Ranathunga R.A.C.D. - 190501V

Excercise 11

Github link = https://github.com/ChamithDilshan

Question 01

In [ ]:
```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import datasets, layers, models
import numpy as np
import matplotlib.pyplot as plt
```

In [ ]:
```
mnist = keras.datasets.mnist
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()

paddings = tf.constant([[0, 0], [2, 2], [2, 2]])
train_images = tf.pad(train_images, paddings, constant_values=0)
test_images = tf.pad(test_images, paddings, constant_values=0)

print('train_images.shape: ', train_images.shape)
print('train_labels.shape: ', train_labels.shape)
print('test_images.shape:', test_images.shape)
print('test_labels.shape:', test_labels.shape)

class_names = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9']
train_images = tf.dtypes.cast(train_images, tf.float32)
test_images = tf.dtypes.cast(test_images, tf.float32)
train_images, test_images = train_images[..., np.newaxis]/255.0, test_images[..., np.newaxis]/255.0
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11493376/11490434 [==============================] - 0s 0us/step
11501568/11490434 [==============================] - 0s 0us/step
train_images.shape:  (60000, 32, 32)
train_labels.shape:  (60000,)
test_images.shape: (10000, 32, 32)
test_labels.shape: (10000,)
```

In [ ]:
```
model = models.Sequential()
model.add(layers.Conv2D(6,(5,5),activation = 'relu',input_shape = (32,32,1)))
model.add(layers.AveragePooling2D((2,2)))
model.add(layers.Conv2D(16,(5,5),activation = 'relu'))
model.add(layers.AveragePooling2D((2,2)))
model.add(layers.Flatten())
model.add(layers.Dense(120,activation = 'relu'))
model.add(layers.Dense(84,activation = 'relu'))
model.add(layers.Dense(10))
model.compile(optimizer = 'adam',loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),metrics = ['accuracy'])
print(model.summary())
model.fit(train_images,train_labels,epochs = 5)
test_loss, test_accuracy = model.evaluate(test_images,test_labels,verbose = 2)
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 28, 28, 6)         156

 average_pooling2d (AverageP  (None, 14, 14, 6)        0
 ooling2D)

 conv2d_1 (Conv2D)           (None, 10, 10, 16)        2416

 average_pooling2d_1 (Averag  (None, 5, 5, 16)         0
 ePooling2D)

 flatten (Flatten)           (None, 400)               0

 dense (Dense)               (None, 120)               48120

 dense_1 (Dense)             (None, 84)                10164

 dense_2 (Dense)             (None, 10)                850

=================================================================
Total params: 61,706
Trainable params: 61,706
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/5
1875/1875 [==============================] - 32s 17ms/step - loss: 0.2222 - accuracy: 0.9322
Epoch 2/5
1875/1875 [==============================] - 33s 18ms/step - loss: 0.0738 - accuracy: 0.9772
Epoch 3/5
1875/1875 [==============================] - 34s 18ms/step - loss: 0.0521 - accuracy: 0.9833
Epoch 4/5
1875/1875 [==============================] - 34s 18ms/step - loss: 0.0406 - accuracy: 0.9873
Epoch 5/5
1875/1875 [==============================] - 34s 18ms/step - loss: 0.0339 - accuracy: 0.9892
313/313 - 2s - loss: 0.0356 - accuracy: 0.9871 - 2s/epoch - 7ms/step
```

Question 02

In [ ]:
```python
(train_images, train_labels), (test_images, test_labels) = datasets.cifar10.load_data()

train_images, test_images = train_images / 255.0, test_images / 255.0
class_names = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']
print('train_images.shape: ', train_images.shape)
print('train_labels.shape: ', train_labels.shape)
print('test_images.shape:', test_images.shape)
print('test_labels.shape:', test_labels.shape)
```

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170500096/170498071 [==============================] - 2s 0us/step
170508288/170498071 [==============================] - 2s 0us/step
train_images.shape:  (50000, 32, 32, 3)
train_labels.shape:  (50000, 1)
test_images.shape: (10000, 32, 32, 3)
test_labels.shape: (10000, 1)
```

In [ ]:
```python
model = models.Sequential()
model.add(layers.Conv2D(32,(5,5),activation = 'relu',input_shape = (32,32,3)))
model.add(layers.MaxPool2D((2,2)))
model.add(layers.Conv2D(64,(3,3),activation = 'relu'))
model.add(layers.MaxPool2D((2,2)))
model.add(layers.Conv2D(128,(3,3),activation = 'relu'))
model.add(layers.MaxPool2D((2,2)))
model.add(layers.Flatten())
model.add(layers.Dense(64,activation = 'relu'))
model.add(layers.Dense(10))
```

In [ ]:
```python
model.compile(optimizer = keras.optimizers.Adam(learning_rate=0.001),loss = tf.keras.losses.SparseCategoricalCrossentropy(from_l
print(model.summary)
model.fit(train_images,train_labels,epochs = 5)
test_loss, test_accuracy = model.evaluate(test_images,test_labels,verbose = 2)
print(test_accuracy)
```

```
<bound method Model.summary of <keras.engine.sequential.Sequential object at 0x7fc3e0efc290>>
Epoch 1/5
1563/1563 [==============================] - 81s 52ms/step - loss: 1.5622 - accuracy: 0.4282
Epoch 2/5
1563/1563 [==============================] - 76s 49ms/step - loss: 1.1783 - accuracy: 0.5821
Epoch 3/5
1563/1563 [==============================] - 76s 49ms/step - loss: 1.0172 - accuracy: 0.6428
Epoch 4/5
1563/1563 [==============================] - 76s 49ms/step - loss: 0.9116 - accuracy: 0.6829
Epoch 5/5
1563/1563 [==============================] - 78s 50ms/step - loss: 0.8325 - accuracy: 0.7082
313/313 - 4s - loss: 0.9351 - accuracy: 0.6805 - 4s/epoch - 12ms/step
0.6804999709129333
```

Question 3

In [ ]:
```
mnist = keras.datasets.mnist
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()

paddings = tf.constant([[0, 0], [2, 2], [2, 2]])
train_images = tf.pad(train_images, paddings, constant_values=0)
test_images = tf.pad(test_images, paddings, constant_values=0)

print('train_images.shape: ', train_images.shape)
print('train_labels.shape: ', train_labels.shape)
print('test_images.shape:', test_images.shape)
print('test_labels.shape:', test_labels.shape)

class_names = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9']
train_images = tf.dtypes.cast(train_images, tf.float32)
test_images = tf.dtypes.cast(test_images, tf.float32)
train_images, test_images = train_images[..., np.newaxis]/255.0, test_images[..., np.newaxis]/255.0

model_base = models.Sequential()
model_base.add(layers.Conv2D(32,(3,3),activation = 'relu',input_shape = (32,32,1)))
model_base.add(layers.MaxPool2D((2,2)))
model_base.add(layers.Conv2D(64,(3,3),activation = 'relu'))
model_base.add(layers.MaxPool2D((2,2)))
model_base.add(layers.Conv2D(64,(3,3),activation = 'relu'))
model_base.add(layers.Flatten())
model_base.add(layers.Dense(64,activation = 'relu'))
model_base.add(layers.Dense(10))

model_base.compile(optimizer = keras.optimizers.Adam(),loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True), me
print(model_base.summary())
model_base.fit(train_images,train_labels,epochs = 2)
test_loss, test_accuracy = model_base.evaluate(test_images,test_labels,verbose = 2)
model_base.save_weights('saved_weights/')
```

```
train_images.shape:  (60000, 32, 32)
train_labels.shape:  (60000,)
test_images.shape: (10000, 32, 32)
test_labels.shape: (10000,)
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_5 (Conv2D)           (None, 30, 30, 32)        320

 max_pooling2d_3 (MaxPooling  (None, 15, 15, 32)       0
 2D)

 conv2d_6 (Conv2D)           (None, 13, 13, 64)        18496

 max_pooling2d_4 (MaxPooling  (None, 6, 6, 64)         0
 2D)

 conv2d_7 (Conv2D)           (None, 4, 4, 64)          36928

 flatten_2 (Flatten)         (None, 1024)              0

 dense_5 (Dense)             (None, 64)                65600

 dense_6 (Dense)             (None, 10)                650

=================================================================
Total params: 121,994
Trainable params: 121,994
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/2
1875/1875 [==============================] - 65s 35ms/step - loss: 0.1333 - accuracy: 0.9585
Epoch 2/2
1875/1875 [==============================] - 66s 35ms/step - loss: 0.0415 - accuracy: 0.9873
313/313 - 3s - loss: 0.0374 - accuracy: 0.9880 - 3s/epoch - 9ms/step
```

Question 4

In [ ]:
```
model_lw = models.Sequential()
model_lw.add(layers.Conv2D(32,(3,3),activation = 'relu',input_shape = (32,32,1)))
model_lw.add(layers.MaxPool2D((2,2)))
model_lw.add(layers.Conv2D(64,(3,3),activation = 'relu'))
model_lw.add(layers.MaxPool2D((2,2)))
model_lw.add(layers.Conv2D(64,(3,3),activation = 'relu'))
model_lw.add(layers.Flatten())
model_lw.add(layers.Dense(64,activation = 'relu'))
model_lw.add(layers.Dense(10))

