Somprakash_mlp

March 21, 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])
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-
datasets/mnist.npz
11493376/11490434 [==============================] - 1s 0us/step
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: 0.4408 -
accuracy: 0.8612 - val_loss: 0.1254 - val_accuracy: 0.9593
Epoch 2/20
469/469 [==============================] - 5s 10ms/step - loss: 0.1038 -
accuracy: 0.9673 - val_loss: 0.0806 - val_accuracy: 0.9773
Epoch 3/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0734 -
accuracy: 0.9763 - val_loss: 0.0796 - val_accuracy: 0.9757
Epoch 4/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0579 -
accuracy: 0.9819 - val_loss: 0.0761 - val_accuracy: 0.9788
Epoch 5/20
```

```
469/469 [==============================] - 4s 9ms/step - loss: 0.0493 -
accuracy: 0.9846 - val_loss: 0.0767 - val_accuracy: 0.9804
Epoch 6/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0427 -
accuracy: 0.9878 - val_loss: 0.0806 - val_accuracy: 0.9817
Epoch 7/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0360 -
accuracy: 0.9884 - val_loss: 0.0842 - val_accuracy: 0.9810
Epoch 8/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0323 -
accuracy: 0.9907 - val_loss: 0.0916 - val_accuracy: 0.9800
Epoch 9/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0297 -
accuracy: 0.9913 - val_loss: 0.1030 - val_accuracy: 0.9808
Epoch 10/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0265 -
accuracy: 0.9924 - val_loss: 0.0905 - val_accuracy: 0.9825
Epoch 11/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0256 -
accuracy: 0.9927 - val_loss: 0.0827 - val_accuracy: 0.9849
Epoch 12/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0212 -
accuracy: 0.9935 - val_loss: 0.1070 - val_accuracy: 0.9805
Epoch 13/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0236 -
accuracy: 0.9942 - val_loss: 0.1002 - val_accuracy: 0.9813
Epoch 14/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0208 -
accuracy: 0.9944 - val_loss: 0.1283 - val_accuracy: 0.9809
Epoch 15/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0222 -
accuracy: 0.9938 - val_loss: 0.1063 - val_accuracy: 0.9823
Epoch 16/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0184 -
accuracy: 0.9947 - val_loss: 0.1257 - val_accuracy: 0.9813
Epoch 17/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0187 -
accuracy: 0.9945 - val_loss: 0.1207 - val_accuracy: 0.9835
Epoch 18/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0153 -
accuracy: 0.9960 - val_loss: 0.1238 - val_accuracy: 0.9831
Epoch 19/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0198 -
accuracy: 0.9948 - val_loss: 0.1343 - val_accuracy: 0.9828
Epoch 20/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0175 -
accuracy: 0.9955 - val_loss: 0.1398 - val_accuracy: 0.9828
Test loss: 0.1398448497056961
```

```
Test accuracy: 0.9828000068664551
```

[ ]: