Index Number: 190337X

Name: C. J. Kurukulasuriya

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
```

Q1) LeNet5 network for MNIST

In [ ]:

```
mnist = keras.datasets.mnist
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()

print("Before padding:")
print('train_images.shape: ', train_images.shape)
print('test_images.shape:', test_images.shape)
print()

# Padding
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("After padding:")
print('train_images.shape: ', train_images.shape)
print('test_images.shape:', test_images.shape)
print('train_labels.shape: ', train_labels.shape)
print('test_labels.shape:', test_labels.shape)
class_names = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9']

#print(train_images.dtype)

#change dtype
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
#print(train_images.dtype)
```

```
Before padding:
train_images.shape:  (60000, 28, 28)
test_images.shape: (10000, 28, 28)

After padding:
train_images.shape:  (60000, 32, 32)
test_images.shape: (10000, 32, 32)
train_labels.shape:  (60000,)
test_labels.shape: (10000,)
```

In [ ]:

```
# LeNet5 for MNIST (as discussed in class)
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_lost, test_accuracy = model.evaluate(test_images, test_labels, verbose=2)
```

```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
```

```
=================================================================
 conv2d_17 (Conv2D)          (None, 28, 28, 6)         156       
                                                                 
 average_pooling2d_2 (Averag  (None, 14, 14, 6)        0         
 ePooling2D)                                                     
                                                                 
 conv2d_18 (Conv2D)          (None, 10, 10, 16)        2416      
                                                                 
 average_pooling2d_3 (Averag  (None, 5, 5, 16)         0         
 ePooling2D)                                                     
                                                                 
 flatten_6 (Flatten)         (None, 400)               0         
                                                                 
 dense_16 (Dense)            (None, 120)               48120     
                                                                 
 dense_17 (Dense)            (None, 84)                10164     
                                                                 
 dense_18 (Dense)            (None, 10)                850       
                                                                 
=================================================================
Total params: 61,706
Trainable params: 61,706
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/5
1875/1875 [==============================] - 20s 10ms/step - loss: 0.2024 - accuracy: 0.9385
Epoch 2/5
1875/1875 [==============================] - 16s 9ms/step - loss: 0.0676 - accuracy: 0.9792
Epoch 3/5
1875/1875 [==============================] - 19s 10ms/step - loss: 0.0486 - accuracy: 0.9848
Epoch 4/5
1875/1875 [==============================] - 18s 9ms/step - loss: 0.0385 - accuracy: 0.9880
Epoch 5/5
1875/1875 [==============================] - 18s 10ms/step - loss: 0.0325 - accuracy: 0.9902
313/313 - 1s - loss: 0.0331 - accuracy: 0.9882 - 1s/epoch - 4ms/step
```

Q2) CNN for CIFAR 10

In [ ]:

```
# CIFAR10
(train_images, train_labels), (test_images,
                               test_labels) = datasets.cifar10.load_data()

#print(test_images.dtype)
# Normalize pixel values to be between 0 and 1
train_images, test_images = train_images / 255.0, test_images / 255.0
#print(test_images.dtype)
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 = ['airplane', 'automobile', 'bird', 'cat',
               'deer', 'dog', 'frog', 'horse', 'ship', 'truck']
```

```
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 [ ]:

```
# CNN for CIFAR10
model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), 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))


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)
print(test_accuracy)
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d_5 (Conv2D)           (None, 30, 30, 32)        896       
                                                                 
 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, 128)         73856     
                                                                 
 max_pooling2d_5 (MaxPooling  (None, 2, 2, 128)        0         
 2D)                                                             
                                                                 
 flatten_2 (Flatten)         (None, 512)               0         
                                                                 
 dense_5 (Dense)             (None, 64)                32832     
                                                                 
 dense_6 (Dense)             (None, 10)                650       
                                                                 
=================================================================
Total params: 126,730
Trainable params: 126,730
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/5
1563/1563 [==============================] - 65s 41ms/step - loss: 1.5034 - accuracy: 0.4483
Epoch 2/5
1563/1563 [==============================] - 55s 35ms/step - loss: 1.1233 - accuracy: 0.6028
Epoch 3/5
1563/1563 [==============================] - 38s 25ms/step - loss: 0.9668 - accuracy: 0.6624
Epoch 4/5
1563/1563 [==============================] - 38s 25ms/step - loss: 0.8636 - accuracy: 0.6988
Epoch 5/5
1563/1563 [==============================] - 39s 25ms/step - loss: 0.7941 - accuracy: 0.7234
313/313 - 2s - loss: 0.8826 - accuracy: 0.6960 - 2s/epoch - 8ms/step
0.6959999799728394
```

Q3) model_base for MNIST

In [ ]:

```
mnist = keras.datasets.mnist
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()

# Padding
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
```

```
train_images.shape:  (60000, 32, 32)
train_labels.shape:  (60000,)
test_images.shape: (10000, 32, 32)
test_labels.shape: (10000,)
```

In [ ]:

```
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),
                   metrics=['accuracy'])
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)

# saving the weights for the Q4 next
model_base.save_weights('saved_weights/')
```

```
Model: "sequential_7"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d_19 (Conv2D)          (None, 30, 30, 32)        320       
                                                                 
 max_pooling2d_12 (MaxPoolin  (None, 15, 15, 32)       0         
 g2D)                                                            
                                                                 
 conv2d_20 (Conv2D)          (None, 13, 13, 64)        18496     
                                                                 
 max_pooling2d_13 (MaxPoolin  (None, 6, 6, 64)         0         
 g2D)                                                            
                                                                 
 conv2d_21 (Conv2D)          (None, 4, 4, 64)          36928     
                                                                 
 flatten_7 (Flatten)         (None, 1024)              0         
                                                                 
 dense_19 (Dense)            (None, 64)                65600     
                                                                 
 dense_20 (Dense)            (None, 10)                650       
                                                                 
=================================================================
Total params: 121,994
Trainable params: 121,994
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/2
1875/1875 [==============================] - 79s 42ms/step - loss: 0.1335 - accuracy: 0.9597
Epoch 2/2
1875/1875 [==============================] - 87s 46ms/step - loss: 0.0427 - accuracy: 0.9865
313/313 - 4s - loss: 0.0292 - accuracy: 0.9915 - 4s/epoch - 13ms/step
```

Q4) Loading weights to model_lw

In [ ]:

```
# The second network with exact same structure as in 3
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),
                 metrics=['accuracy'])

print(model_lw.summary())

# load the saved weights in 3
model_lw.load_weights('saved_weights/')

# train for two epochs
model_lw.fit(train_images, train_labels, epochs=2)
test_loss, test_accuracy = model_lw.evaluate(test_images, test_labels, verbose=2)

# save this model for Q5
model_lw.save('saved_model/')
```

```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
```

```
=================================================================
 conv2d_22 (Conv2D)          (None, 30, 30, 32)        320       
                                                                 
 max_pooling2d_14 (MaxPoolin  (None, 15, 15, 32)       0         
 g2D)                                                            
                                                                 
 conv2d_23 (Conv2D)          (None, 13, 13, 64)        18496     
                                                                 
 max_pooling2d_15 (MaxPoolin  (None, 6, 6, 64)         0         
 g2D)                                                            
                                                                 
 conv2d_24 (Conv2D)          (None, 4, 4, 64)          36928     
                                                                 
 flatten_8 (Flatten)         (None, 1024)              0         
                                                                 
 dense_21 (Dense)            (None, 64)                65600     
                                                                 
 dense_22 (Dense)            (None, 10)                650       
                                                                 
=================================================================
Total params: 121,994
Trainable params: 121,994
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/2
1875/1875 [==============================] - 87s 46ms/step - loss: 0.0304 - accuracy: 0.9905
Epoch 2/2
1875/1875 [==============================] - 73s 39ms/step - loss: 0.0232 - accuracy: 0.9925
313/313 - 5s - loss: 0.0303 - accuracy: 0.9906 - 5s/epoch - 15ms/step
WARNING:absl:Found untraced functions such as _jit_compiled_convolution_op, _jit_compiled_convolution_op, _jit_compiled_convolution_op while saving (showing 3 of 3). These functions will not be directly callable after loading.
INFO:tensorflow:Assets written to: saved_model/assets
INFO:tensorflow:Assets written to: saved_model/assets
```