model_lw.compile(optimizer = keras.optimizers.Adam(),loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),metr

print(model_lw.summary())

model_lw.load_weights('saved_weights/')
model_lw.fit(train_images,train_labels,epochs = 2)
test_loss, test_accuracy = model_lw.evaluate(test_images,test_labels,verbose = 2)
model_lw.save('saved_model/')
```

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_8 (Conv2D)           (None, 30, 30, 32)        320

 max_pooling2d_5 (MaxPooling  (None, 15, 15, 32)       0
 2D)

 conv2d_9 (Conv2D)           (None, 13, 13, 64)        18496

 max_pooling2d_6 (MaxPooling  (None, 6, 6, 64)         0
 2D)

 conv2d_10 (Conv2D)          (None, 4, 4, 64)          36928

 flatten_3 (Flatten)         (None, 1024)              0

 dense_7 (Dense)             (None, 64)                65600

 dense_8 (Dense)             (None, 10)                650

=================================================================
Total params: 121,994
Trainable params: 121,994
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/2
1875/1875 [==============================] - 65s 35ms/step - loss: 0.0282 - accuracy: 0.9907
Epoch 2/2
1875/1875 [==============================] - 65s 35ms/step - loss: 0.0220 - accuracy: 0.9933
313/313 - 3s - loss: 0.0303 - accuracy: 0.9906 - 3s/epoch - 9ms/step
INFO:tensorflow:Assets written to: saved_model/assets
```

Question 5 - load the model

In [ ]:
```
model_ld = keras.models.load_model('saved_model/')
print(model_ld.summary())
model_ld.evaluate(test_images, test_labels, verbose = 2)
```

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_8 (Conv2D)           (None, 30, 30, 32)        320

 max_pooling2d_5 (MaxPooling  (None, 15, 15, 32)       0
 2D)

 conv2d_9 (Conv2D)           (None, 13, 13, 64)        18496

 max_pooling2d_6 (MaxPooling  (None, 6, 6, 64)         0
 2D)

 conv2d_10 (Conv2D)          (None, 4, 4, 64)          36928

 flatten_3 (Flatten)         (None, 1024)              0

 dense_7 (Dense)             (None, 64)                65600

 dense_8 (Dense)             (None, 10)                650

=================================================================
Total params: 121,994
Trainable params: 121,994
Non-trainable params: 0
_________________________________________________________________
None
313/313 - 4s - loss: 0.0303 - accuracy: 0.9906 - 4s/epoch - 11ms/step
```

Out[ ]:
```
[0.030343422666192055, 0.9905999898910522]
```

Question 6

In [ ]:
```
# fine tune the model
base_inputs = model_ld.layers[0].input
base_outputs = model_ld.layers[-2].output
output = layers.Dense(10)(base_outputs)

new_model = keras.Model(inputs = base_inputs, outputs = output)
new_model.compile(optimizer = keras.optimizers.Adam(), loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),me
print(new_model.summary())

new_model.fit(train_images, train_labels, epochs=3)
test_loss, test_accuracy = new_model.evaluate(test_images, test_labels, verbose = 2)
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_8_input (InputLayer)  [(None, 32, 32, 1)]      0

 conv2d_8 (Conv2D)           (None, 30, 30, 32)        320

 max_pooling2d_5 (MaxPooling  (None, 15, 15, 32)       0
 2D)

 conv2d_9 (Conv2D)           (None, 13, 13, 64)        18496

 max_pooling2d_6 (MaxPooling  (None, 6, 6, 64)         0
 2D)

 conv2d_10 (Conv2D)          (None, 4, 4, 64)          36928

 flatten_3 (Flatten)         (None, 1024)              0

 dense_7 (Dense)             (None, 64)                65600

 dense_9 (Dense)             (None, 10)                650

=================================================================
Total params: 121,994
Trainable params: 121,994
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/3
1875/1875 [==============================] - 72s 38ms/step - loss: 0.0708 - accuracy: 0.9799
Epoch 2/3
1875/1875 [==============================] - 72s 39ms/step - loss: 0.0173 - accuracy: 0.9948
Epoch 3/3
1875/1875 [==============================] - 71s 38ms/step - loss: 0.0142 - accuracy: 0.9956
313/313 - 4s - loss: 0.0259 - accuracy: 0.9932 - 4s/epoch - 14ms/step
```

Question 7

In [ ]:
```
model_for_tl = keras.models.load_model('saved_model/')
model_for_tl.trainable = False

