mnist_mlp

September 4, 2021

[1]:

```
from tensorflow import keras
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.optimizers import RMSprop

batch_size = 128
num_classes = 10
epochs = 20

# the data, split between train and test sets
(x_train, y_train), (x_test, y_test) = mnist.load_data()

x_train = x_train.reshape(60000, 784)
x_test = x_test.reshape(10000, 784)
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /= 255
x_test /= 255
print(x_train.shape[0], 'train samples')
print(x_test.shape[0], 'test samples')

# convert class vectors to binary class matrices
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)

model = Sequential()
model.add(Dense(512, activation='relu', input_shape=(784,)))
model.add(Dropout(0.2))
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))

model.summary()

model.compile(loss='categorical_crossentropy',
              optimizer=RMSprop(),
              metrics=['accuracy'])

history = model.fit(x_train, y_train,
                    batch_size=batch_size,
                    epochs=epochs,
                    verbose=1,
                    validation_data=(x_test, y_test))
score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
60000 train samples
10000 test samples
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 512)               401920
_________________________________________________________________
dropout (Dropout)            (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               262656
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 10)                5130
=================================================================
Total params: 669,706
Trainable params: 669,706
Non-trainable params: 0
_________________________________________________________________
Epoch 1/20
469/469 [==============================] - 5s 10ms/step - loss: 0.4337 -
accuracy: 0.8609 - val_loss: 0.0994 - val_accuracy: 0.9706
Epoch 2/20
469/469 [==============================] - 5s 10ms/step - loss: 0.1047 -
accuracy: 0.9685 - val_loss: 0.0774 - val_accuracy: 0.9780
Epoch 3/20
469/469 [==============================] - 5s 10ms/step - loss: 0.0743 -
accuracy: 0.9763 - val_loss: 0.0897 - val_accuracy: 0.9747
Epoch 4/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0568 -
accuracy: 0.9826 - val_loss: 0.0837 - val_accuracy: 0.9782
Epoch 5/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0474 -
accuracy: 0.9849 - val_loss: 0.0737 - val_accuracy: 0.9827
Epoch 6/20
```

```
469/469 [==============================] - 4s 9ms/step - loss: 0.0397 -
accuracy: 0.9874 - val_loss: 0.0842 - val_accuracy: 0.9802
Epoch 7/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0375 -
accuracy: 0.9885 - val_loss: 0.0949 - val_accuracy: 0.9791
Epoch 8/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0322 -
accuracy: 0.9906 - val_loss: 0.0905 - val_accuracy: 0.9810
Epoch 9/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0276 -
accuracy: 0.9919 - val_loss: 0.0860 - val_accuracy: 0.9833
Epoch 10/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0277 -
accuracy: 0.9914 - val_loss: 0.0907 - val_accuracy: 0.9823
Epoch 11/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0242 -
accuracy: 0.9927 - val_loss: 0.1009 - val_accuracy: 0.9823
Epoch 12/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0205 -
accuracy: 0.9943 - val_loss: 0.0953 - val_accuracy: 0.9827
Epoch 13/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0209 -
accuracy: 0.9942 - val_loss: 0.1068 - val_accuracy: 0.9822
Epoch 14/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0211 -
accuracy: 0.9941 - val_loss: 0.1072 - val_accuracy: 0.9831
Epoch 15/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0175 -
accuracy: 0.9946 - val_loss: 0.1216 - val_accuracy: 0.9812
Epoch 16/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0188 -
accuracy: 0.9947 - val_loss: 0.1312 - val_accuracy: 0.9822
Epoch 17/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0169 -
accuracy: 0.9958 - val_loss: 0.1184 - val_accuracy: 0.9826
Epoch 18/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0166 -
accuracy: 0.9954 - val_loss: 0.1259 - val_accuracy: 0.9840
Epoch 19/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0179 -
accuracy: 0.9956 - val_loss: 0.1208 - val_accuracy: 0.9819
Epoch 20/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0154 -
accuracy: 0.9955 - val_loss: 0.1470 - val_accuracy: 0.9810
Test loss: 0.14697803556919098
Test accuracy: 0.9810000061988831
```

[ ]: