TRINITY COLLEGE DUBLIN

School of Computer Science and Statistics
Week 8 Assignment CS7CS4/CSU44061 Machine Learning

Min Zhang

Answers to question 1:

(i)

(a) :

Original data：

```
Data[[1,  2,  0,  1],
     [1,  1,  3,  2],
     [0,  2,  2,  1],
     [1,  2,  3,  4]]
```

kernel：

```
kernel1[[-1,  -1,  -1],
        [-1,  8,  -1],
        [-1,  -1,  -1]]
```

result：

```
result1 [[-3.   13.]
         [ 3.   -2.]]
```

Original data：

```
Data[[1,  2,  0,  1],
     [1,  1,  3,  2],
     [0,  2,  2,  1],
     [1,  2,  3,  4]]
```

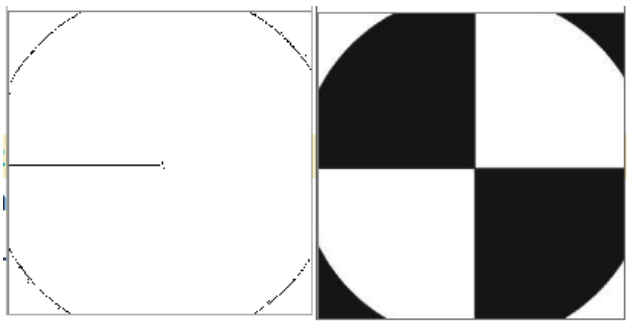
kernel：

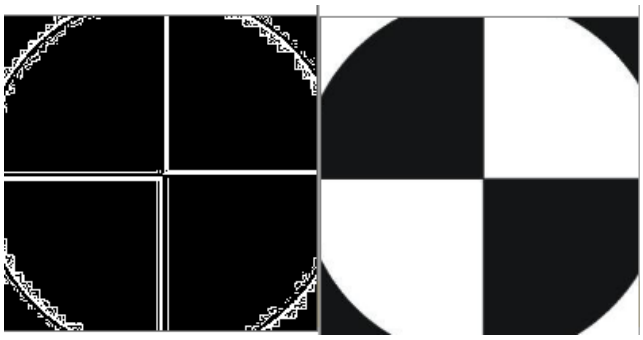
```
kernel2[[0,  -1,  0],
        [-1,  8,  -1],
        [0,  -1,  0]]
```

result：

```
result2 [[ 0.   19.]
         [11.   7.]]
```

(b) :





Answers to question 2:

Environment: tensorflow2.0-gpu.

a.

architecture of the ConvNet:

layer name	kernel size	output channels
conv2d	(3,3)	16
conv2d_1	(3,3)	16
conv2d_2	(3,3)	32
conv2d_3	(3,3)	32
flatten	-	-
dense	-	-

b.

(i)

Total params: 37,146

Trainable params: 37,146

Non-trainable params: 0

The dense layer has the most parameters, because the dense layer is fully mapped, while the convolution layer shares parameters.

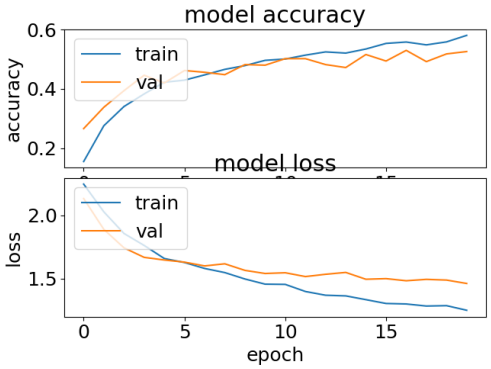
The accuracy on the test data is lower than that on the training data.

Results of training data:

accuracy			0.63	4999
macro avg	0.64	0.63	0.63	4999
weighted avg	0.64	0.63	0.63	4999

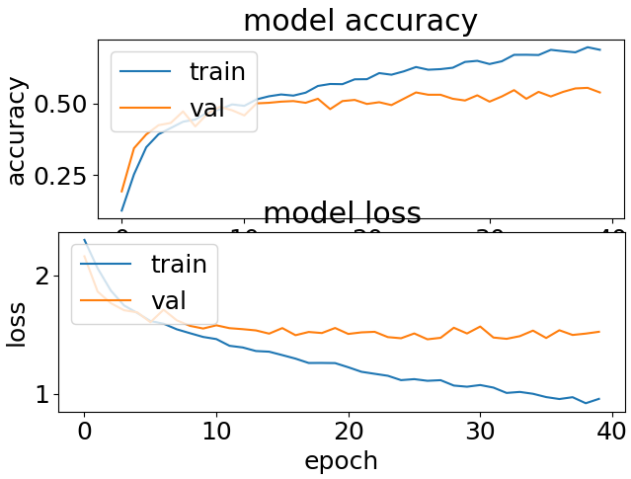
Results of test data:

accuracy			0.49	10000
macro avg	0.49	0.49	0.48	10000
weighted avg	0.49	0.49	0.48	10000



(ii)

There may be some overfitting in the training process. The training accuracy is much higher than the test accuracy. And if we set epochs to 40, I can find obvious overfitting from the loss line of test data as well.



