Multilayer_Perceptron

December 7, 2023

[1]:
```
import tensorflow as tf
from tensorflow.keras import layers, models
from tensorflow.keras.datasets import mnist
from tensorflow.keras.utils import to_categorical

# Load and preprocess the MNIST dataset
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()
train_images = train_images.reshape((60000, 28, 28, 1)).astype('float32') / 255
test_images = test_images.reshape((10000, 28, 28, 1)).astype('float32') / 255

train_labels = to_categorical(train_labels)
test_labels = to_categorical(test_labels)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-
datasets/mnist.npz
11490434/11490434 [==============================] - 0s 0us/step
```

[2]:
```
# Build the MLP model
model = models.Sequential()
model.add(layers.Flatten(input_shape=(28, 28, 1)))
model.add(layers.Dense(128, activation='relu'))
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(10, activation='softmax'))

# Compile the model
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

[3]:
```
# Train the model
model.fit(train_images, train_labels, epochs=5, batch_size=64,␣
 ↪validation_split=0.2)

# Evaluate the model on the test set
test_loss, test_acc = model.evaluate(test_images, test_labels)
print(f'Test accuracy: {test_acc}')
```

```
Epoch 1/5
```

```
750/750 [==============================] - 3s 3ms/step - loss: 0.3085 -
accuracy: 0.9118 - val_loss: 0.1684 - val_accuracy: 0.9521
Epoch 2/5
750/750 [==============================] - 2s 3ms/step - loss: 0.1292 -
accuracy: 0.9615 - val_loss: 0.1246 - val_accuracy: 0.9638
Epoch 3/5
750/750 [==============================] - 2s 3ms/step - loss: 0.0893 -
accuracy: 0.9726 - val_loss: 0.1010 - val_accuracy: 0.9697
Epoch 4/5
750/750 [==============================] - 2s 3ms/step - loss: 0.0659 -
accuracy: 0.9801 - val_loss: 0.1057 - val_accuracy: 0.9668
Epoch 5/5
750/750 [==============================] - 2s 2ms/step - loss: 0.0517 -
accuracy: 0.9841 - val_loss: 0.1035 - val_accuracy: 0.9690
313/313 [==============================] - 0s 1ms/step - loss: 0.0940 -
accuracy: 0.9698
Test accuracy: 0.9697999954223633
```

0.1 Extra Code for Understanding Purpose

[4]:
```
# Example integer labels
labels = [0, 1, 2, 0, 2]

# Number of classes (digits 0 to 9 in the case of MNIST)
num_classes = 10

# Convert labels to one-hot encoding
one_hot_labels = to_categorical(labels, num_classes=num_classes)

print(one_hot_labels)
```

```
[[1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]]
```