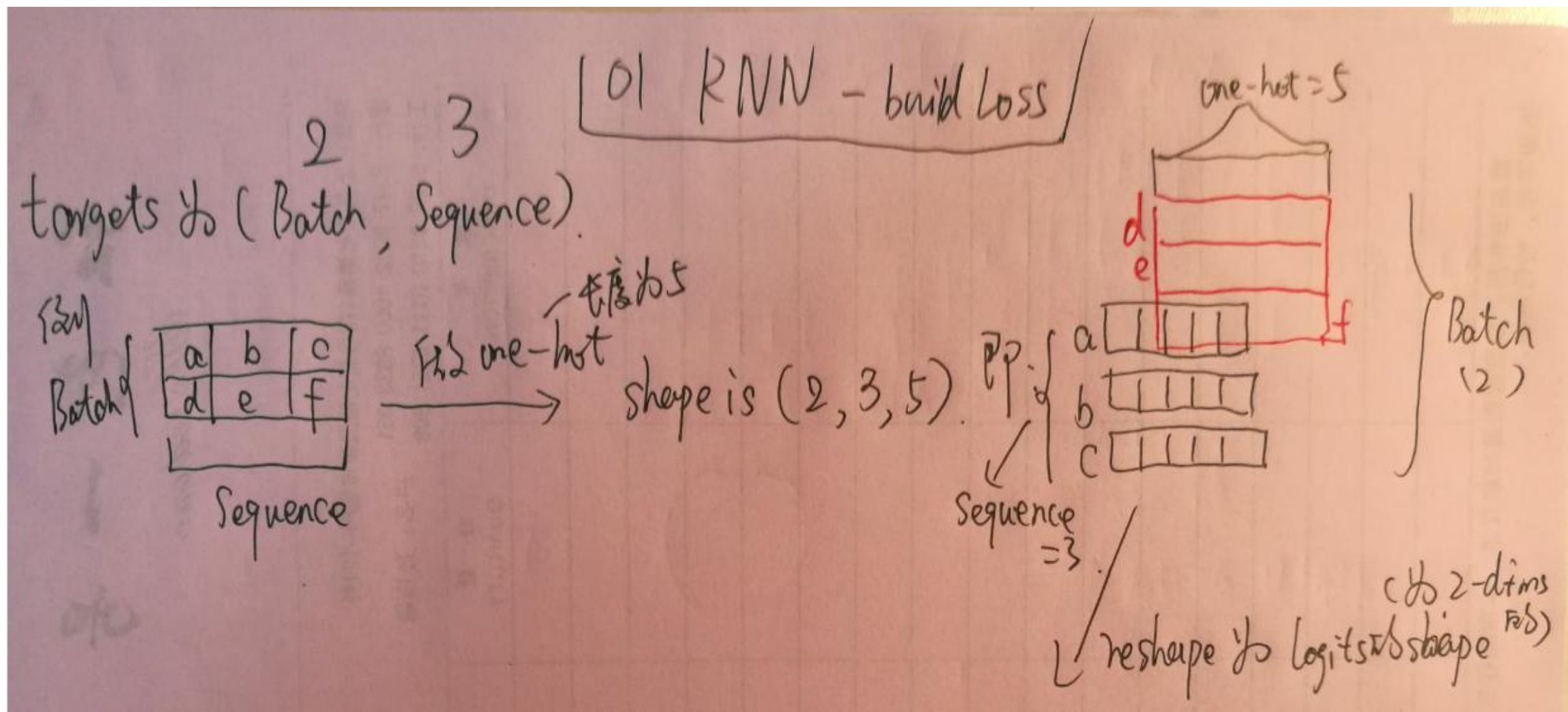
$y = \text{np.zeros}(x.\text{shape})$

$= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ 将 y-temp 4 就搞定, 最后的 19 就为 0 了

01 字符级RNN-hidden output



01 字符级RNN项目--Loss



Word2vec—子采样

② Word2Vec - 子采样

子采样. $P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}$

↓
丢弃 Prob

↑
→ 单词在总数据集中出现的 freq.
即单词 freq 越大, $P(w_i)$ 越大

若 $\text{random}(0, 1) - \text{个数} < (1 - \frac{P(w_i)}{\text{越大, 则 } (1 - P(w_i)) \text{ 越小}})$
为真, 则保留该单词.

62 Word2Vec

(a, b, c, d, e).

