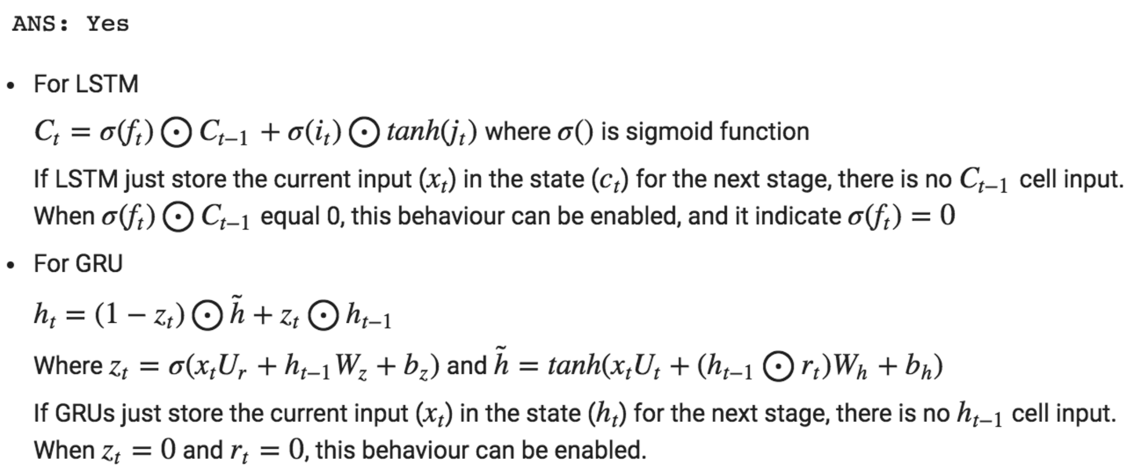
Deep Learning Coursework 2

WU Zhi 17040772
UCABWUA@UCL.AC.UK
Department of Computer Science
University College London

Q1: Understanding LSTM vs GRU

1. Can LSTMs (and respectively GRUs) just store the current input in the state (c_t for LSTM and h_t for GRU, in the class notation) for the next step? If so, give the gates activation that would enable this behaviour. If not, explain why not. [10 pts]



```
ANS: Yes
```

- For LSTM

  $C_t = \sigma(f_t) \odot C_{t-1} + \sigma(i_t) \odot tanh(j_t)$ where $\sigma()$ is sigmoid function

  If LSTM just store the current input (x_t) in the state (c_t) for the next stage, there is no C_{t-1} cell input. When $\sigma(f_t) \odot C_{t-1}$ equal 0, this behaviour can be enabled, and it indicate $\sigma(f_t) = 0$

- For GRU

  $h_t = (1 - z_t) \odot \tilde{h} + z_t \odot h_{t-1}$

  Where $z_t = \sigma(x_t U_r + h_{t-1} W_z + b_z)$ and $\tilde{h} = tanh(x_t U_t + (h_{t-1} \odot r_t) W_h + b_h)$

  If GRUs just store the current input (x_t) in the state (h_t) for the next stage, there is no h_{t-1} cell input. When $z_t = 0$ and $r_t = 0$, this behaviour can be enabled.

2. Can LSTMs (and respectively GRUs) just store a previous state into the current state and ignore the current input? If so, give the gates' activation that would enable this. If not, explain why not. [10 pts]

`ANS: Yes`

- For LSTM

  Before generating new memories, we need to determine whether the new words we currently see are important. The input gate determines whether or not it participates in generating a new memory by determining whether the input value is worth keeping according to the input and past hidden layer states.

  For this question, when LSTMs and GRUs just store a previous state into the current state and ignore the current input, it means the input gate is 0.

  $C_t = \sigma(f_t) \odot C_{t-1} + \sigma(i_t) \odot tanh(j_t)$ where $\sigma()$ is sigmoid function

  when input gata equal 0, $\sigma(i_t) \odot tanh(j_t) = 0$, and it means C_t ingore the input value in the case of the input information is not important.

- For GRU

  $h_t = (1 - z_t) \odot \tilde{h} + z_t \odot h_{t-1}$

  Where $z_t = \sigma(x_t U_r + h_{t-1} W_z + b_z)$ and $\tilde{h} = tanh(x_t U_t + (h_{t-1} \odot r_t) W_h + b_h)$

  If GRUs just store the previous state (h_{t-1}) in the state (h_t) for the next stage, it means C_t ingore the input value in the case of the input information is not important. When $(1 - z_t) \odot \tilde{h} = 0$ and $z_t = 1$, this behaviour can be enabled.

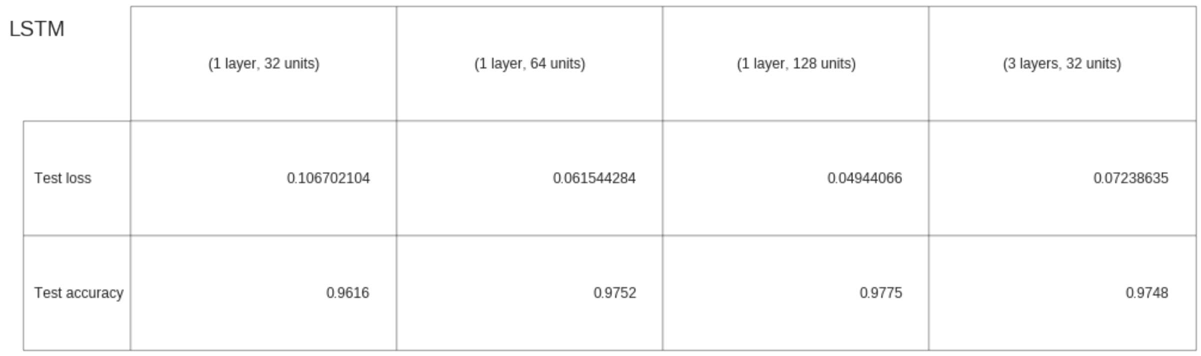
3. Are GRUs a special case of LSTMs? If so, give the expression of the GRU gates in term of LSTM's gates (o_t,i_t,f_t). If not, give a counter-example. Assume here the same input. [10 pts]

- ANS: NO

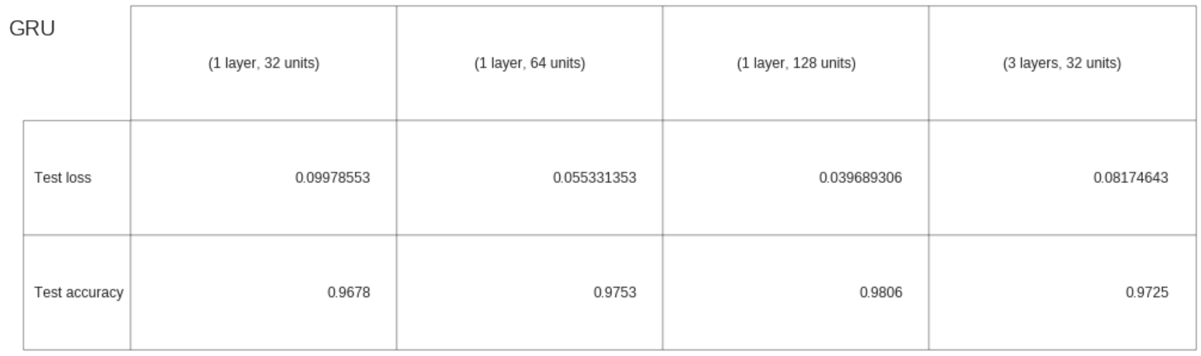
  LSTMs use i_t,f_t and o_t to impress input gata, forget gata and output gata respectively, whereas GRUs use z_t and r_t to impress updata and reset gatas. We can show that the the LSTM output with gates set to $f_t = z_t$ (z_t is from the GRU), $i_t = 1 - f_t$, open output-gate $o_t = 1$, is equal to the GRU output with open-reset gate $r_t = 1$, only if the output-activation function of the LSTM is replaced by a linear function. As we need to make additional assumptions on the gates of the GRU, it is not a special case of the LSTM.

Q2: Implementation. Line-by-Line MNIST Classification

Result



LSTM	(1 layer, 32 units)	(1 layer, 64 units)	(1 layer, 128 units)	(3 layers, 32 units)
Test loss	0.106702104	0.061544284	0.04944066	0.07238635
Test accuracy	0.9616	0.9752	0.9775	0.9748



GRU	(1 layer, 32 units)	(1 layer, 64 units)	(1 layer, 128 units)	(3 layers, 32 units)
Test loss	0.09978553	0.055331353	0.039689306	0.08174643
Test accuracy	0.9678	0.9753	0.9806	0.9725

Q3: Analyse the results

1. How does this compare with the results you obtained in the first assignment(DL1), when training a model that "sees" the entire image at once? Explain differences. [5 pts]

   ANS:

   Comparing LSTMs and GRUs trained model with NN and CNN trained model, we can find that the test accuracy of LSTMs and GRUs trained model are higher than NN model but lower than CNN model.By comparing RNN with NN, RNN process is to add a hidden state at each step. This state is used to express all memory, and this memory will directly affect the "output" of this step, so RNN trained model result is higher than NN model. By comparing RNN with CNN, CNN work better than RNN because CNN will learn to recognize components of an image and then learn to combine these components to recognize larger structures. But in case of RNN, it will similarly learn to recognize patterns across time.

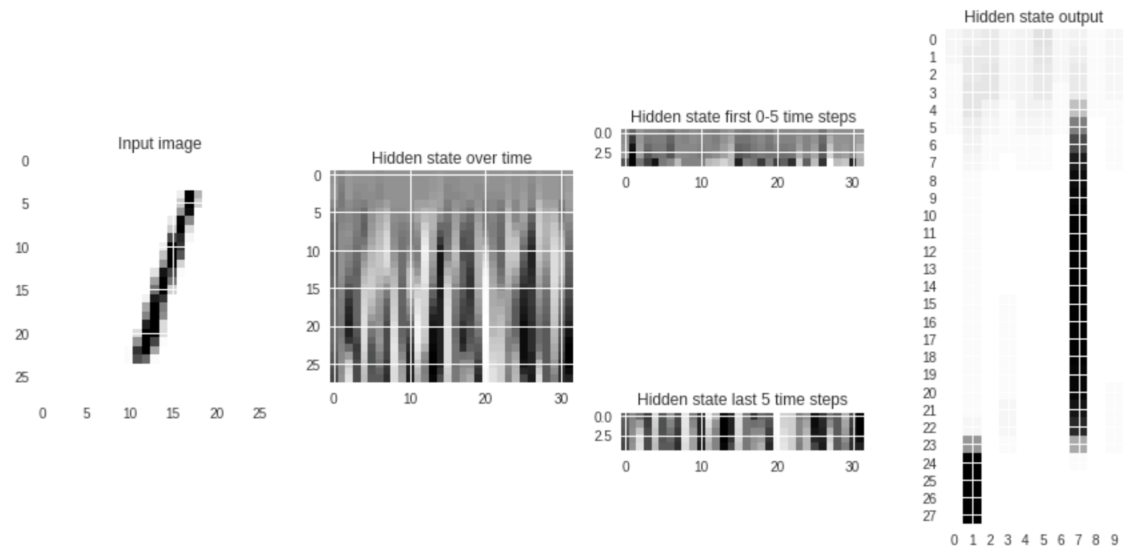
2. Let us take a look closer look at one of the trained models: say GRU (32). Plot the outputs of the RNN layer and hidden state over time. In particular, look at the first 3-5 time steps. Plot the input image along side. You can use python plt.imshow(output_GRU_over_time) for these, where output_GRU_over_time.shape is (T=28,hidden_units) dimensional. What do you observe? Show at least one pair of these plots to support your observation(s). [5 pts]

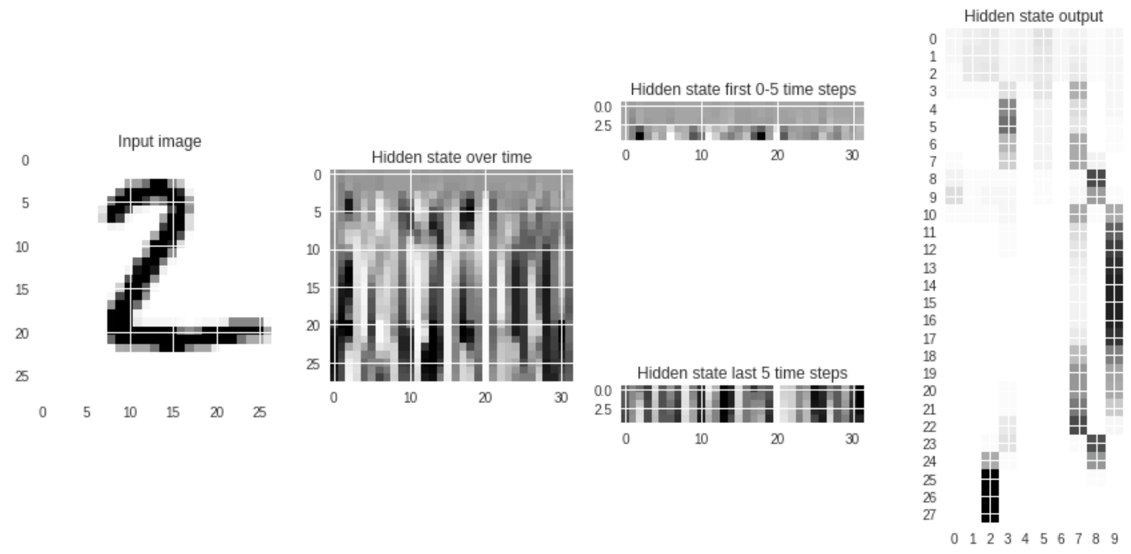
```
ANS:

The figures below (the cell output below) shows GRU (32) outputs of the RNN layer and
hidden state over time and input image, and two pair of figures are plotted. From fitst
0-5 time steps, it can be found that the first pair figures(input number '1') 32 hidden
units value are similar at the begining, but the difference become bigger in the fifth
step. The second pair figures(input number '2') 32 hidden units value are similar at the
begining as well, but the difference become bigger in the third step. The time when 32
hidden units value become bigger is decided by the input image. Because the first input
image(number '1') handwriting begin at row 5, whereas the second input image(number '2')
handwriting begin at row 3.
```

3. Now, look at the last 3-5 time steps. What do you observe? When is the classification decision made? To validate your answer to the second part of the question, provide the classification predictions for the last 5 time steps -- that is, pretend whatever the output of the GRU is at that time step is in fact the last output in the computation, and feed that into the classification mapping. [10 pts]

```
ANS:

The figures below (the cell output below) shows GRU (32) outputs of the RNN layer and
hidden state over time and input image, and two pair of figures are plotted.

From the last 5 step, it can be found that both two images made decision before inputting
the last row image pixel. For the first pair of figure (input number '1'), the
classification decision made at step 23, and the second pair of figures (input number
'2')  made decision at step 25.

The classification predictions are plotted as well in the right of two pair figures. From
the classification predictions figure, we can find the first image made disicion at step
24, and the second made decision at step 25. The answer in hidden state value prediction
can be validated.
```

Q3 plot


Input image
Hidden state over time
Hidden state first 0-5 time steps
Hidden state last 5 time steps
Hidden state output


Input image
Hidden state over time
Hidden state first 0-5 time steps
Hidden state last 5 time steps
Hidden state output