FIT3181 S2 2023 - ASSIGNMENT 2

Student Information

Surname: **Leong**
Firstname: **Benjamin Tjen Ho**
Student ID: **32809824**
Email: **bleo0009@student.monash.edu**
Your tutorial time: **Friday, 8am - 10am**

In [ ]:
```
# Necessary imports for code
from math import e, log
import tensorflow as tf
```

Section 1: Knowledge Questions

Question 1

The general architecture of a RNN is the encoder-decoder architecture. The encoder takes in an input, x, and summarises it into a latent code, z. z is then inputted into the decoder that generates an output related to x. When combining CNN and RNN, the CNN architecture allows the RNN to encode spatial data and decode it to generate relevant outputs.

Some example outputs would be **Image Captioning** as well as **Video Analysis**. To elaborate on the first example, the CNN will be able to summarise the image into some context vector (latent code) and pass the value of z into the RNN, which then decodes that z value into a sequence of text which will act as the image's caption.

The second example will take in a sequence of images (a video) to produce a z value which is then decoded to produce some sequence of text which could summarise or provide some analysis to the video.

[1 mark]

Question 2

In **RNN**s, we are only able to process the sequential data per unit input in the sequence (e.g., per word in a sentence). This is because each subsequent input is dependent on its previous input's context. This means that if our input is very large, then it will take a very long time to fully process the input.

However, in **Transformer**s, it removes this dependencies by using positional encoding to consider the position of every input when processing the input (encoding). This means that all inputs can be processed at once making use of parallel computing. With that, **Transformer**s can also perform Multihead-Attention (MHA), i.e., Attention for all inputs

Besides that, both **RNN**s and **Transformer**s use Attention with the difference that **Transformer**s use 3 types of Attentions while **RNN**s only have 1. **Transformer**s apply MHA in 3 ways, Self-Attention, Encoder-Decoder Attention and Masked Self-Attention. Self-Attention allows the encoder/decoder in **Transformer**s to refer to the other inputs which then allows long-term dependencies to be captured.

[2 marks]

Question 3

$$x_0 = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix} \quad y_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad U = \begin{bmatrix} 1 & 2 & 3 \\ -1 & 0 & 1 \\ 2 & -1 & 0 \end{bmatrix} \quad b = \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix}$$

$$x_1 = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix} \quad y_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \quad W = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 1 & 0 \\ -1 & 2 & 1 \end{bmatrix} \quad c = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

$$x_2 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \quad y_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad V = \begin{bmatrix} 2 & -1 & 0 \\ 1 & 1 & -1 \\ -1 & 2 & 1 \end{bmatrix}$$

$$\bar{h}_0 = Ux_0 + b,\ h_0 = tanh(\bar{h}_0),\ \hat{y}_0 = softmax(Vh_0 + c)$$

$$\text{For t} > 0,\ \bar{h}_t = Wh_{t-1} + Ux_t + b,\ h_t = tanh(\bar{h}_t),\ \hat{y}_t = softmax(Vh_t + c)$$

[12 marks]

In [ ]:
```
# Inputs and Ground-truth Labels (as one-hot vector)
x_0, x_1, x_2 = (
    tf.constant([[1], [-1], [0]], dtype="float64"),
    tf.constant([[-1], [1], [0]], dtype="float64"),
    tf.constant([[1], [1], [0]], dtype="float64"),
)
y_0, y_1, y_2 = (
    tf.constant([[0], [0], [1]], dtype="float64"),
    tf.constant([[0], [1], [0]], dtype="float64"),
    tf.constant([[0], [0], [1]], dtype="float64"),
)

### Parameters!
# Weights
U = tf.constant([[1, 2, 3], [-1, 0, 1], [2, -1, 0]], dtype="float64")
W = tf.constant([[1, 0, -1], [2, 1, 0], [-1, 2, 1]], dtype="float64")
V = tf.constant([[2, -1, 0], [1, 1, -1], [-1, 2, 1]], dtype="float64")

# Biases
b, c = tf.constant([[0], [1], [-1]], dtype="float64"), tf.constant(
    [[1], [1], [0]], dtype="float64"
)

### Functions
# Tanh Activation Function
tanh = lambda x: (e**x - e ** (-x)) / (e**x + e ** (-x))
tanh_mat = lambda mat: tf.constant(
    [[tanh(num).numpy() for num in row] for row in mat], dtype="float64"
)

# Softmax Activation Function
softmax = lambda pred, vec: (e ** pred.numpy()) / sum(
    [e ** num.numpy() for num in tf.transpose(vec)[0]]
)
softmax_mat = lambda vec: tf.constant(
```

```
    [[softmax(num, vec) for num in row] for row in vec], dtype="float64"
)

# Cross Entropy Loss Function (CE loss)
ce_loss = lambda y, y_hat: -1 * sum([y[i][0] * log(y_hat[i][0]) for i in range(len(y))])
```

Question 3.1

$$
\begin{aligned}
\bar{h}_0 &= Ux_0 + b \\
&= \begin{bmatrix} 1 & 2 & 3 \\ -1 & 0 & 1 \\ 2 & -1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix} \\
&= \begin{bmatrix} -1 \\ -1 \\ 3 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix} \\
&= \begin{bmatrix} -1 \\ 0 \\ 2 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
h_0 &= tanh(\bar{h}_0) \\
&= tanh\left(\begin{bmatrix} -1 \\ 0 \\ 2 \end{bmatrix}\right) \\
&= \begin{bmatrix} -0.7615942 \\ 0.0000000 \\ 0.9640275 \end{bmatrix} \approx \begin{bmatrix} -0.7616 \\ 0.0000 \\ 0.9640 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\bar{h}_1 &= Wh_0 + Ux_1 + b \\
&= \begin{bmatrix} 1 & 0 & -1 \\ 2 & 1 & 0 \\ -1 & 2 & 1 \end{bmatrix} \begin{bmatrix} -0.7615942 \\ 0.0000000 \\ 0.9640275 \end{bmatrix} + \begin{bmatrix} 1 & 2 & 3 \\ -1 & 0 & 1 \\ 2 & -1 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix} \\
&= \begin{bmatrix} -1.7256217 \\ -1.5231884 \\ 1.7256217 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ -3 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix} \\
&= \begin{bmatrix} -0.7256217 \\ 0.47681165 \\ -2.2743783 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
h_1 &= tanh(\bar{h}_1) \\
&= tanh\left(\begin{bmatrix} -0.7256217 \\ 0.47681165 \\ -2.2743783 \end{bmatrix}\right) \\
&= \begin{bmatrix} -0.6203794 \\ 0.44368652 \\ -0.9790609 \end{bmatrix} \approx \begin{bmatrix} -0.6204 \\ 0.4437 \\ -0.9791 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\bar{h}_2 &= Wh_1 + Ux_2 + b \\
&= \begin{bmatrix} 1 & 0 & -1 \\ 2 & 1 & 0 \\ -1 & 2 & 1 \end{bmatrix} \begin{bmatrix} -0.6203794 \\ 0.44368652 \\ -0.9790609 \end{bmatrix} + \begin{bmatrix} 1 & 2 & 3 \\ -1 & 0 & 1 \\ 2 & -1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix} \\
&= \begin{bmatrix} 0.3586815 \\ -0.7970723 \\ 0.52869153 \end{bmatrix} + \begin{bmatrix} 3 \\ -1 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix} \\
&= \begin{bmatrix} 3.3586814 \\ -0.7970723 \\ 0.52869153 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
h_2 &= tanh(\bar{h}_2) \\
&= tanh\left(\begin{bmatrix} 0.3586815 \\ -0.7970723 \\ 0.52869153 \end{bmatrix}\right) \\
&= \begin{bmatrix} 0.9975835 \\ -0.66239685 \\ 0.48438028 \end{bmatrix} \approx \begin{bmatrix} 0.9976 \\ -0.6624 \\ 0.4844 \end{bmatrix}
\end{aligned}
$$

