HW 10 | Question 2

In the lecture the forward process of LSTM was outlined. GRU is a “simpler version” of LSTM. In this problem you’ll practice stepping through the forward process of GRU. Calculate value of the output state and each gate for t1 and t2 as was done for LSTM in lecture. (calculation should be rounded to four decimals)

$$z = sigmoid(x_tU^z + s_(t-1)W^z)$$

$$r = sigmoid(x_tU^r + s_(t-1)W^r)$$

$$h = tanh(x_tU^h + (s_(t-1) \circ r)W^h)$$

$$s_t = (1-z) \circ h + z \circ s_(t-1)$$

$$U^z = \begin{vmatrix} 0.35 \\ 0.6 \end{vmatrix}$$

$$U^r = \begin{vmatrix} 0.45 \\ 0.12 \end{vmatrix}$$

$$U^h = \begin{vmatrix} 0.56 & 0.1 & 0.3 \\ 0.2 & 0.5 & 0.21 \end{vmatrix}$$

$$W^z = \begin{vmatrix} 0.12 \\ 0.67 \\ 0.34 \end{vmatrix}$$

$$W^r = \begin{vmatrix} 0.21 \\ 0.22 \\ 0.45 \end{vmatrix}$$

$$W^h = \begin{vmatrix} 0.54 & 0.1 & 0.2 \\ 0.2 & 0.66 & 0.3 \\ 0.9 & 0.87 & 0.3 \end{vmatrix}$$

In [ ]:

```
import numpy as np

# creating the sigmoid and tanhn functions

def sigmoid(x):
    return 1/(1 + np.exp(-x))

def tanh(x):
    return np.tanh(x)
```

In [ ]:

```
# creating arrays of the given data

# t1 and t2
x_t1 = np.array([0.23, 0.12])
x_t2 = np.array([0.48, 0.98])

# weights matrices
uz = np.array([[0.35], [0.6]])
ur = np.array([[0.45], [0.12]])
uh = np.array([[0.56, 0.1, 0.3], [0.2, 0.5, 0.21]])

wz = np.array([[0.12], [0.67], [0.34]])
wr = np.array([[0.21], [0.22], [0.45]])
wh = np.array([[0.54, 0.1, 0.2], [0.2, 0.66, 0.3], [0.9, 0.87, 0.3]])

s_t0 = np.array([[0, 0, 0]])
```

In [ ]:

```
# Calculating the values for t1
z_t1 = sigmoid(x_t1 @ uz + s_t0 @ wz)
r_t1 = sigmoid(x_t1 @ ur + s_t0 @ wr)
h_t1 = tanh(x_t1 @ uh + (s_t0 * r_t1) @ wh)
s_t1 = (1 - z_t1) * h_t1 + z_t1 * s_t0

# Calculating the values for t2 using the state from t1
z_t2 = sigmoid(x_t2 @ uz + s_t1 @ wz)
r_t2 = sigmoid(x_t2 @ ur + s_t1 @ wr)
```

```
h_t2 = tanh(x_t2 @ uh + (s_t1 * r_t2) @ wh)
s_t2 = (1 - z_t2) * h_t2 + z_t2 * s_t1

# Rounding the results to four decimal places as requested
output_t1_t2 = {
    "t1": {
        "z": np.round(z_t1, 4),
        "r": np.round(r_t1, 4),
        "h": np.round(h_t1, 4),
        "s_t": np.round(s_t1, 4)
    },
    "t2": {
        "z": np.round(z_t2, 4),
        "r": np.round(r_t2, 4),
        "h": np.round(h_t2, 4),
        "s_t": np.round(s_t2, 4)
    }
}
```

```
In [ ]: output_t1_t2['t1']
```

```
Out[ ]: {'z': array([[0.5381]]),
         'r': array([[0.5294]]),
         'h': array([[0.1516, 0.0828, 0.0939]]),
         's_t': array([[0.07  , 0.0383, 0.0434]])}
```

```
In [ ]: output_t1_t2['t2']
```

```
Out[ ]: {'z': array([[0.691]]),
         'r': array([[0.593]]),
         'h': array([[0.4738, 0.5223, 0.3563]]),
         's_t': array([[0.1948, 0.1878, 0.1401]])}
```