HW5
Deep Learning

Baktash Ansari
99521082

Q1)

a)
Many-to-one RNN:
The second one. Emotional classification because the input is a text and the output is just zero and one. Also, **The third one is true** because first we create the appropriate embedding and input the embedding sequence to the RNN, and output the label. The embedding is sequential data because we don't have any grid data and instead we have sequential.

b)
The answer is a one-way RNN (third one) because the yt depends on the inputs of x1…xt not future inputs.

c) The best answer is $p(y_t|y_1, y_2, \ldots, y_{t-1})$. Because in each timestep the output of a hidden state is calculated based on the output value of all previously hidden layers.

Q2)

Date.

Subject.

a) $\frac{\partial J_t}{\partial o_t} = \frac{\partial\left(-\sum_{i=1}^{2} y_{t,i} \log \hat{y}_{t,i}\right)}{\partial o_t}$

$= -\sum_{i=1}^{2} y_{t,i} \frac{\partial \log \hat{y}_{t,i}}{\partial o_t} \Rightarrow$

$\frac{\partial \log \hat{y}_{t,i}}{\partial o_t} = \frac{\partial \hat{y}_{t,i}}{\partial o_t} / \hat{y}_{t,i}$

$\Rightarrow \hat{y}_{t,i} = \sigma(o_t) \rightsquigarrow \frac{\partial \hat{y}_{t,i}}{\partial(o_t)} = \hat{y}_{t,i}(1-\hat{y}_{t,i})$

$\Rightarrow \frac{\partial \log \hat{y}_{t,i}}{\partial o_t} = \frac{}{1-\hat{y}_{t,i}}$

$\Rightarrow \frac{\partial J_t}{\partial o_t} = -\left(\frac{y_{t,1}}{1-\hat{y}_{t,1}} + \frac{y_{t,2}}{1-\hat{y}_{t,2}}\right)$

$\Rightarrow \frac{\partial J_t}{\partial o_t} = -\left(y_{t,1}(1-\hat{y}_{t,1}) + y_{t,2}(1-\hat{y}_{t,2})\right)$

Date.

Subject.

b) $g_{ot} = \frac{\partial J_t}{\partial o_t}$

$$\frac{\partial J_t}{\partial h_i} = \underbrace{\frac{\partial J_t}{\partial \hat{y}_{t,i}} \alpha \frac{\partial \hat{y}_{t,i}}{\partial o_t}}_{g_{ot}} \times \frac{\partial o_t}{\partial h_t}$$

$$\rightarrow \frac{\partial J_t}{\partial h_i} = \underline{g_{ot} \times W_{yh}}$$

---

c) $g_{ht} = \frac{\partial J_t}{\partial h_i}$

$$\frac{\partial J_t}{\partial W_{hh}} = \underbrace{\frac{\partial J_t}{\partial \hat{y}_{t,i}} \alpha \frac{\partial \hat{y}_{t,i}}{\partial o_t} \alpha \frac{\partial o_t}{\partial h_t}}_{g_{ht}} \alpha \frac{\partial h_t}{\partial z_t} \times \frac{\partial z_t}{\partial W_{hh}}$$

$$= g_{ht} \alpha \frac{\partial h_t}{\partial z_t} \alpha h_{t-1}$$

$$= g_{ht} \alpha \gamma'(z_t) \alpha h_{t-1}$$

d)

$$\frac{\partial J_t}{\partial w_{hh}} = \frac{\partial w_{hh}, t}{}$$

$$\psi'(z_i) = \psi(z_i) \times (1 - \psi(z_i))$$

$$\Rightarrow \frac{\partial \psi(z_i)}{\partial h_{i-1}} = \psi'(z_i) \times w_{hh}$$

$$\Rightarrow \frac{\partial h_{j-1}}{\partial w_{hh}} = h_{i-2}$$

$$\Rightarrow \frac{\partial J}{\partial w_{hh}} = \sum_{i=1}^{r} \frac{\partial w_{hh}, t}{} \psi'(z_i) \alpha \psi(z_i)$$

$$\alpha w_{hh} \alpha h_{i-2}$$

Q3)

Q3) $q = \begin{bmatrix} 3 \\ -1 \\ -1 \end{bmatrix}$

$q * Keys = \begin{bmatrix} 3 \\ -1 \\ -1 \end{bmatrix} * \left\{ \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix}, \begin{bmatrix} 0 \\ -2 \\ -4 \end{bmatrix} \right\}$

$= \{-2, 3, 0, 6\}$

Based on the information of the question, we apply the Argmax function on the similarity of q and keys:

$Argmax(\{-2, 3, 0, 6\}) = [0, 0, 0, 1]$ ↝ the answer is index 3

Now the corresponding value of the index 3 is values[3]

which is equal to $\begin{bmatrix} 6 \\ 1 \\ 2 \end{bmatrix}$ ⟶ The ouput of the attention layer for the related query = [6, 1, 2]

b)

Advantages of Argmax:

- **Simplicity**: Argmax is a simple operation that finds the argument that gives the maximum value from a target function. It's straightforward to understand and implement.
- **Computational Cost**: The softmax function involves exponentiation and normalization, making it computationally more expensive than argmax, which simply selects the maximum value.
- **Focus on Maximum:** Argmax focuses on the maximum value, which can be useful when we are only interested in the most significant feature.

Disadvantages of Argmax:

- **Loss of Information**: Argmax selects only the maximum value and discards all others. This could lead to a loss of information as only the most relevant feature would be considered and all others would be discarded.
- **Non-differentiability:** Argmax is non-differentiable, which means it does not provide a gradient that can be used for backpropagation during the training process. This could make it difficult for the model to learn and adjust the parameters effectively during training.
- **No Soft Weights:** Unlike softmax, argmax does not provide soft weights that can change during each runtime. This could limit the flexibility of the model in focusing on different parts of the input based on their relevance.

The use of argmax in the attention mechanism presents a significant challenge for improving keys and queries during training. This is primarily due to the non-differentiability of the argmax function, which does not provide a gradient that can be used for backpropagation during the training process. This could make it difficult for the model to learn and adjust the parameters effectively during training.

Q4)

Codes are available at the Q4 notebook in the zip file.