CAP 6619 Deep Learning

2024 Summer

Homework 4 [12 Pts, Due: June 18 2024. Late Penalty: -2/day]

[If two homework submissions are found to be similar to each other, both submissions will receive 0 grade]

[Homework solutions must be submitted through Canvas. No email submission is accepted. If you have multiple files, please include all files as one zip file, and submit zip file online (only zip, pdf, or word files are allowed). You can always update your submissions. Only the latest version will be graded.]

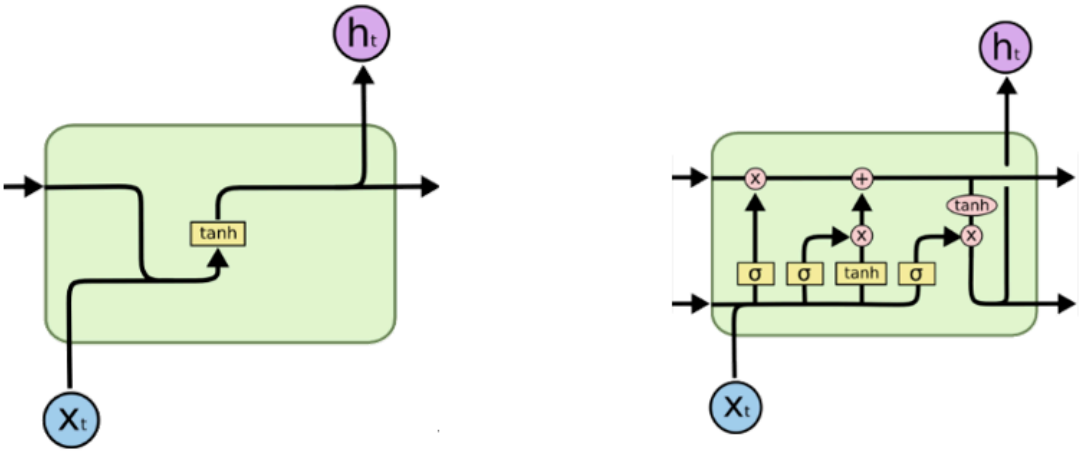
Question 1 [2 pts]: Figure 1 shows the structure of an RNN cell vs. an LSTM cell.

- Summarize major difference between RNN cell vs LSTM cell in terms of their neural architectures [1 pt]

  The architecture of LSTM is more complex than RNN. The major differences are:
  a. LSTM has a gating architecture to control and manage information flow. RNN has no such mechanism.
  b. LSTM uses cells to maintain memories and has the ability to choose what to keep in memory.
  c. LSTM uses different functions to compute hidden states.

- Why LSTM can achieve long-short term memory, whereas RNN cannot [1 pt]

LSTM is capable of long short term memory because of its gating mechanism. The gating mechanism has 3 gates (forget gate, input gate, and output gate). The forget gate removes all irrelevant information, the input gate decides which values to update, while the output gate updates the value. RNNs cannot do so due to vanishing gradient problem.

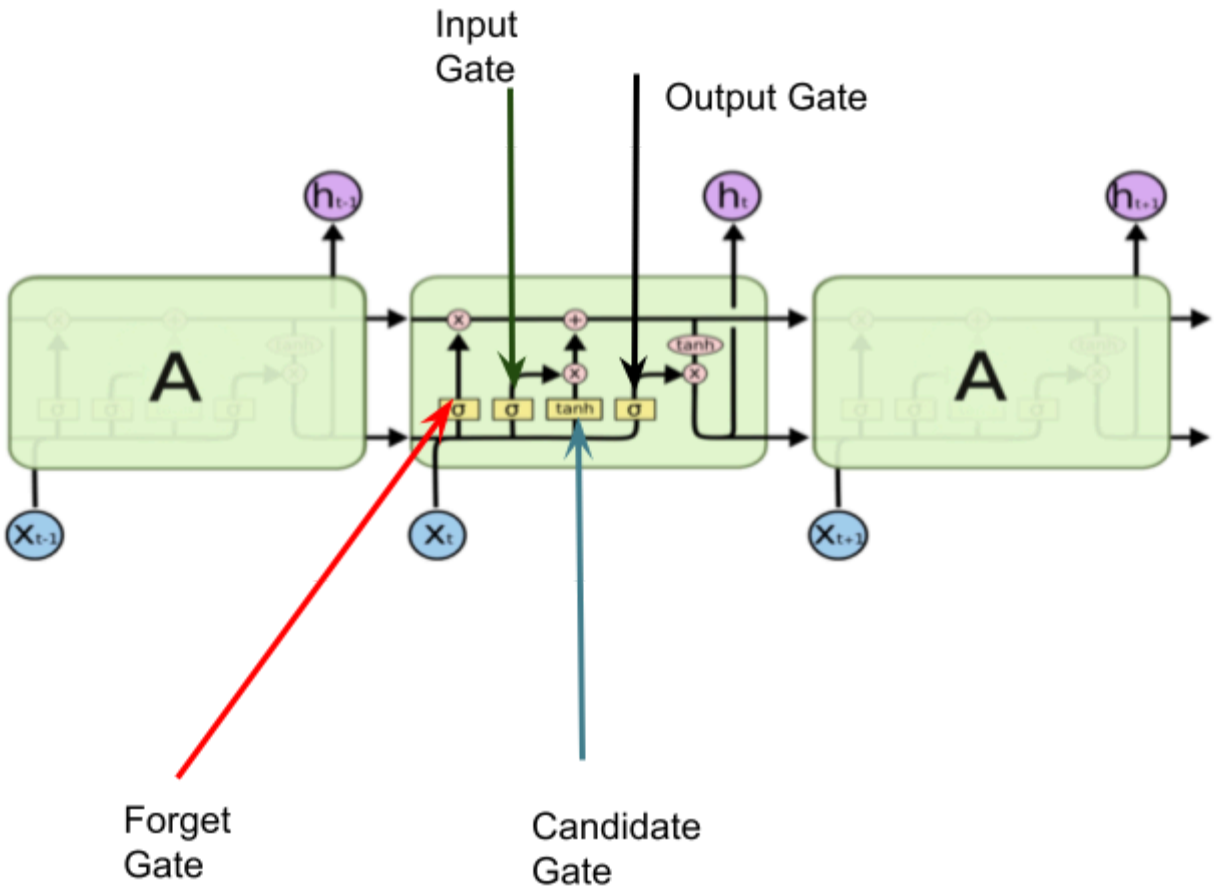


(a) RNN Cell (b) LSTM Cell

Figure 1: RNN vs LSTM Cell

Question 2 [2 pts]: Figure 1 shows the structure of an LSTM cell, and its connection to adjacent cells to form a recurrent neural network.

- Please mark forget gate, input gate, output gate, and candidate layer, respectively [1 pt]



- Explain the main role/functionality of the forget gate, input gate, output gate, and candidate layer, respectively [1 pt]

LSTM consists of a gating mechanism.

1. Forget Gate: For a given cell state, the forget gate decides which information from the previous cell state should be remembered or not. It receives the current input and the previous hidden state as inputs, and for each number in the cell state, it produces a number between 0(remember) and 1(forget).

2. Input Gate: The input gate decides how much of each component should be added to the cell state. It consists of two parts: a sigmoid layer (which outputs numbers between 0 and 1) called the input gate, and a tanh layer that creates a vector of new input values.

3. Output Gate: The output gate updates the value.
4. Candidate Layer: It generates fresh cell state candidate values. These are possible additional values that the input gate will filter and then add to the cell state.

Question 3 [3 pts]: Figure 3 shows an unfolded LSTM network with two consecutive cells. Using h_t and c_t to denote output and cell memory of the cell at time point t. Use f_t, i_t, o_t to denote

forget gate, input gate, output gate of the cell at time point *t*. Use $\tilde{c}_t$ to denote candidate layer output at time point *t*.

1. Use mathematical equations to show the relationship between cell memory at time point *t*, with respect to cell memory and output at time point *t-2*. [1 pt]

The equations for LSTM are:

① Candidate Cell:

$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$

h_{t-1} = output from previous time step.

② Input Gate

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

③ Forget Gate

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

④ ~~Cell mem~~ Output Gate:

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

⑤ Cell Memory Update

$$c_t = f_t \, c_{t-1} + i_t \, \tilde{c}_t$$

⑥ Output, h_t

$$h_t = o_t \cdot \tanh(c_t)$$

Here, c_{t-1} and $\tilde{c}_t$ depend on c_{t-2}, h_{t-2}, and current input x_t. The cell c_t indirectly incopporate information from c_{t-2} through c_{t-1}.