In [ ]:

```
# Computing h_0
pre_h_bar_0 = tf.matmul(U, x_0)
h_bar_0 = tf.add(pre_h_bar_0, b)
h_0 = tanh_mat(h_bar_0)
print("====== h_0 values ======")
print(f"UX_0 = {pre_h_bar_0}")
print(f"h_bar_0 = UX_0 + b = {h_bar_0}")
print(f"h_0 = {h_0}")
print("========================\n")

# Computing h_1
pre1_h_bar_1, pre2_h_bar_1 = tf.matmul(W, h_0), tf.matmul(U, x_1)
h_bar_1 = tf.add(tf.add(pre1_h_bar_1, pre2_h_bar_1), b)
h_1 = tanh_mat(h_bar_1)
print("====== h_1 values ======")
print(f"wh_0 = {pre1_h_bar_1}")
print(f"Ux_1 = {pre2_h_bar_1}")
print(f"h_bar_1 = Wh_0 + Ux_1 + b = {h_bar_1}")
print(f"h_1 = {h_1}")
print("========================\n")

# Computing h_2
pre1_h_bar_2, pre2_h_bar_2 = tf.matmul(W, h_1), tf.matmul(U, x_2)
h_bar_2 = tf.add(tf.add(pre1_h_bar_2, pre2_h_bar_2), b)
h_2 = tanh_mat(h_bar_2)
print("====== h_2 values ======")
print(f"Wh_1 = {pre1_h_bar_2}")
print(f"Ux_2 = {pre2_h_bar_2}")
print(f"h_bar_2 = Wh_1 + Ux_2 + b = {h_bar_2}")
print(f"h_2 = {h_2}")
print("========================\n")
```

```
====== h_0 values ======
UX_0 = [[-1.]
 [-1.]
 [ 3.]]
h_bar_0 = UX_0 + b = [[-1.]
 [ 0.]
 [ 2.]]
h_0 = [[-0.76159416]
 [ 0.        ]
 [ 0.96402758]]
========================

====== h_1 values ======
wh_0 = [[-1.72562174]
 [-1.52318831]
 [ 1.72562174]]
Ux_1 = [[ 1.]
 [ 1.]
 [-3.]]
h_bar_1 = Wh_0 + Ux_1 + b = [[-0.72562174]
 [ 0.47681169]
 [-2.27437826]]
h_1 = [[-0.62037946]
 [ 0.44368657]
 [-0.97906084]]
========================

====== h_2 values ======
Wh_1 = [[ 0.35868137]
 [-0.79707236]
 [ 0.52869176]]
Ux_2 = [[ 3.]
 [-1.]
 [ 1.]]
h_bar_2 = Wh_1 + Ux_2 + b = [[ 3.35868137]
 [-0.79707236]
 [ 0.52869176]]
h_2 = [[ 0.99758347]
 [-0.66239687]
 [ 0.48438043]]
========================
```

Question 3.2

$$
\begin{aligned}
\hat{y}_0 &= softmax(Vh_0 + c) \\
&= softmax\left(\begin{bmatrix} 2 & -1 & 0 \\ 1 & 1 & -1 \\ -1 & 2 & 1 \end{bmatrix}\begin{bmatrix} -0.7615942 \\ 0.0000000 \\ 0.9640275 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}\right) \\
&= softmax\left(\begin{bmatrix} -1.5231884 \\ -1.7256217 \\ 1.7256217 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}\right) \\
&= softmax\left(\begin{bmatrix} -0.5231884 \\ -0.7256217 \\ 1.7256217 \end{bmatrix}\right) \\
&= \begin{bmatrix} 0.08854891 \\ 0.07232152 \\ 0.83912957 \end{bmatrix} \approx \begin{bmatrix} 0.0885 \\ 0.0723 \\ 0.8391 \end{bmatrix}
\end{aligned}
$$

$$\hat{y}_1 = softmax(Vh_1 + c)$$

$$= softmax\left(\begin{bmatrix} 2 & -1 & 0 \\ 1 & 1 & -1 \\ -1 & 2 & 1 \end{bmatrix}\begin{bmatrix} -0.6203794 \\ 0.44368652 \\ -0.9790609 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}\right)$$

$$= softmax\left(\begin{bmatrix} -1.6844453 \\ 0.80236804 \\ 0.52869153 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}\right)$$

$$= softmax\left(\begin{bmatrix} -0.6844453 \\ 1.80236804 \\ 0.52869153 \end{bmatrix}\right)$$

$$= \begin{bmatrix} 0.06102427 \\ 0.73368883 \\ 0.2052869 \end{bmatrix} \approx \begin{bmatrix} 0.0610 \\ 0.7337 \\ 0.2053 \end{bmatrix}$$

$$\hat{y}_2 = softmax(Vh_2 + c)$$

$$= softmax\left(\begin{bmatrix} 2 & -1 & 0 \\ 1 & 1 & -1 \\ -1 & 2 & 1 \end{bmatrix}\begin{bmatrix} 0.9975835 \\ -0.66239685 \\ 0.48438028 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}\right)$$

$$= softmax\left(\begin{bmatrix} 2.657564 \\ -0.14919361 \\ -1.837997 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}\right)$$

$$= softmax\left(\begin{bmatrix} 3.657564 \\ 0.85080636 \\ -1.837997 \end{bmatrix}\right)$$

$$= \begin{bmatrix} 0.93940334 \\ 0.05674045 \\ 0.00385621 \end{bmatrix} \approx \begin{bmatrix} 0.9394 \\ 0.0567 \\ 0.0039 \end{bmatrix}$$

In [ ]:

```
# Computing y_hat_0
pre1_y_hat_0 = tf.matmul(V, h_0)
pre2_y_hat_0 = tf.add(pre1_y_hat_0, c)
y_hat_0 = softmax_mat(pre2_y_hat_0)
print("====== y_hat_0 values ======")
print(f"Vh_0 = {pre1_y_hat_0}")
print(f"Vh_0 + c = {pre2_y_hat_0}")
print(f"y_hat_0 = softmax(Vh_0 + c) = {y_hat_0}")
print("============================\n")

# Computing y_hat_1
pre1_y_hat_1 = tf.matmul(V, h_1)
pre2_y_hat_1 = tf.add(pre1_y_hat_1, c)
y_hat_1 = softmax_mat(pre2_y_hat_1)
print("====== y_hat_1 values ======")
print(f"Vh_1 = {pre1_y_hat_1}")
print(f"Vh_1 + c = {pre2_y_hat_1}")
print(f"y_hat_0 = softmax(Vh_0 + c) = {y_hat_1}")
print("============================\n")

# Computing y_hat_2
pre1_y_hat_2 = tf.matmul(V, h_2)
pre2_y_hat_2 = tf.add(pre1_y_hat_2, c)
y_hat_2 = softmax_mat(pre2_y_hat_2)
print("====== y_hat_2 values ======")
print(f"Vh_2 = {pre1_y_hat_2}")
print(f"Vh_2 + c = {pre2_y_hat_2}")
print(f"y_hat_0 = softmax(Vh_0 + c) = {y_hat_2}")
print("============================\n")
```

```
====== y_hat_0 values ======
Vh_0 = [[-1.52318831]
 [-1.72562174]
 [ 1.72562174]]
Vh_0 + c = [[-0.52318831]
 [-0.72562174]
 [ 1.72562174]]
y_hat_0 = softmax(Vh_0 + c) = [[0.08854891]
 [0.07232151]
 [0.83912957]]
============================

====== y_hat_1 values ======
Vh_1 = [[-1.6844455 ]
 [ 0.80236794]
 [ 0.52869176]]
Vh_1 + c = [[-0.6844455 ]
 [ 1.80236794]
 [ 0.52869176]]
y_hat_0 = softmax(Vh_0 + c) = [[0.06102426]
 [0.73368879]
 [0.20528695]]
============================

====== y_hat_2 values ======
Vh_2 = [[ 2.65756382]
 [-0.14919383]
 [-1.83799679]]
Vh_2 + c = [[ 3.65756382]
 [ 0.85080617]
 [-1.83799679]]
y_hat_0 = softmax(Vh_0 + c) = [[0.93940335]
 [0.05674045]
 [0.00385621]]
============================
```

Question 3.3

$$l_t = CE(y_t, \hat{y}_t) \text{ for } t \in \{0, 1, 2\} \quad L = \frac{1}{3}(l_0 + l_1 + l_2)$$

$$CE(p, q) = -\sum_{i=0}^{M} p_i \log q_i$$

$$\begin{aligned}
l_0 &= CE(y_0, \hat{y}_0) \\
&= CE\left(\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0.08854891 \\ 0.07232152 \\ 0.83912957 \end{bmatrix}\right) \\
&= -\sum_{i=0}^{M} y_{0i} \log \hat{y}_{0i} \\
&= -\log 0.83912957 \\
&= 0.17539015412330627 \approx 0.1754
\end{aligned}$$

$$
\begin{aligned}
l_1 &= CE(y_1, \hat{y}_1) \\
&= CE\left(\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0.06102427 \\ 0.73368883 \\ 0.2052869 \end{bmatrix}\right) \\
&= -\sum_{i=0}^{M} y_{1i} \log \hat{y}_{1i} \\
&= -\log 0.73368883 \\
&= 0.30967026948928833 \approx 0.3097
\end{aligned}
$$

$$
\begin{aligned}
l_2 &= CE(y_2, \hat{y}_2) \\
&= CE\left(\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0.93940334 \\ 0.05674045 \\ 0.00385621 \end{bmatrix}\right) \\
&= -\sum_{i=0}^{M} y_{2i} \log \hat{y}_{2i} \\
&= -\log 0.00385621 \\
&= 5.558071136474609 \approx 5.5581
\end{aligned}
$$

$$
\begin{aligned}
L &= \frac{1}{3}(l_0 + l_1 + l_2) \\
&= \frac{1}{3}(0.17539015412330627 + 0.30967026948928833 + 5.558071136474609) \\
&= \frac{1}{3}(6.043131351470947) \\
&= 2.0143771171569824 \\
&\approx 2.0144
\end{aligned}
$$

In [ ]:

```python
# Computing L_1
l_0 = ce_loss(y_0, y_hat_0)
print("====== l_0 value ======")
print(f"l_0 = CE(y_0, y_hat_0) = {l_0}")
print("=======================\n")

# Computing L_2
l_1 = ce_loss(y_1, y_hat_1)
print("====== l_1 value ======")
print(f"l_1 = CE(y_0, y_hat_1) = {l_1}")
print("=======================\n")

# Computing L_3
l_2 = ce_loss(y_2, y_hat_2)
print("====== l_2 value ======")
print(f"l_2 = CE(y_2, y_hat_2) = {l_2}")
print("=======================\n")

# Computing L
pre_L = (l_0 + l_1 + l_2)
L = (1 / 3) * pre_L
print("====== L values ======")
print(f"(l_0 + l_1 + l_2) = {pre_L}")
print(f"L = 1/3 (l_0 + l_1 + l_2) = {L}")
print("======================\n")
```

```
====== l_0 value ======
l_0 = CE(y_0, y_hat_0) = 0.17539014728134006
=======================

====== l_1 value ======
l_1 = CE(y_0, y_hat_1) = 0.30967033498751567
=======================

====== l_2 value ======
l_2 = CE(y_2, y_hat_2) = 5.55807095760499
=======================

====== L values ======
(l_0 + l_1 + l_2) = 6.043131439873846
L = 1/3 (l_0 + l_1 + l_2) = 2.014377146624615
======================
```

Question 3.4

In this question I will be using the following notations:

- t subscript (e.g., x_t) represents the t-th timestamp.
- 1_y is a one-hot vector representing the truth label for some input x
  - Since y_n is already a one-hot vector, then $1_{yt} = y_t$
  - I still use this notation for consistency with the lecture slides/content
- 1_{yt} is a one-hot vector for the t-th timestamp
- 1_{ytm} is the m-th element of the one-hot vector for the t-th timestamp
- k subscript (e.g., x_k) is the truth label for some input k.
  - So, if I had a truth label of 3 and $y = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix}$, then $y_k = y_3 = 6$

The gradient (partial derivative) $\frac{\partial L}{\partial h_1}$ consists of several other gradients. It can be computed using a formula, which can be derived like so:

$$L = \frac{1}{3}(l_0 + l_1 + l_2)$$

$$\begin{aligned} \frac{\partial L}{\partial h_1} &= \frac{\partial}{\partial h_1}\left(\frac{1}{3}(l_0 + l_1 + l_2)\right) \\ &= \frac{1}{3}\left(\frac{\partial l_1}{\partial h_1} + \frac{\partial l_2}{\partial h_1}\right) \end{aligned}$$

$$\frac{\partial l_1}{\partial h_1} = \frac{\partial l_1}{\partial \hat{y}_1} \times \frac{\partial \hat{y}_1}{\partial h_1}$$

$$\frac{\partial l_2}{\partial h_1} = \frac{\partial l_2}{\partial \hat{y}_2} \times \frac{\partial \hat{y}_2}{\partial h_2} \times \frac{\partial h_2}{\partial \bar{h}_2} \times \frac{\partial \bar{h}_2}{\partial h_1}$$

Let $\mu_t = Vh_t + c$. So,

$$\mu_t = Vh_t + c$$
$$\frac{\partial \mu_t}{\partial h_t} = \frac{\partial \mu_1}{\partial h_1} = \frac{\partial \mu_2}{\partial h_2} = V$$

$$\hat{y}_t = softmax(Vh_t + c) = softmax(\mu_t)$$

With this addition, we can alter our formulas accordingly

$$L = \frac{1}{3}(l_0 + l_1 + l_2)$$

$$\frac{\partial L}{\partial h_1} = \frac{\partial}{\partial h_1}\left(\frac{1}{3}(l_0 + l_1 + l_2)\right)$$
$$= \frac{1}{3}\left(\frac{\partial l_1}{\partial h_1} + \frac{\partial l_2}{\partial h_1}\right)$$

$$\frac{\partial l_1}{\partial h_1} = \frac{\partial l_1}{\partial \hat{y}_1} \times \frac{\partial \hat{y}_1}{\partial \mu_1} \times \frac{\partial \mu_1}{\partial h_1}$$

$$\frac{\partial l_2}{\partial h_1} = \frac{\partial l_2}{\partial \hat{y}_2} \times \frac{\partial \hat{y}_2}{\partial \mu_2} \times \frac{\partial \mu_2}{\partial h_2} \times \frac{\partial h_2}{\partial \bar{h}_2} \times \frac{\partial \bar{h}_2}{\partial h_1}$$

With that, we can now find the individual derivatives.