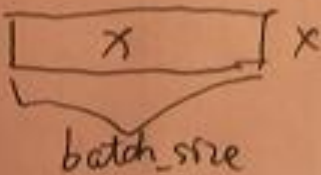
<1> y.extend(b, c, d, e)
x.extend(a, a, a, a)

<2> y (a c d e)
x (b b b b)

input_x shape: [None]
input_y shape: [None, 1]

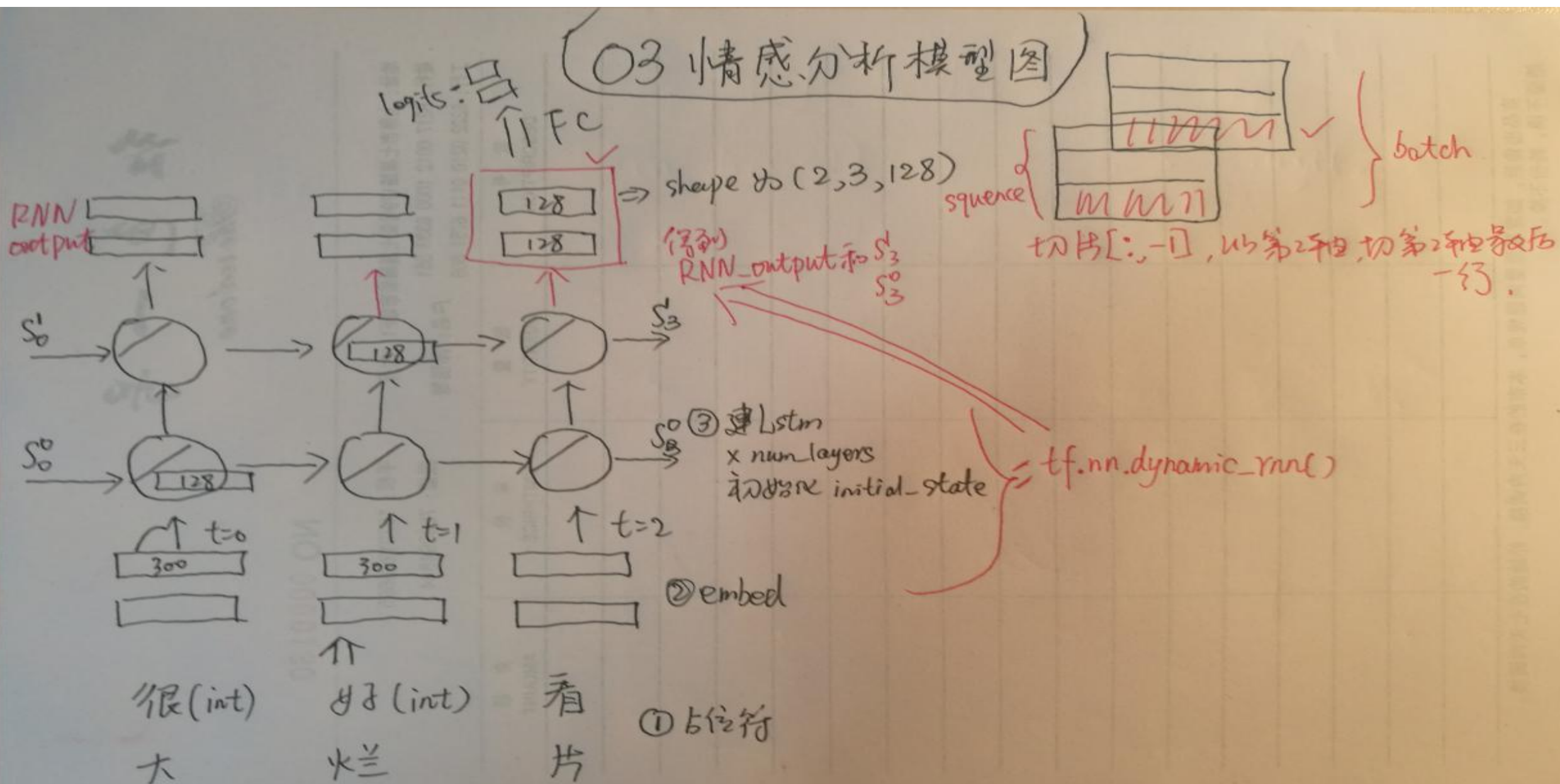
\Rightarrow

y = [b, c, d, e, a c d e]
x = [a a a a, b b b b]