for layer in model_for_tl.layers:
    assert layer.trainable == False

base_inputs = model_for_tl.layers[0].input
base_outputs = model_for_tl.layers[-2].output
output = layers.Dense(10)(base_outputs)

new_model = keras.Model(inputs = base_inputs, outputs = output)
new_model.compile(optimizer = keras.optimizers.Adam(), loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),me
new_model.fit(train_images, train_labels, epochs=3,verbose = 2)
new_model.evaluate(test_images, test_labels, verbose = 2)
```

```
Epoch 1/3
1875/1875 - 24s - loss: 0.1629 - accuracy: 0.9626 - 24s/epoch - 13ms/step
Epoch 2/3
1875/1875 - 18s - loss: 0.0156 - accuracy: 0.9957 - 18s/epoch - 10ms/step
Epoch 3/3
1875/1875 - 20s - loss: 0.0115 - accuracy: 0.9967 - 20s/epoch - 11ms/step
313/313 - 3s - loss: 0.0218 - accuracy: 0.9939 - 3s/epoch - 11ms/step
```

Out[ ]:
```
[0.021838992834091187, 0.9939000010490417]
```

Question 8

In [ ]:
```
model = keras.applications.resnet_v2.ResNet50V2(include_top= True)
model.trainable = False

for layer in model.layers:
    assert layer.trainable == False

base_inputs = model.layers[0].input
base_outputs = model.layers[-2].output
output = layers.Dense(5)(base_outputs)

images = tf.random.normal(shape=(5,224,224,3))
labels = tf.constant(list(np.random.randint(0,4,5)))

new_model = keras.Model(inputs = base_inputs, outputs = output)
new_model.compile(optimizer = keras.optimizers.Adam(), loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),me
new_model.fit(images, labels, epochs=15,verbose = 2)
```

```
Epoch 1/15
1/1 - 4s - loss: 3.1782 - accuracy: 0.0000e+00 - 4s/epoch - 4s/step
Epoch 2/15
1/1 - 1s - loss: 2.7754 - accuracy: 0.0000e+00 - 579ms/epoch - 579ms/step
Epoch 3/15
1/1 - 1s - loss: 2.4127 - accuracy: 0.0000e+00 - 573ms/epoch - 573ms/step
Epoch 4/15
1/1 - 1s - loss: 2.0989 - accuracy: 0.0000e+00 - 584ms/epoch - 584ms/step
Epoch 5/15
1/1 - 1s - loss: 1.8355 - accuracy: 0.2000 - 582ms/epoch - 582ms/step
Epoch 6/15
1/1 - 1s - loss: 1.6166 - accuracy: 0.2000 - 577ms/epoch - 577ms/step
Epoch 7/15
1/1 - 1s - loss: 1.4369 - accuracy: 0.6000 - 579ms/epoch - 579ms/step
Epoch 8/15
1/1 - 1s - loss: 1.2944 - accuracy: 0.4000 - 574ms/epoch - 574ms/step
Epoch 9/15
1/1 - 1s - loss: 1.1873 - accuracy: 0.4000 - 587ms/epoch - 587ms/step
Epoch 10/15
1/1 - 1s - loss: 1.1115 - accuracy: 0.4000 - 568ms/epoch - 568ms/step
Epoch 11/15
1/1 - 1s - loss: 1.0606 - accuracy: 0.4000 - 587ms/epoch - 587ms/step
Epoch 12/15
1/1 - 1s - loss: 1.0268 - accuracy: 0.4000 - 587ms/epoch - 587ms/step
Epoch 13/15
1/1 - 1s - loss: 1.0032 - accuracy: 0.6000 - 571ms/epoch - 571ms/step
Epoch 14/15
1/1 - 1s - loss: 0.9848 - accuracy: 0.8000 - 577ms/epoch - 577ms/step
Epoch 15/15
1/1 - 1s - loss: 0.9682 - accuracy: 0.8000 - 579ms/epoch - 579ms/step
```

Out[ ]:
```
<keras.callbacks.History at 0x7fc3d07ebad0>
```