$$\bar{h}_t = Wh_{t-1} + Ux_t + b$$
$$\frac{\partial \bar{h}_t}{\partial h_{t-1}} = \frac{\partial}{\partial h_{t-1}}(Wh_{t-1} + Ux_t + b) = W$$

$$h_t = tanh(\bar{h}_t)$$
$$\frac{\partial h_t}{\partial \bar{h}_t} = \frac{\partial}{\partial \bar{h}_t} tanh(\bar{h}_t)$$
$$= 1 - \tanh^2(\bar{h}_t)$$
$$= 1 - h_t^2$$

$$\mu_t = Vh_t + c$$
$$\frac{\partial \mu_t}{\partial h_t} = \frac{\partial \mu_1}{\partial h_1} = \frac{\partial \mu_2}{\partial h_2} = V$$

$$
\begin{aligned}
y_t &= softmax(\mu_t) \\
&= \frac{\exp\{\mu_{tk}\}}{\sum_{m=1}^{M} \exp\{\mu_{tm}\}}
\end{aligned}
$$

$$
\begin{aligned}
l_t &= CE(y_t, \hat{y}_t) = CE(1_{yt}, \hat{y}_t) \\
&= -\sum_{m=1}^{M} 1_{ytm} \log \hat{y}_{tm} \\
&= -\log y_{tk} \\
&= -\log\left(\frac{\exp\{\mu_{tk}\}}{\sum_{m=1}^{M} \exp\{\mu_{tm}\}}\right) \\
&= \log \sum_{m=1}^{M} \exp\{\mu_{tm}\} - \log \exp\{\mu_{tk}\} \\
&= \log \sum_{m=1}^{M} \exp\{\mu_{tm}\} - \mu_{tk}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial l_t}{\partial \mu_t} &= \frac{\partial}{\partial \mu_t}\left(\log \sum_{m=1}^{M} \exp\{\mu_{tm}\} - \mu_{tk}\right) \\
&= \frac{\exp\{\mu_{tk}\}}{\sum_{m=1}^{M} \exp\{\mu_{tm}\}} - 1_{yt} \\
&= softmax(\mu_t) - 1_{yt} \\
&= \hat{y}_t - 1_{yt} = \hat{y}_t - y_t
\end{aligned}
$$

With these gradient formulas, we can now compute $\frac{\partial L}{\partial h_1}$

$$
\begin{aligned}
\frac{\partial l_1}{\partial \mu_1} &= \hat{y}_1 - y_1 \\
&= \begin{bmatrix} 0.0610 \\ 0.7337 \\ 0.2053 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \\
&= \begin{bmatrix} 0.0610 \\ -0.2663 \\ 0.2053 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mu_1}{\partial h_1} &= V \\
&= \begin{bmatrix} 2 & -1 & 0 \\ 1 & 1 & -1 \\ -1 & 2 & 1 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial l_1}{\partial h_1} &= \frac{\partial l_1}{\partial \hat{y}_1} \times \frac{\partial \hat{y}_1}{\partial \mu_1} \times \frac{\partial \mu_1}{\partial h_1} \\
&= \frac{\partial l_1}{\partial \mu_1} \times \frac{\partial \mu_1}{\partial h_1} \\
&= \begin{bmatrix} 0.0610 \\ -0.2663 \\ 0.2053 \end{bmatrix}^T \begin{bmatrix} 2 & -1 & 0 \\ 1 & 1 & -1 \\ -1 & 2 & 1 \end{bmatrix} \\
&= \begin{bmatrix} -0.34954964 & 0.08323843 & 0.47159816 \end{bmatrix} \\
&\approx \begin{bmatrix} -0.3495 & 0.0832 & 0.4716 \end{bmatrix}
\end{aligned}
$$

In [ ]:

```
def diag(vect):
    return tf.constant(
        [
            [
                vect[index][0].numpy() if index == i else 0
                for index in range(vect.shape[0])
            ]
            for i in range(vect.shape[0])
        ]
    )


y_1 = tf.cast(y_1, dtype="float64")
dl1_dmu1 = tf.subtract(y_hat_1, y_1)
dl1_dh1 = tf.matmul(tf.transpose(dl1_dmu1), V)

print("====== l_1 values ======")
print(f"dl_1/dmu_1 = \n{dl1_dmu1}")
print(f"dmu_1/dh_1 = V = \n{V}")
print(f"dl_1/dh_1 = \n{dl1_dh1}")
print("========================\n")
```

```
====== l_1 values ======
dl_1/dmu_1 =
[[ 0.06102426]
 [-0.26631121]
 [ 0.20528695]]
dmu_1/dh_1 = V =
[[ 2. -1.  0.]
 [ 1.  1. -1.]
 [-1.  2.  1.]]
dl_1/dh_1 =
[[-0.34954964  0.08323843  0.47159816]]
========================
```

$$
\begin{aligned}
\frac{\partial l_2}{\partial \mu_2} &= \hat{y}_2 - y_2 \\
&= \begin{bmatrix} 0.9394 \\ 0.0567 \\ 0.0039 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} 0.9394 \\ 0.0567 \\ -0.9961 \end{bmatrix} \\
\frac{\partial \mu_2}{\partial h_2} &= V \\
&= \begin{bmatrix} 2 & -1 & 0 \\ 1 & 1 & -1 \\ -1 & 2 & 1 \end{bmatrix} \\
\frac{\partial h_2}{\partial \bar{h}_2} &= 1 - h_2^2 \\
&= \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 0.9976^2 \\ -0.6624^2 \\ 0.4844^2 \end{bmatrix} \\
&= \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 0.99517279 \\ 0.43876962 \\ 0.2346244 \end{bmatrix} \\
&= \begin{bmatrix} 0.00482721 \\ 0.56123038 \\ 0.7653756 \end{bmatrix} \\
&\approx \begin{bmatrix} 0.0048 \\ 0.5612 \\ 0.7654 \end{bmatrix} \\
\frac{\partial \bar{h}_2}{\partial h_1} &= W \\
&= \begin{bmatrix} 1 & 0 & -1 \\ 2 & 1 & 0 \\ -1 & 2 & 1 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial l_2}{\partial h_1} &= \frac{\partial l_2}{\partial \hat{y}_2} \times \frac{\partial \hat{y}_2}{\partial \mu_2} \times \frac{\partial \mu_2}{\partial h_2} \times \frac{\partial h_2}{\partial \bar{h}_2} \times \frac{\partial \bar{h}_2}{\partial h_1} \\
&= \frac{\partial l_2}{\partial \mu_2} \times \frac{\partial \mu_2}{\partial h_2} \times \frac{\partial h_2}{\partial \bar{h}_2} \times \frac{\partial \bar{h}_2}{\partial h_1} \\
&= \begin{bmatrix} 0.9394 \\ 0.0567 \\ -0.9961 \end{bmatrix}^T \begin{bmatrix} 2 & -1 & 0 \\ 1 & 1 & -1 \\ -1 & 2 & 1 \end{bmatrix} diag\left(\begin{bmatrix} 0.0048 \\ 0.5612 \\ 0.7654 \end{bmatrix}\right) \begin{bmatrix} 1 & 0 & -1 \\ 2 & 1 & 0 \\ -1 & 2 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 0.9394 \\ 0.0567 \\ -0.9961 \end{bmatrix}^T \begin{bmatrix} 2 & -1 & 0 \\ 1 & 1 & -1 \\ -1 & 2 & 1 \end{bmatrix} \begin{bmatrix} 0.0048 & 0 & 0 \\ 0 & 0.5612 & 0 \\ 0 & 0 & 0.7654 \end{bmatrix} \begin{bmatrix} 1 & 0 & -1 \\ 2 & 1 & 0 \\ -1 & 2 & 1 \end{bmatrix} \\
&= \begin{bmatrix} -2.40701532 & -3.22521336 & -0.82000379 \end{bmatrix}
\end{aligned}
$$