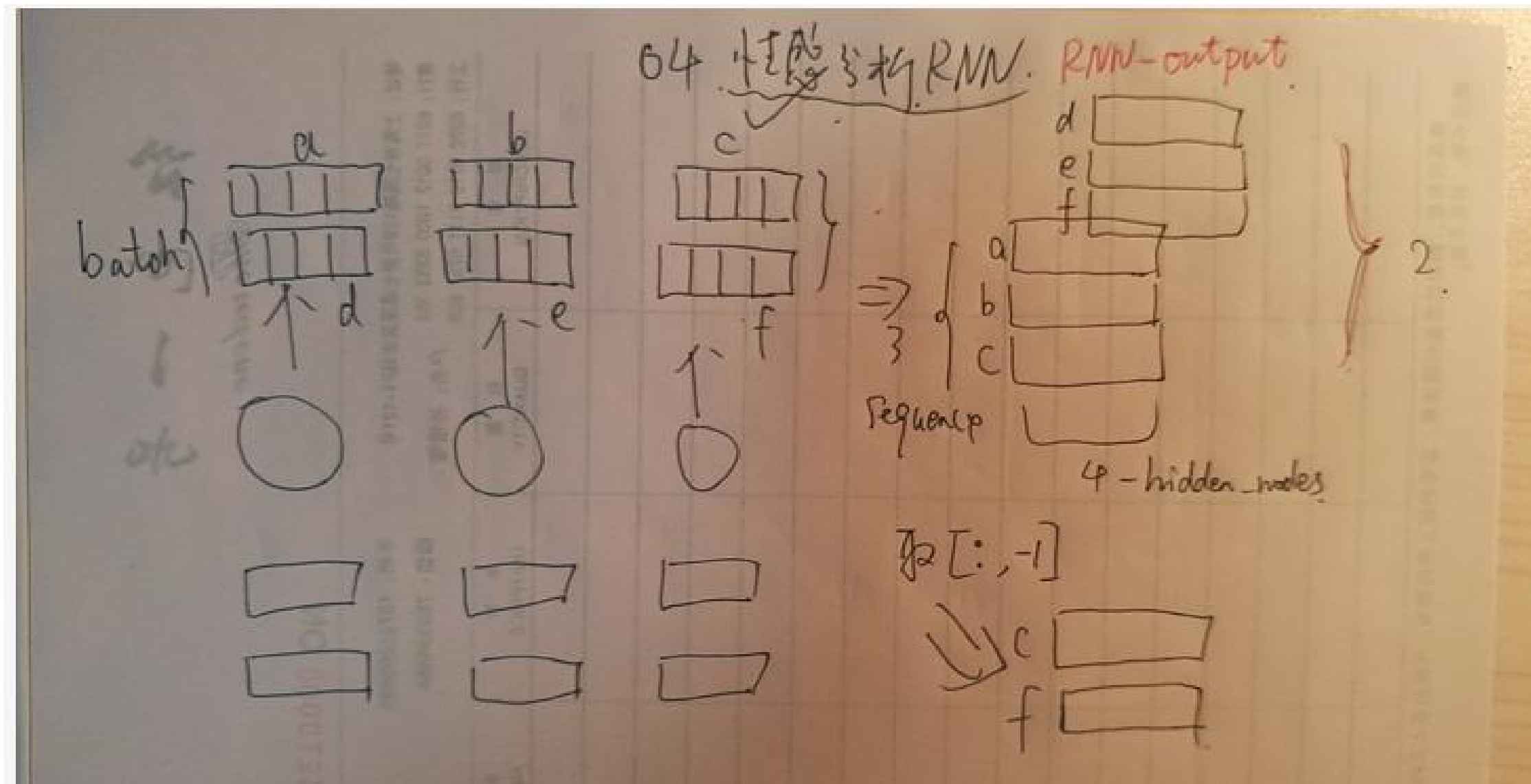


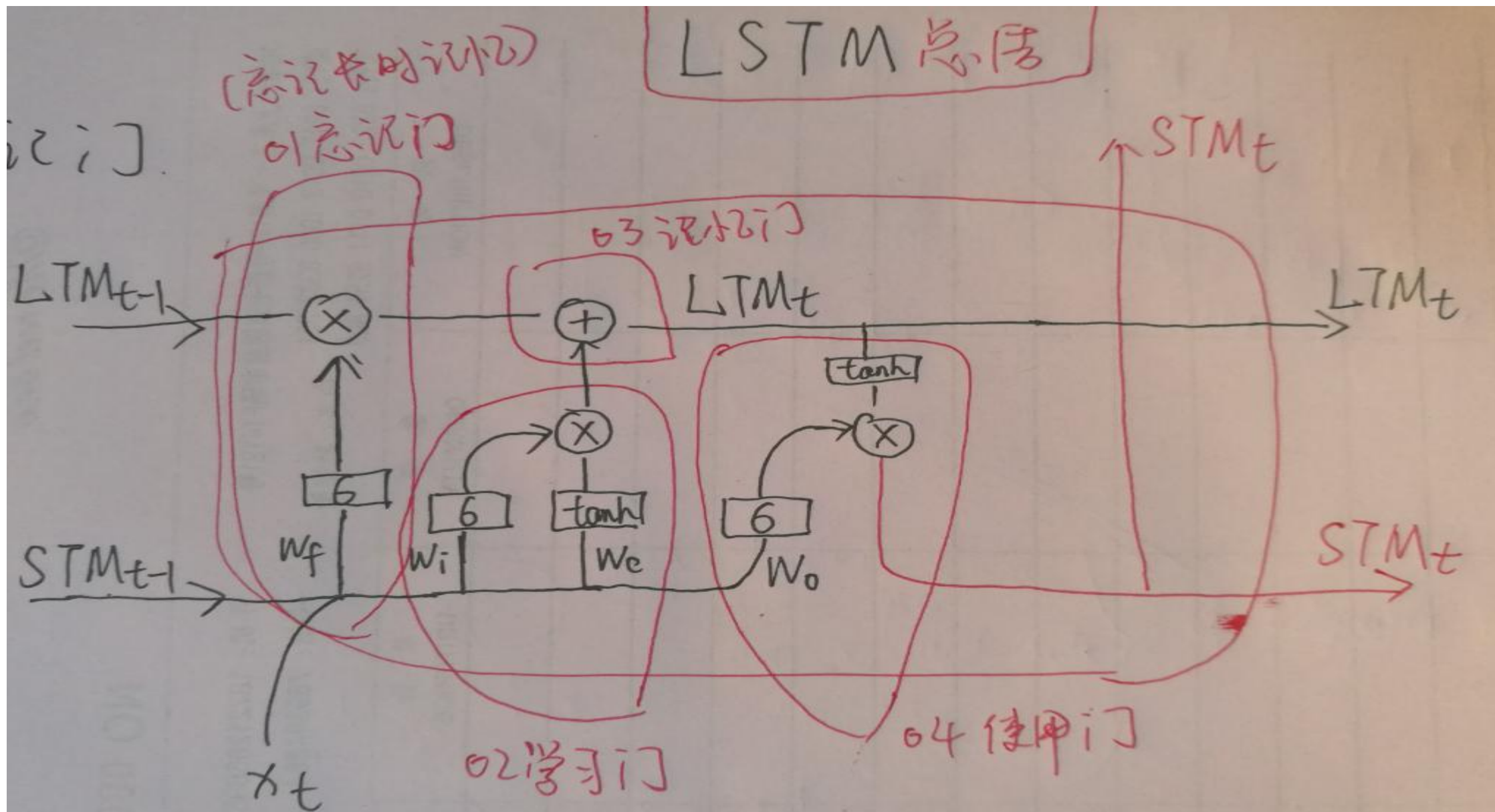
The diagram shows a horizontal rectangle labeled 'x' inside. Below the rectangle, a bracket spans the width and is labeled 'batch_size'. To the right of the rectangle, the letter 'x' indicates the dimension.

04 情感分析RNN-output图



04 情感分析RNN-output图





$$LTM = (128,)$$

$$STM = (128,) \text{ --- 同上}$$

RNN隐藏层节点数

LSTM隐藏层节点数

$$x_t = (50,) \text{ embed 数}$$

01 忘记门

$$6([STM; x_t] \cdot W_f) = 6([178,] \cdot (178, 128)) = 6(128,)$$

f_t 忘记因子

⊗ LTM(128,)

构成忘记门

02 学习门

$$\begin{aligned} & 6([STM; x_t] \cdot W_i + b_i) = 6((178,) \cdot (178, 128)) = 6(128,) \\ & \tanh([STM; x_t] \cdot W_c + b_c) = \tanh(\text{同上}) = \tanh(128,) \end{aligned}$$

⊗ 最终构成学习门