$$c_t = f_t \, c_{t-1} + i_t \cdot \tilde{c}_t$$

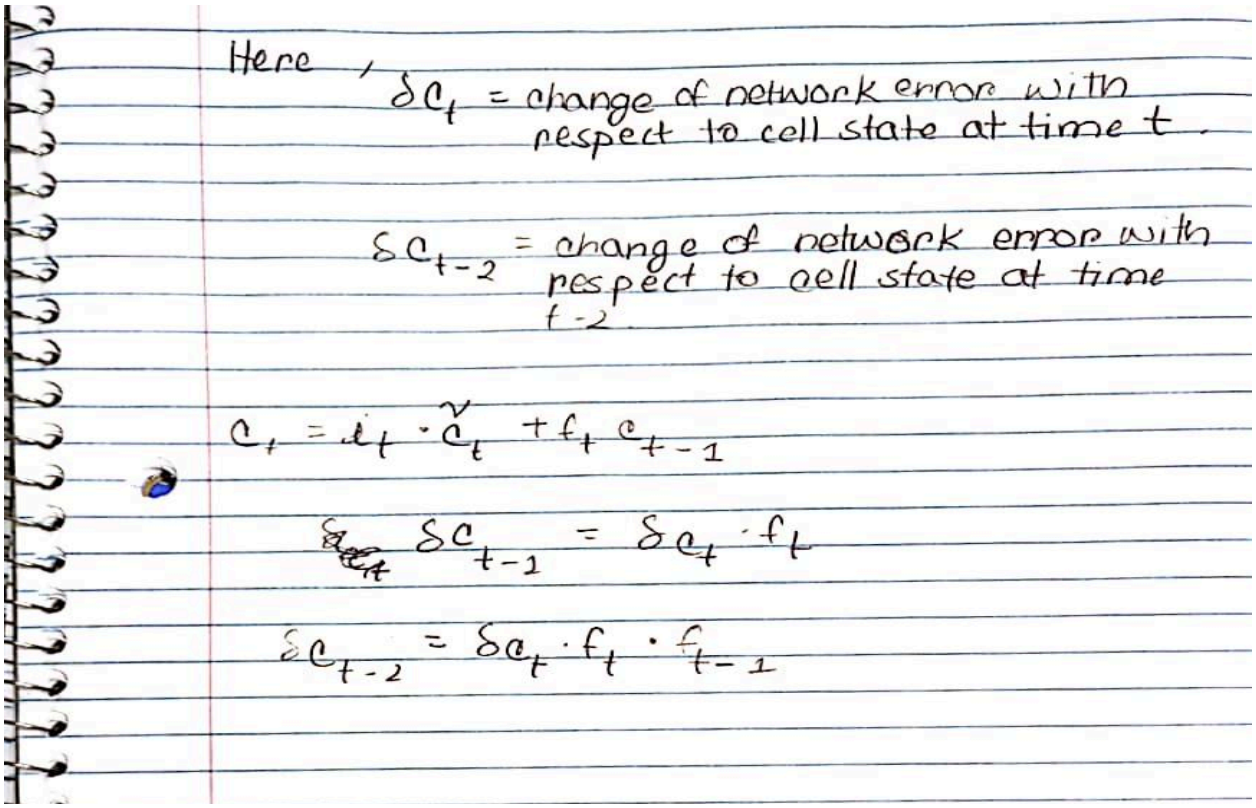
$$c_{t-1} = f_{t-1} \, c_{t-2} + i_{t-1} \, \tilde{c}_{t-1}$$

So,

$$c_t = f_t \left(f_{t-1} \, c_{t-2} + i_{t-1} \, \tilde{c}_{t-1} \right) + i_t \, \tilde{c}_t$$

2. Use δC_t to denote change of network error with respect to cell state at time point *t*, i.e., $\delta C_t = \frac{\partial E}{\partial C_t}$.

   a. Derive relationship between δC_{t-2} and δC_t [1 pt]



   b. Explain why LSTM cell can alleviate weight vanishing or exploding in deep neural network learning. [1 pt]

   Deep network designs and activation functions like as sigmoid are the main sources of the declining gradient problem that occurs during backpropagation.

   The gated structure of LSTM architectures (forget gate, input gate, output gate) allows to selectively keep or discard information over time. As a result, information can pass between its gates and be selectively remembered or forgotten. Furthermore, it employs the sigmoid and tanh activation functions, maintaining an output range of 1 to -1. Because of this, LSTM is especially well-suited for applications that call for simulating long-range dependencies in sequential data.