In [ ]:
```
dl2_dmu2 = tf.subtract(y_hat_2, y_2)
dh2_dhbar2 = tf.subtract(1, tf.math.multiply(h_2, h_2))
dl2_dh1 = tf.matmul(
    tf.matmul(tf.matmul(tf.transpose(dl2_dmu2), V), diag(dh2_dhbar2)), W
)

print("====== l_2 values ======")
print(f"dl_2/dmu_2 = \n{dl2_dmu2}")
print(f"dmu_2/dh_2 = V = \n{V}")
print(f"dh_2/dh_bar_2 = \n{dh2_dhbar2}")
print(f"dhl2_dh1 = \n{dl2_dh1}")
print("========================\n")
```

```
====== l_2 values ======
dl_2/dmu_2 =
[[ 0.93940335]
 [ 0.05674045]
 [-0.99614379]]
dmu_2/dh_2 = V =
[[ 2. -1.  0.]
 [ 1.  1. -1.]
 [-1.  2.  1.]]
dh_2/dh_bar_2 =
[[0.00482721]
 [0.56123038]
 [0.7653756 ]]
dhl2_dh1 =
[[-2.40701532 -3.22521336 -0.82000379]]
========================
```

$$
\begin{aligned}
\frac{\partial L}{\partial h_1} &= \frac{1}{3}\left(\frac{\partial l_1}{\partial h_1} + \frac{\partial l_2}{\partial h_1}\right) \\
&= \frac{1}{3}\left(\begin{bmatrix} -0.3495 \\ 0.0832 \\ 0.4716 \end{bmatrix}^T + \begin{bmatrix} -2.40701532 \\ -3.22521336 \\ -0.82000379 \end{bmatrix}^T\right) \\
&= \begin{bmatrix} -0.91885499 & -1.04732498 & -0.11613521 \end{bmatrix}
\end{aligned}
$$

In [ ]:
```
dL_dh1 = tf.math.scalar_mul(1/3, tf.add(dl1_dh1, dl2_dh1))

print("====== L values ======")
print(f"dL/dh_1 = \n{dL_dh1}")
print("======================\n")
```

```
====== L values ======
dL/dh_1 =
[[-0.91885499 -1.04732498 -0.11613521]]
======================
```

Question 3.5

In this question, I will be using the same definition of μ_t from **Question 3.4**. Therefore,

$$\begin{aligned}\frac{\partial L}{\partial V} &= \frac{\partial}{\partial V}\left[\frac{1}{3}(l_0 + l_1 + l_2)\right] \\ &= \frac{1}{3}\left[\frac{\partial l_0}{\partial V} + \frac{\partial l_1}{\partial V} + \frac{\partial l_2}{\partial V}\right]\end{aligned}$$

$$\begin{aligned}\frac{\partial l_t}{\partial V} &= \frac{\partial l_t}{\partial \hat{y}_t} \times \frac{\partial \hat{y}_t}{\partial \mu_t} \times \frac{\partial \mu_t}{\partial V} \\ &= \frac{\partial l_t}{\partial \mu_t} \times \frac{\partial \mu_t}{\partial V}\end{aligned}$$

$$\frac{\partial l_t}{\partial \mu_t} = \hat{y}_t - y_t$$

$$\begin{aligned}\mu_t &= Vh_t + c \\ \frac{\partial \mu_t}{\partial V} &= \frac{\partial}{\partial V}(Vh_t + c) \\ &= h_t\end{aligned}$$

$$
\begin{aligned}
\frac{\partial l_0}{\partial \mu_0} &= \hat{y}_0 - y_0 \\
&= \begin{bmatrix} 0.0885 \\ 0.0723 \\ 0.8391 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} 0.0885 \\ 0.0723 \\ -0.1609 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mu_0}{\partial V} &= h_0 \\
&= \begin{bmatrix} -0.7616 \\ 0.0000 \\ 0.9640 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial l_0}{\partial V} &= \frac{\partial l_0}{\partial \mu_0} \times \frac{\partial \mu_0}{\partial V} \\
&= \begin{bmatrix} 0.0885 \\ 0.0723 \\ -0.1609 \end{bmatrix} \begin{bmatrix} -0.7616 \\ 0.0000 \\ 0.9640 \end{bmatrix}^T \\
&= \begin{bmatrix} -0.06743833 & 0.00000000 & 0.08536359 \\ -0.05507964 & 0.00000000 & 0.06971993 \\ 0.12251798 & 0.00000000 & -0.15508353 \end{bmatrix} \\
&\approx \begin{bmatrix} -0.0675 & 0.0000 & 0.0857 \\ -0.0551 & 0.0000 & 0.0697 \\ 0.1225 & 0.0000 & -0.1551 \end{bmatrix}
\end{aligned}
$$

In [ ]:

```python
dl0_dmu0 = y_hat_0 - y_0
dl0_dV = tf.matmul(dl0_dmu0, tf.transpose(h_0))

print("====== l_0 / V ======")
print(f"dl_0 / dmu_0 = \n{dl0_dmu0}")
print(f"dmu_0 / dV = \n{h_0}")
print(f"dl_0 / dV = \n{dl0_dV}")
print("=====================\n")
```

```
====== l_0 / V ======
dl_0 / dmu_0 =
[[ 0.08854891]
 [ 0.07232151]
 [-0.16087043]]
dmu_0 / dV =
[[-0.76159416]
 [ 0.        ]
 [ 0.96402758]]
dl_0 / dV =
[[-0.06743833  0.          0.08536359]
 [-0.05507964  0.          0.06971993]
 [ 0.12251798  0.         -0.15508353]]
=====================
```

$$
\begin{aligned}
\frac{\partial l_1}{\partial \mu_1} &= \hat{y}_1 - y_1 \\
&= \begin{bmatrix} 0.0610 \\ 0.7337 \\ 0.2053 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \\
&= \begin{bmatrix} 0.0610 \\ -0.2663 \\ 0.2053 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mu_1}{\partial V} &= h_1 \\
&= \begin{bmatrix} -0.6204 \\ 0.4437 \\ -0.9791 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial l_1}{\partial V} &= \frac{\partial l_1}{\partial \mu_1} \times \frac{\partial \mu_1}{\partial V} \\
&= \begin{bmatrix} 0.0610 \\ -0.2663 \\ 0.2053 \end{bmatrix} \begin{bmatrix} -0.6204 \\ 0.4437 \\ -0.9791 \end{bmatrix}^T \\
&= \begin{bmatrix} -0.0378582 & 0.02707564 & -0.05974646 \\ 0.16521401 & -0.11815871 & 0.26073488 \\ -0.12735581 & 0.09108306 & -0.20098841 \end{bmatrix} \\
&\approx \begin{bmatrix} -0.0379 & 0.0271 & -0.0598 \\ 0.1652 & -0.1182 & 0.2607 \\ -0.1274 & 0.0911 & -0.2010 \end{bmatrix}
\end{aligned}
$$

In [ ]:
```python
dl1_dmu1 = y_hat_1 - y_1
dl1_dV = tf.matmul(dl1_dmu1, tf.transpose(h_1))

print("====== l_1 / V ======")
print(f"dl_1 / dmu_1 = \n{dl1_dmu1}")
print(f"dmu_1 / dV = \n{h_1}")
print(f"dl_1 / dV = \n{dl1_dV}")
print("=====================\n")
```

```
====== l_1 / V ======
dl_1 / dmu_1 =
[[ 0.06102426]
 [-0.26631121]
 [ 0.20528695]]
dmu_1 / dV =
[[-0.62037946]
 [ 0.44368657]
 [-0.97906084]]
dl_1 / dV =
[[-0.0378582   0.02707564 -0.05974646]
 [ 0.16521401 -0.11815871  0.26073488]
 [-0.12735581  0.09108306 -0.20098841]]
=====================
```

$$
\begin{aligned}
\frac{\partial l_2}{\partial \mu_2} &= \hat{y}_2 - y_2 \\
&= \begin{bmatrix} 0.9394 \\ 0.0567 \\ 0.0039 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} 0.0885 \\ 0.0723 \\ -0.9961 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mu_2}{\partial V} &= h_2 \\
&= \begin{bmatrix} 0.9976 \\ -0.6624 \\ 0.4844 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial l_2}{\partial V} &= \frac{\partial l_2}{\partial \mu_2} \times \frac{\partial \mu_2}{\partial V} \\
&= \begin{bmatrix} 0.0885 \\ 0.0723 \\ -0.9961 \end{bmatrix} \begin{bmatrix} 0.9976 \\ -0.6624 \\ 0.4844 \end{bmatrix}^T \\
&= \begin{bmatrix} 0.93713325 & -0.62225784 & 0.4550286 \\ 0.05660333 & -0.03758469 & 0.02748396 \\ -0.99373659 & 0.65984253 & -0.48251256 \end{bmatrix} \\
&\approx \begin{bmatrix} 0.9371 & -0.6223 & 0.4550 \\ 0.0566 & -0.0376 & 0.0275 \\ -0.9937 & 0.6598 & -0.4825 \end{bmatrix}
\end{aligned}
$$

In [ ]:
```python
dl2_dmu2 = y_hat_2 - y_2
dl2_dV = tf.matmul(dl2_dmu2, tf.transpose(h_2))

print("====== l_2 / V ======")
print(f"dl_2 / dmu_2 = \n{dl2_dmu2}")
print(f"dmu_2 / dV = \n{h_2}")
print(f"dl_2 / dV = \n{dl2_dV}")
print("=====================\n")
```

```
====== l_2 / V ======
dl_2 / dmu_2 =
[[ 0.93940335]
 [ 0.05674045]
 [-0.99614379]]
dmu_2 / dV =
[[ 0.99758347]
 [-0.66239687]
 [ 0.48438043]]
dl_2 / dV =
[[ 0.93713325 -0.62225784  0.4550286 ]
 [ 0.05660333 -0.03758469  0.02748396]
 [-0.99373659  0.65984253 -0.48251256]]
=====================
```

Finally, we can compute $\frac{\partial L}{\partial V}$.

$$\frac{\partial L}{\partial V} = \frac{1}{3}\left[\frac{\partial l_0}{\partial V} + \frac{\partial l_1}{\partial V} + \frac{\partial l_2}{\partial V}\right]$$

$$= \frac{1}{3}\left[\begin{bmatrix} -0.0675 & 0.0000 & 0.0857 \\ -0.0551 & 0.0000 & 0.0697 \\ 0.1225 & 0.0000 & -0.1551 \end{bmatrix} + \begin{bmatrix} -0.0379 & 0.0271 & -0.0598 \\ 0.1652 & -0.1182 & 0.2607 \\ -0.1274 & 0.0911 & -0.2010 \end{bmatrix} + \begin{bmatrix} 0.9371 & -0. \\ 0.0566 & -0. \\ -0.9937 & 0.6 \end{bmatrix}\right.$$

$$= \begin{bmatrix} 0.27727891 & -0.19839407 & 0.16021524 \\ 0.05557923 & -0.05191447 & 0.11931292 \\ -0.33285814 & 0.25030853 & -0.27952817 \end{bmatrix}$$

$$\approx \begin{bmatrix} 0.2773 & -0.1984 & 0.1602 \\ 0.0556 & -0.0519 & 0.1193 \\ -0.3329 & 0.2503 & -0.2795 \end{bmatrix}$$

In [ ]:
```python
dL_dV = (dl0_dV + dl1_dV + dl2_dV) / 3

print("====== dL / dV ======")
print(f"dL / dV = \n{dL_dV}")
print("============================\n")
```

```
====== dL / dV ======
dL / dV =
[[ 0.27727891 -0.19839407  0.16021524]
 [ 0.05557923 -0.05191447  0.11931292]
 [-0.33285814  0.25030853 -0.27952817]]
============================
```

Question 3.6

In this question, I will be using the same definition of μ_t from **Question 3.4**. Therefore,

$$\frac{\partial L}{\partial U} = \frac{\partial}{\partial U}\left[\frac{1}{3}(l_0 + l_1 + l_2)\right]$$

$$= \frac{1}{3}\left(\frac{\partial l_0}{\partial U} + \frac{\partial l_1}{\partial U} + \frac{\partial l_2}{\partial U}\right)$$

$$\frac{\partial l_t}{\partial U} = \frac{\partial l_t}{\partial \hat{y}_t} \times \frac{\partial \hat{y}_t}{\partial \mu_t} \times \frac{\partial \mu_t}{\partial h_t} \times \frac{\partial h_t}{\partial \bar{h}_t} \times \frac{\partial \bar{h}_t}{\partial U}$$
$$= \frac{\partial l_t}{\partial \mu_t} \times \frac{\partial \mu_t}{\partial h_t} \times \frac{\partial h_t}{\partial \bar{h}_t} \times \frac{\partial \bar{h}_t}{\partial U}$$

$$\frac{\partial l_0}{\partial U} = \frac{\partial l_0}{\partial \mu_0} \times \frac{\partial \mu_0}{\partial h_0} \times \frac{\partial h_0}{\partial \bar{h}_0} \times \frac{\partial \bar{h}_0}{\partial U}$$

$$\frac{\partial l_0}{\partial \mu_0} = \begin{bmatrix} 0.0885 \\ 0.0723 \\ -0.1609 \end{bmatrix} \text{ from Question 3.5}$$

$$\frac{\partial \mu_0}{\partial h_0} = V = \begin{bmatrix} 2 & -1 & 0 \\ 1 & 1 & -1 \\ -1 & 2 & 1 \end{bmatrix}$$

$$\begin{aligned} \frac{\partial h_0}{\partial \bar{h}_0} &= 1 - h_0^2 \\ &= \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} -0.7616^2 \\ 0.0000^2 \\ 0.9640^2 \end{bmatrix} \\ &= \begin{bmatrix} 0.41997434 \\ 1.000 \\ 0.07065082 \end{bmatrix} \approx \begin{bmatrix} 0.4200 \\ 1.000 \\ 0.0707 \end{bmatrix} \\ \frac{\partial \bar{h}_0}{\partial U} &= \frac{\partial}{\partial U}(Ux_0 + b) \\ &= x_0 = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix} \end{aligned}$$

$$\begin{aligned} \frac{\partial l_0}{\partial U} &= \frac{\partial l_0}{\partial \mu_0} \times \frac{\partial \mu_0}{\partial h_0} \times \frac{\partial h_0}{\partial \bar{h}_0} \times \frac{\partial \bar{h}_0}{\partial U} \\ &= \begin{bmatrix} 0.0885 \\ 0.0723 \\ -0.1609 \end{bmatrix}^T \begin{bmatrix} 2 & -1 & 0 \\ 1 & 1 & -1 \\ -1 & 2 & 1 \end{bmatrix} diag\left(\begin{bmatrix} 0.4200 \\ 1.000 \\ 0.0707 \end{bmatrix}\right) \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix} \\ &= 0.51027943 \approx 0.513 \end{aligned}$$

In [ ]:
```
dh0_dhbar0 = tf.subtract(1, tf.math.multiply(h_0, h_0))
dl0_dU = tf.matmul(tf.matmul(tf.matmul(tf.transpose(dl0_dmu0), V), diag(dh0_dhbar0)), x_0)

print("====== dl_0 / dU values ======")
print(f"dl_0 / dmu_0 = \n{dl0_dmu0}")
print(f"dmu_0 / dh_0 = \n{V}")
print(f"dh_0 / dh_bar_0 = \n{dh0_dhbar0}")
print(f"dh_bar_0 / dU = \n{x_0}")
print(f"dl_0 / dU = \n{dl0_dU}")
print("==============================\n")
```

```
====== dl_0 / dU values ======
dl_0 / dmu_0 =
[[ 0.08854891]
 [ 0.07232151]
 [-0.16087043]]
dmu_0 / dh_0 =
[[ 2. -1.  0.]
 [ 1.  1. -1.]
 [-1.  2.  1.]]
dh_0 / dh_bar_0 =
[[0.41997434]
 [1.        ]
 [0.07065082]]
dh_bar_0 / dU =
[[ 1.]
 [-1.]
 [ 0.]]
dl_0 / dU =
[[0.51027943]]
==============================
```

$$\frac{\partial l_1}{\partial U} = \frac{\partial l_1}{\partial \mu_1} \times \frac{\partial \mu_1}{\partial h_1} \times \frac{\partial h_1}{\partial \bar{h}_1} \times \frac{\partial \bar{h}_1}{\partial U}$$

$$\frac{\partial l_1}{\partial \mu_1} = \begin{bmatrix} 0.0610 \\ -0.2663 \\ 0.2053 \end{bmatrix} \text{from Question 3.4}$$

$$\begin{aligned} \frac{\partial \mu_1}{\partial h_1} &= V \\ &= \begin{bmatrix} 2 & -1 & 0 \\ 1 & 1 & -1 \\ -1 & 2 & 1 \end{bmatrix} \end{aligned}$$

$$\begin{aligned} \frac{\partial h_1}{\partial \bar{h}_1} &= 1 - h_1^2 \\ &= \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} -0.6204^2 \\ 0.4437^2 \\ -0.9791^2 \end{bmatrix} \\ &= \begin{bmatrix} 0.61512932 \\ 0.80314223 \\ 0.04143987 \end{bmatrix} \approx \begin{bmatrix} 0.6151 \\ 0.8031 \\ 0.0414 \end{bmatrix} \end{aligned}$$

$$
\begin{aligned}
\frac{\partial \bar{h}_1}{\partial U} &= \frac{\partial}{\partial U}(Wh_0 + Ux_1 + b) \\
&= W\left(\frac{\partial h_0}{\partial U}\right) + x_1 \\
&= W\left(\frac{\partial h_0}{\partial \bar{h}_0} \times \frac{\partial \bar{h}_0}{\partial U}\right) + x_1 \\
&= W\left(diag(1 - h_0^2)\frac{\partial \bar{h}_0}{\partial U}\right) + x_1 \\
&= \begin{bmatrix} 1 & 0 & -1 \\ 2 & 1 & 0 \\ -1 & 2 & 1 \end{bmatrix}\left(diag\left(\begin{bmatrix} 0.4200 \\ 1.000 \\ 0.0707 \end{bmatrix}\right)\begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}\right) + \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix} \quad \frac{\partial \bar{h}_0}{\partial U} \text{ from previous calculation} \\
&= \begin{bmatrix} -0.58002566 \\ 0.83994868 \\ -2.41997434 \end{bmatrix} \approx \begin{bmatrix} -0.5800 \\ 0.8399 \\ -2.4200 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial l_1}{\partial U} &= \frac{\partial l_1}{\partial \mu_1} \times \frac{\partial \mu_1}{\partial h_1} \times \frac{\partial h_1}{\partial \bar{h}_1} \times \frac{\partial \bar{h}_1}{\partial U} \\
&= \begin{bmatrix} 0.9394 \\ 0.0567 \\ -0.9961 \end{bmatrix}^T \begin{bmatrix} 2 & -1 & 0 \\ 1 & 1 & -1 \\ -1 & 2 & 1 \end{bmatrix} diag\left(\begin{bmatrix} 0.6151 \\ 0.8031 \\ 0.0414 \end{bmatrix}\right)\begin{bmatrix} -0.5800 \\ 0.8399 \\ -2.4200 \end{bmatrix} \\
&= 0.13357511 \approx 0.1336
\end{aligned}
$$

In [ ]:

```
dh1_dhbar1 = tf.subtract(1, tf.math.multiply(h_1, h_1))
dhbar1_dU = tf.add(
    tf.matmul(W, tf.matmul(diag(tf.subtract(1, tf.multiply(h_0, h_0))), x_0)), x_1
)
dl1_dU = tf.matmul(tf.matmul(tf.matmul(tf.transpose(dl1_dmu1), V), diag(dh1_dhbar1)), dhbar1_

print("====== dl_1 / dU values ======")
print(f"dl_1 / dmu_1 = \n{dl1_dmu1}")
print(f"dmu_1 / dh_1 = \n{V}")
print(f"dh_1 / dh_bar_1 = \n{dh1_dhbar1}")
print(f"dh_bar_1 / dU = \n{dhbar1_dU}")
print(f"dl_1 / dU = \n{dl1_dU}")
print("==============================\n")
```

```
====== dl_1 / dU values ======
dl_1 / dmu_1 =
[[ 0.06102426]
 [-0.26631121]
 [ 0.20528695]]
dmu_1 / dh_1 =
[[ 2. -1.  0.]
 [ 1.  1. -1.]
 [-1.  2.  1.]]
dh_1 / dh_bar_1 =
[[0.61512932]
 [0.80314223]
 [0.04143987]]
dh_bar_1 / dU =
[[-0.58002566]
 [ 0.83994868]
 [-2.41997434]]
dl_1 / dU =
[[0.13357511]]
==============================
```

$$\frac{\partial l_2}{\partial U} = \frac{\partial l_2}{\partial \mu_2} \times \frac{\partial \mu_2}{\partial h_2} \times \frac{\partial h_2}{\partial \bar{h}_2} \times \frac{\partial \bar{h}_2}{\partial U}$$

$$\frac{\partial l_2}{\partial \mu_2} = \begin{bmatrix} 0.0885 \\ 0.0723 \\ -0.9961 \end{bmatrix} \text{ from Question 3.4}$$

$$\frac{\partial \mu_2}{\partial h_2} = V = \begin{bmatrix} 2 & -1 & 0 \\ 1 & 1 & -1 \\ -1 & 2 & 1 \end{bmatrix}$$

$$\frac{\partial h_2}{\partial \bar{h}_2} = \begin{bmatrix} 0.0048 \\ 0.5612 \\ 0.7654 \end{bmatrix} \text{ from Question 3.4}$$

$$\begin{aligned}
\frac{\partial \bar{h}_2}{\partial U} &= \frac{\partial}{\partial U}(Wh_1 + Ux_2 + b) \\
&= W\left(\frac{\partial h_1}{\partial U}\right) + x_2 \\
&= W\left(\frac{\partial h_1}{\partial \bar{h}_1} \times \frac{\partial \bar{h}_1}{\partial U}\right) + x_2 \\
&= W\left(diag(1 - h_1^2)\, \frac{\partial \bar{h}_1}{\partial U}\right) + x_2 \\
&= \begin{bmatrix} 1 & 0 & -1 \\ 2 & 1 & 0 \\ -1 & 2 & 1 \end{bmatrix} \left(diag\left(\begin{bmatrix} 0.6151 \\ 0.8031 \\ 0.0414 \end{bmatrix}\right) \begin{bmatrix} -0.5800 \\ 0.8399 \\ -2.4200 \end{bmatrix}\right) + \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \quad \frac{\partial \bar{h}_1}{\partial U} \text{ from previous calculat:} \\
&= \begin{bmatrix} 0.74349264 \\ 0.96101668 \\ 1.60570387 \end{bmatrix} \approx \begin{bmatrix} 0.7435 \\ 0.9610 \\ 1.6057 \end{bmatrix}
\end{aligned}$$

$$\begin{aligned}
\frac{\partial l_2}{\partial U} &= \frac{\partial l_2}{\partial \mu_2} \times \frac{\partial \mu_2}{\partial h_2} \times \frac{\partial h_2}{\partial \bar{h}_2} \times \frac{\partial \bar{h}_2}{\partial U} \\
&= \begin{bmatrix} 0.0885 \\ 0.0723 \\ -0.9961 \end{bmatrix}^T \begin{bmatrix} 2 & -1 & 0 \\ 1 & 1 & -1 \\ -1 & 2 & 1 \end{bmatrix} diag\left(\begin{bmatrix} 0.0048 \\ 0.5612 \\ 0.7654 \end{bmatrix}\right) \begin{bmatrix} 0.7435 \\ 0.9610 \\ 1.6057 \end{bmatrix} \\
&= -2.83404729 \approx -2.8340
\end{aligned}$$

In [ ]:

```
dhbar2_dU = tf.add(
    tf.matmul(W, tf.matmul(diag(tf.subtract(1, tf.multiply(h_1, h_1))), dhbar1_dU)), x_2
)
dl2_dU = tf.matmul(
    tf.matmul(tf.matmul(tf.transpose(dl2_dmu2), V), diag(dh2_dhbar2)), dhbar2_dU
)

print("====== dl_1 / dU values ======")
print(f"dl_2 / dmu_2 = \n{dl2_dmu2}")
print(f"dmu_2 / dh_2 = \n{V}")
print(f"dh_2 / dh_bar_2 = \n{dh2_dhbar2}")
print(f"dh_bar_2 / dU = \n{dhbar2_dU}")
```

```
print(f"dl_2 / dU = \n{dl2_dU}")
print("==============================\n")
```

```
====== dl_1 / dU values ======
dl_2 / dmu_2 =
[[ 0.93940335]
 [ 0.05674045]
 [-0.99614379]]
dmu_2 / dh_2 =
[[ 2. -1.  0.]
 [ 1.  1. -1.]
 [-1.  2.  1.]]
dh_2 / dh_bar_2 =
[[0.00482721]
 [0.56123038]
 [0.7653756 ]]
dh_bar_2 / dU =
[[0.74349264]
 [0.96101668]
 [1.60570387]]
dl_2 / dU =
[[-2.83404729]]
==============================
```

Finally, we can compute $\frac{\partial L}{\partial U}$.

$$\begin{aligned}\frac{\partial L}{\partial U} &= \frac{1}{3}\left[\frac{\partial l_0}{\partial U} + \frac{\partial l_1}{\partial U} + \frac{\partial l_2}{\partial U}\right] \\ &= \frac{1}{3}[0.513 + 0.1336 + (-2.8340)] \\ &= -0.73006425 \approx -0.7300\end{aligned}$$

In [ ]:
```
dL_dU = (dl0_dU + dl1_dU + dl2_dU) / 3

print("====== dL / dU ======")
print(f"dL / dU = \n{dL_dU}")
print("====================\n")
```

```
====== dL / dV ======
dL / dV =
[[-0.73006425]]
============================
```

Appendix

Question 3.4

$$\begin{aligned}\frac{\partial L}{\partial h_1} &= \frac{1}{3}\left[\frac{\partial l_1}{\partial h_1} + \frac{\partial l_2}{\partial h_1}\right] \\ &= \frac{1}{3}\left[\left(\frac{\partial l_1}{\partial \mu_1} \times \frac{\partial \mu_1}{\partial h_1}\right) + \left(\frac{\partial l_2}{\partial \mu_2} \times \frac{\partial \mu_2}{\partial h_2} \times \frac{\partial h_2}{\partial \bar{h}_2} \times \frac{\partial \bar{h}_2}{\partial h_1}\right)\right]\end{aligned}$$

Question 3.5

$$
\begin{aligned}
\frac{\partial L}{\partial V} &= \frac{1}{3}\left[\frac{\partial l_0}{\partial V} + \frac{\partial l_1}{\partial V} + \frac{\partial l_2}{\partial V}\right] \\
&= \frac{1}{3}\left[\left(\frac{\partial l_0}{\partial \mu_0} \times \frac{\partial \mu_0}{\partial V}\right) + \left(\frac{\partial l_1}{\partial \mu_1} \times \frac{\partial \mu_1}{\partial V}\right) + \left(\frac{\partial l_2}{\partial \mu_2} \times \frac{\partial \mu_2}{\partial V}\right)\right] \\
&= \frac{1}{3}\left[\left(\frac{\partial l_0}{\partial \mu_0} \times h_0\right) + \left(\frac{\partial l_1}{\partial \mu_1} \times h_1\right) + \left(\frac{\partial l_2}{\partial \mu_2} \times h_2\right)\right]
\end{aligned}
$$

Question 3.6

$$
\begin{aligned}
\frac{\partial L}{\partial U} &= \frac{1}{3}\left[\frac{\partial l_0}{\partial U} + \frac{\partial l_1}{\partial U} + \frac{\partial l_2}{\partial U}\right] \\
&= \frac{1}{3}\left[\left(\frac{\partial l_0}{\partial \mu_0} \times \frac{\partial \mu_0}{\partial h_0} \times \frac{\partial h_0}{\partial \bar{h}_0} \times \frac{\partial \bar{h}_0}{\partial U}\right) + \left(\frac{\partial l_1}{\partial \mu_1} \times \frac{\partial \mu_1}{\partial h_1} \times \frac{\partial h_1}{\partial \bar{h}_1} \times \frac{\partial \bar{h}_1}{\partial U}\right) + \left(\frac{\partial l_2}{\partial \mu_2} \times \frac{\partial \mu_2}{\partial h_2} \times \frac{\partial h_2}{\partial \bar{h}_2}\right.\right. \\
&= \frac{1}{3}\left[\left(\frac{\partial l_t}{\partial \mu_t} \times \frac{\partial \mu_t}{\partial h_t} \times \frac{\partial h_t}{\partial \bar{h}_t} \times x_0\right) + \left(\frac{\partial l_1}{\partial \mu_1} \times \frac{\partial \mu_1}{\partial h_1} \times \frac{\partial h_1}{\partial \bar{h}_1} \times \left(W\left(\frac{\partial h_0}{\partial \bar{h}_0} \times \frac{\partial \bar{h}_0}{\partial U}\right) + x_1\right)\right) + \left(\right.\right. \\
&= \frac{1}{3}\left[\left(\frac{\partial l_t}{\partial \mu_t} \times \frac{\partial \mu_t}{\partial h_t} \times \frac{\partial h_t}{\partial \bar{h}_t} \times x_0\right) + \left(\frac{\partial l_1}{\partial \mu_1} \times \frac{\partial \mu_1}{\partial h_1} \times \frac{\partial h_1}{\partial \bar{h}_1} \times \left(W\left(\frac{\partial h_0}{\partial \bar{h}_0} \times x_0\right) + x_1\right)\right) + \left(\frac{\partial}{\partial}\right.\right. \\
&= \frac{1}{3}\left[\left(\frac{\partial l_t}{\partial \mu_t} \times \frac{\partial \mu_t}{\partial h_t} \times \frac{\partial h_t}{\partial \bar{h}_t} \times x_0\right) + \left(\frac{\partial l_1}{\partial \mu_1} \times \frac{\partial \mu_1}{\partial h_1} \times \frac{\partial h_1}{\partial \bar{h}_1} \times \left(W\left(\frac{\partial h_0}{\partial \bar{h}_0} \times x_0\right) + x_1\right)\right) + \left(\frac{\partial}{\partial}\right.\right.
\end{aligned}
$$