Q5) Loading model

In [ ]:

```
# Loading the above model as model_ld
model_ld = keras.models.load_model('saved_model/')
print(model_ld.summary())
model_ld.evaluate(test_images, test_labels, verbose=2)
```

```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d_22 (Conv2D)          (None, 30, 30, 32)        320       
                                                                 
 max_pooling2d_14 (MaxPoolin  (None, 15, 15, 32)       0         
 g2D)                                                            
                                                                 
 conv2d_23 (Conv2D)          (None, 13, 13, 64)        18496     
                                                                 
 max_pooling2d_15 (MaxPoolin  (None, 6, 6, 64)         0         
 g2D)                                                            
                                                                 
 conv2d_24 (Conv2D)          (None, 4, 4, 64)          36928     
                                                                 
 flatten_8 (Flatten)         (None, 1024)              0         
                                                                 
 dense_21 (Dense)            (None, 64)                65600     
                                                                 
 dense_22 (Dense)            (None, 10)                650       
                                                                 
=================================================================
Total params: 121,994
Trainable params: 121,994
Non-trainable params: 0
_________________________________________________________________
None
313/313 - 3s - loss: 0.0303 - accuracy: 0.9906 - 3s/epoch - 10ms/step
```

Out[ ]:

```
[0.030290938913822174, 0.9905999898910522]
```

Q6) Transfer Learning

In [ ]:

```
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),
                  metrics=['accuracy'])
print(new_model.summary())

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

```
Model: "model_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d_22_input (InputLayer  [(None, 32, 32, 1)]      0         
 )                                                               
                                                                 
 conv2d_22 (Conv2D)          (None, 30, 30, 32)        320       
                                                                 
 max_pooling2d_14 (MaxPoolin  (None, 15, 15, 32)       0         
 g2D)                                                            
                                                                 
 conv2d_23 (Conv2D)          (None, 13, 13, 64)        18496     
                                                                 
 max_pooling2d_15 (MaxPoolin  (None, 6, 6, 64)         0         
 g2D)                                                            
                                                                 
 conv2d_24 (Conv2D)          (None, 4, 4, 64)          36928     
                                                                 
 flatten_8 (Flatten)         (None, 1024)              0         
                                                                 
 dense_21 (Dense)            (None, 64)                65600     
                                                                 
 dense_23 (Dense)            (None, 10)                650       
                                                                 
=================================================================
Total params: 121,994
Trainable params: 121,994
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/3
1875/1875 - 56s - loss: 0.0820 - accuracy: 0.9773 - 56s/epoch - 30ms/step
Epoch 2/3
1875/1875 - 63s - loss: 0.0192 - accuracy: 0.9938 - 63s/epoch - 33ms/step
Epoch 3/3
1875/1875 - 60s - loss: 0.0145 - accuracy: 0.9955 - 60s/epoch - 32ms/step
313/313 - 4s - loss: 0.0295 - accuracy: 0.9925 - 4s/epoch - 11ms/step
```

```
Out[ ]: [0.029508670791983604, 0.9925000071525574]
```

Q7) Fine Tuning

In [ ]:

```
# Loading the saved model
model_for_tl = keras.models.load_model('saved_model/')
# makeing the loaded layers non-trainable
model_for_tl.trainable = False
for layer in model_for_tl.layers:
    assert layer.trainable == False

# repeat process in Q6
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),
                  metrics=['accuracy'])

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 - 18s - loss: 0.2527 - accuracy: 0.9451 - 18s/epoch - 10ms/step
Epoch 2/3
1875/1875 - 18s - loss: 0.0168 - accuracy: 0.9954 - 18s/epoch - 10ms/step
Epoch 3/3
1875/1875 - 20s - loss: 0.0124 - accuracy: 0.9965 - 20s/epoch - 11ms/step
313/313 - 3s - loss: 0.0246 - accuracy: 0.9924 - 3s/epoch - 11ms/step
```

```
Out[ ]: [0.024571765214204788, 0.9923999905586243]
```