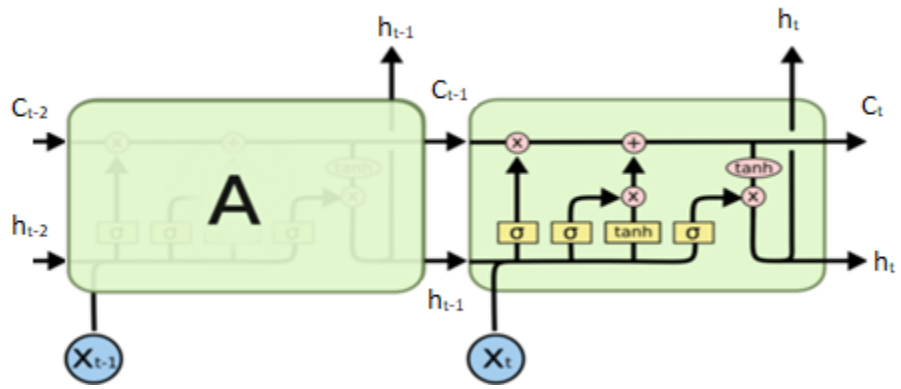


Figure 3: unfolded LSTM cells

Question 4 [2 pts]: The following Keras codes show a deep learning network for text classification (only the network structure part).

1. What is the purpose of the Embedding()? The Embedding() layer output size is 16, what does this mean?, what is the number of weight parameters for the Embedding() layer (show your solutions) [1 pt]

   The embedding layer in neural networks maps categorical variables (particularly words in natural language processing) into a continuous vector space in such a way that relationships between the words can be better understood by the neural network.

   The embedding layer has V * D parameters,
   where,
   V = number of unique words.
   D = size of each embedding vector.

   Here, V = 1000 and D = 16,
   Number of weight parameters = 16000

2. What is the number of weight parameters for the LSTM () layer (Show your solutions) [0.5 pt].

   LSTM consists of 3 weight parameters:

   Weights for input data: 4 * d * n
   Weights from from the previous hidden state to each gate and the candidate cell state: 4 * n * n
   Weights for bias: 4 * n

   Total Weight = 4 (d * n + n * n + n)

where, d = input dimension and n = number of LSTM units

For this problem, d = 16 and n = 32.

So, total weights = 6272

3. What is the total number of weight parameters for the last two dense layers (show your solutions) [0.5 pt].

   The weight parameters = Input size * number of units (weights) + number of units (bias)

   For 1st dense layer, number of weight parameters = 32 * 256 + 256 = 8448
   For 2nd dense layer, number of weight parameters = 256 * 1 + 1 = 257

   Total weight parameters = 8448 + 257 = 8705

```
model = Sequential()
model.add(Embedding(1,000, 16, input_length=200))
model.add(LSTM(32, dropout=0.1, recurrent_dropout=0.1))
model.add(Flatten())
model.add(Dropout(0.1))
model.add(Dense(256, activation='sigmoid'))
model.add(Dense(1, activation='sigmoid'))
```

Question 5 [3 pts]: Figure 4 shows a ResNet block where the input $a^{[l]}$ is a 2x2 matrix. The two layers inside the ResNet block consist of two CNN layers, each has one 3x3 convolution filter. The input images and the two CNN filters are given as follows:

Input image:

-1	-1
1	1

; CNN1 filter:

1	1	1
-1	-1	-1
1	1	1

; CNN2 filter:

1	-1	1
1	-1	1
1	-1	1

Use zero padding with stride 1. Ignore bias, and use ReLu() as the activation functions for all layers.

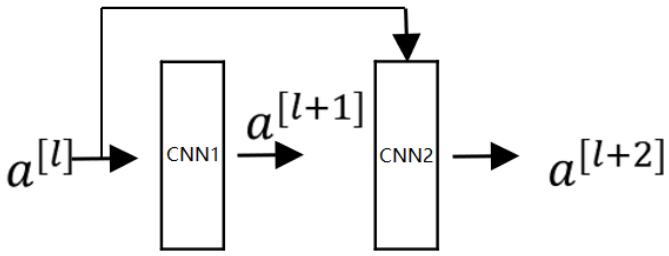
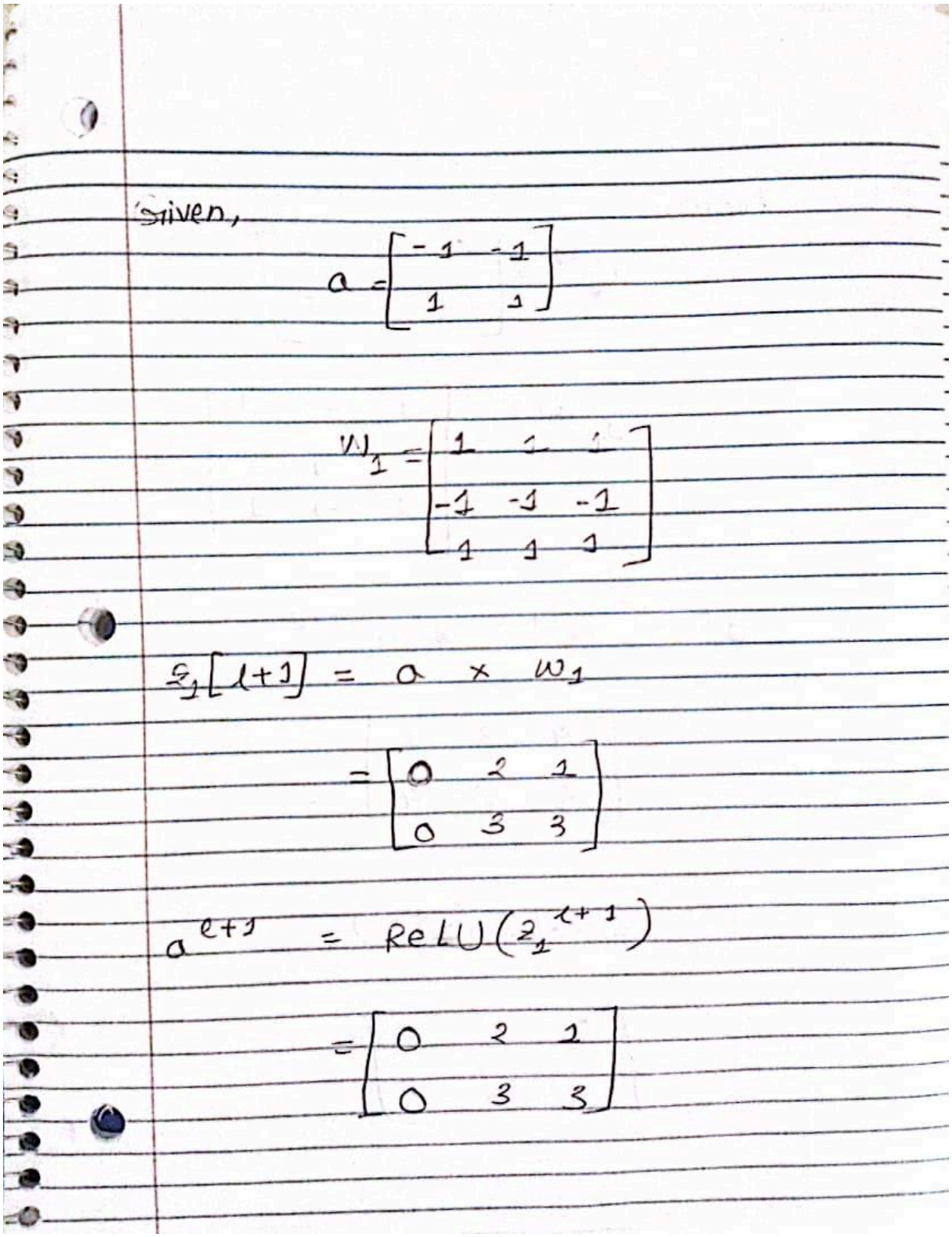


Figure 4

- Calculate output of the first layer of the ResNet block, i.e., $a^{[l+1]}$ (show calculations) [1 pt]



OBJ

- Calculate output of the second layer of the ResNet block, i.e., $a^{[l+2]}$ (show calculations) [2 pts]

Now,

$$a^{l+1} = \begin{bmatrix} 0 & 2 & 1 \\ 0 & 3 & 3 \end{bmatrix}$$

$$w_2^{l+2} = \begin{bmatrix} 1 & -1 & 1 \\ 1 & -1 & 1 \\ 1 & -1 & 1 \end{bmatrix}$$

$$z^{l+2} = a^{l+1} * w_2^{l+2}$$

$$= \begin{bmatrix} -4 & 3 & 1 \\ -3 & 6 & 4 \end{bmatrix}$$

$$a^{l+2} = ReLU(z^{l+2})$$

$$= \begin{bmatrix} 0 & 3 & 1 \\ 0 & 6 & 4 \end{bmatrix}$$