(iii)

Dataset	training accuracy	test accuracy	training time/per epoch
5K	63	49	0s,70µs
10K	65	55	1s,69µs
20K	69	62	1s,70µs
40K	73	67	2s,68µs

As the training data increases, the prediction accuracy and training time will **increase.**

(iv)

L1 value	training accuracy	test accuracy
0	62	49
0.1	36	35
0.01	46	42
0.001	53	45
0.0001	62	48
0.00001	64	49

The larger the value of L1, the lower the accuracy on both dataset.

c.

(i)

Modified code:

```
model.add(Conv2D(16, (3, 3), padding='same',
                 input_shape=x_train.shape[1:], activation='relu'))
model.add(MaxPooling2D(2))
model.add(Conv2D(32, (3, 3), padding='same', activation='relu'))
model.add(MaxPooling2D(2))
```

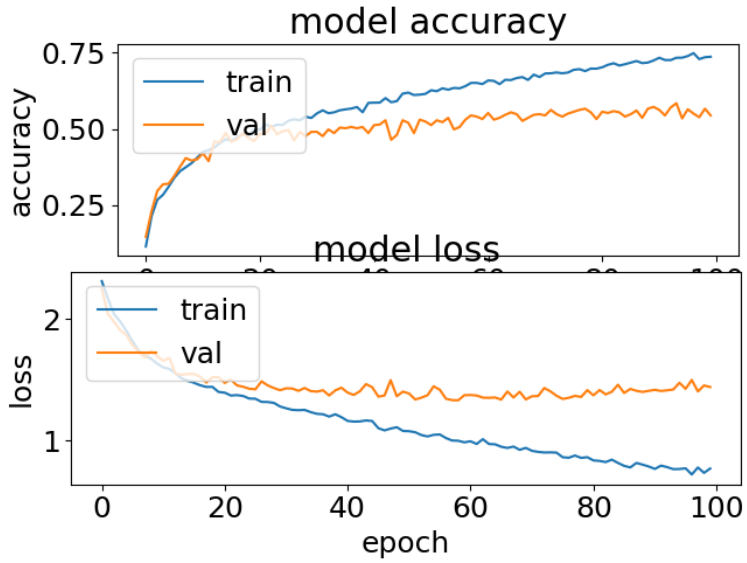
(ii)

Parameters	training accuracy	test accuracy	training time/per epoch
25578	55	49	0s,47µs

It cost 47µs to train the network which is faster than the original network. The reason for this is that max pooling layers can reduce feature size while maintaining the effective information.

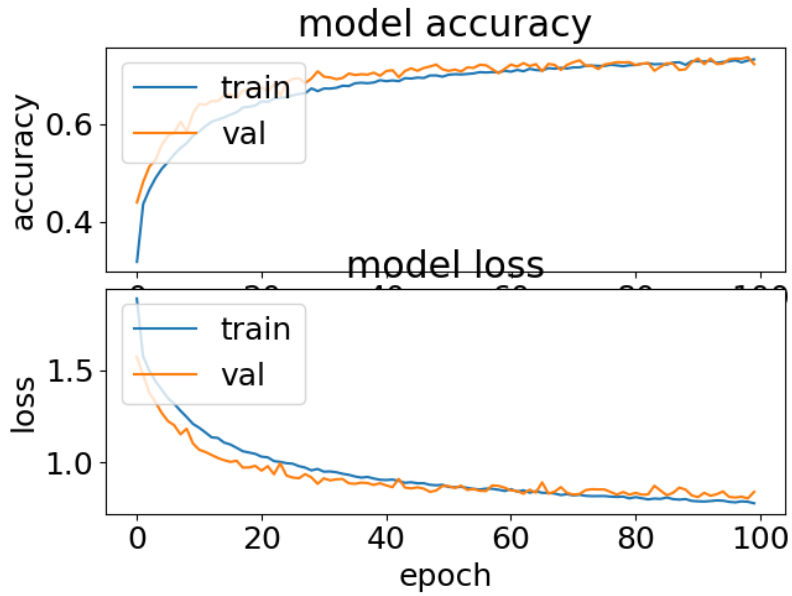
d.

I train the 5K data point by using this thinner and deeper network for 100 epochs.



It takes about 60μs per epoch. The train accuracy and test accuracy are 0.7364 and 0.5440 respectively. The network has less parameters (23,314) than the original one. The thinner and deeper network uses less parameters to get the similar performance as the original one by sacrificing more training time. And it seems that this kind of network has an effectiveness on preventing overfitting.

I retrain the network on the full dataset and its effectiveness is shown below:



The train accuracy and test accuracy are 0.7323 and 0.7224 respectively. It takes about 3s and 58us per epochs. Compared with the result above, it shows that the diversity of samples can improve the generalization ability of the network. Rich samples can prevent overfitting but also take more training time.