mnist_mlp

March 20, 2021

[1]:
```
'''Trains a simple deep NN on the MNIST dataset.

Gets to 98.40% test accuracy after 20 epochs
(there is *a lot* of margin for parameter tuning).
2 seconds per epoch on a K520 GPU.
'''

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 11ms/step - loss: 0.4398 -
accuracy: 0.8616 - val_loss: 0.1205 - val_accuracy: 0.9634
Epoch 2/20
469/469 [==============================] - 4s 10ms/step - loss: 0.1101 -
accuracy: 0.9663 - val_loss: 0.1262 - val_accuracy: 0.9631
```

```
Epoch 3/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0757 -
accuracy: 0.9772 - val_loss: 0.0746 - val_accuracy: 0.9784
Epoch 4/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0539 -
accuracy: 0.9833 - val_loss: 0.0897 - val_accuracy: 0.9764
Epoch 5/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0486 -
accuracy: 0.9847 - val_loss: 0.0741 - val_accuracy: 0.9808
Epoch 6/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0428 -
accuracy: 0.9875 - val_loss: 0.0876 - val_accuracy: 0.9793
Epoch 7/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0381 -
accuracy: 0.9883 - val_loss: 0.0770 - val_accuracy: 0.9821
Epoch 8/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0340 -
accuracy: 0.9894 - val_loss: 0.0865 - val_accuracy: 0.9819
Epoch 9/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0281 -
accuracy: 0.9915 - val_loss: 0.0898 - val_accuracy: 0.9814
Epoch 10/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0256 -
accuracy: 0.9921 - val_loss: 0.0839 - val_accuracy: 0.9840
Epoch 11/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0238 -
accuracy: 0.9931 - val_loss: 0.0921 - val_accuracy: 0.9823
Epoch 12/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0230 -
accuracy: 0.9938 - val_loss: 0.1112 - val_accuracy: 0.9813
Epoch 13/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0231 -
accuracy: 0.9937 - val_loss: 0.0954 - val_accuracy: 0.9845
Epoch 14/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0199 -
accuracy: 0.9939 - val_loss: 0.1168 - val_accuracy: 0.9821
Epoch 15/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0177 -
accuracy: 0.9947 - val_loss: 0.1236 - val_accuracy: 0.9815
Epoch 16/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0178 -
accuracy: 0.9949 - val_loss: 0.1074 - val_accuracy: 0.9831
Epoch 17/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0187 -
accuracy: 0.9950 - val_loss: 0.1050 - val_accuracy: 0.9830
Epoch 18/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0158 -
accuracy: 0.9957 - val_loss: 0.1290 - val_accuracy: 0.9824
```

```
Epoch 19/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0190 -
accuracy: 0.9951 - val_loss: 0.1221 - val_accuracy: 0.9826
Epoch 20/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0162 -
accuracy: 0.9953 - val_loss: 0.1376 - val_accuracy: 0.9812
Test loss: 0.13755345344543457
Test accuracy: 0.9811999797821045
```

[ ]: