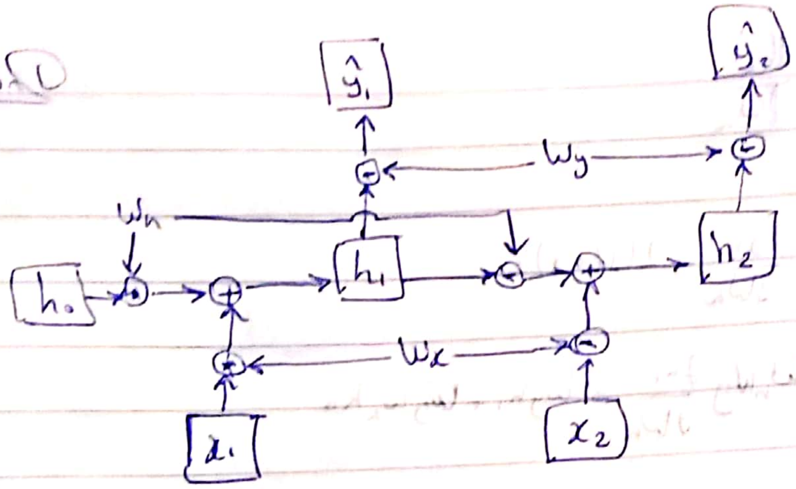
Question①

Question①



1) $\hat{y}_2 = W_y h_2$

$= W_y(W_x x_2 + W_h h_1)$

$= W_y(W_x x_2 + W_h(W_x x_1 + W_h h_0))$

$= 2(0.1 \cdot 10 + 1(0.1 \cdot 10 + 1)) = 6$

2) • $L_t = \sum_i (\hat{y}_i - y_i)^2$

• $\hat{y}_1 = W_y h_1 = W_y(W_x x_1 + W_h h_0) = 2(0.1 \cdot 10 + 1 \cdot 1) = 4$

• $L_t = (4-5)^2 + (6-5)^2 = 2$

3) $\frac{dL_t}{dh_1} = \frac{d}{dh_1}(\hat{y}_1 - y_1)^2 = \frac{d}{dh_1}(W_y h_1 - y_1)^2$

$\frac{dL_t}{dh_1} = 2(W_y h_1 - y_1)(W_y) = 2(2 \cdot h_1 - 5)(2)$

$= 2(4-5)(2) = -4$

4) $\frac{dL_t}{dW_h} = \frac{d}{dW_h}\left((W_y(W_x x_1 + W_h h_0) - y_1)^2 + (W_y(W_x X_2 + W_h(W_x X_1 + W_h h_0)) - y_2)^2\right)$

$= \frac{d}{dW_h}($

4) $\frac{dL_t}{dW_h} = \frac{dL_1}{dW_h} + \frac{dL_2}{dW_h}$

- $\frac{dL_1}{dW_h} = 2(\hat{y}_1 - y_1)\frac{dy_1}{\partial W_h} = 2(4-5)2$

$\frac{dy_2}{dW_h} = W_y h_1 + W_h W_y \frac{\partial h_1}{\partial W_h} = W_y h_1 + W_y W_h h_0$

$= 2(2) + 2(1)(1) = 6$

- $\frac{dL_2}{dW_h} = 2(\hat{y}_2 - y_2)\frac{dy_2}{dW_h} = 2(6-5)\times 6 = 12$

- $\frac{dL_t}{dW_h} = (-4) + 12 = 8$

Question ②:

- $h_t = \tanh(W_{hh}h_{t-1}, W_{xh}x_t)$
- $h_t = \tanh(W_{hh}(\tanh(W_{hh}h_{t-2}, W_{xh}x_t)), W_{xh}x_t)$
- $h_t = \tanh(W_{hh}(\tanh(W_{hh}\tanh(W_{hh}h_{t-2}, W_{xh}x_t))), W_{xh}x_t)$
- As the sequence goes longer & longer, We will be multiplying smaller numbers by each other & this will lead to difficulty in capturing long sequences, as well as leading to vanishing gradient problem

Question ③:

- Because GRU (Gated Recurrent Unit) uses a gate in its architecture which allows the model to learn when to pass the gradients and when to forget it
- This leads to a more robust model able to handle short and long sequences.

Question ④:

- advantage → It leads to less sost for updating parameters as the sequence is truncated

- disadvantage → Not able to capture dependencies longer than updated length

Question 5

a) Because RNN uses shared weights at every time step, and learns from the input sequence as a whole, so it will be able to capture the encryption dependency easier. as the encryption function is the same for all inputs

b) Both are sequence of characters

c) many-to-many, because the input is a sequence of characters as well as the output

d) ex 1) → in : ABCDE
out : DEFGH

ex 2) → In : HELLO
out : KHOOR

ex 3) → in : GOOD
out : JRRG

e) for each batch of data, we zero pad all the samples to the length of the longest sample in the batch.

f)
- Character Tokenization
- Remove Punctuation
- Encode or remove numbers (digits)
- Split long sequences to multiple samples

g) We can use simple encoder decoder architecture with only 1 ~~hidden~~ layer of hidden units as the task is simple

h) We can convert the characters to either one hot encoded vectors or simple integer indices. Then the model will learn to add the cipher shift to generate the deciphered Output