mnist_mlp

December 3, 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'))
```

```python
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: 0.2450 -
accuracy: 0.9241 - val_loss: 0.1008 - val_accuracy: 0.9681
Epoch 2/20
469/469 [==============================] - 4s 10ms/step - loss: 0.1014 -
accuracy: 0.9693 - val_loss: 0.0847 - val_accuracy: 0.9731
Epoch 3/20
469/469 [==============================] - 5s 10ms/step - loss: 0.0761 -
accuracy: 0.9765 - val_loss: 0.0848 - val_accuracy: 0.9761
```

```
Epoch 4/20
469/469 [==============================] - 5s 10ms/step - loss: 0.0597 -
accuracy: 0.9823 - val_loss: 0.0681 - val_accuracy: 0.9807
Epoch 5/20
469/469 [==============================] - 5s 10ms/step - loss: 0.0511 -
accuracy: 0.9843 - val_loss: 0.0712 - val_accuracy: 0.9815
Epoch 6/20
469/469 [==============================] - 5s 10ms/step - loss: 0.0436 -
accuracy: 0.9869 - val_loss: 0.0756 - val_accuracy: 0.9819
Epoch 7/20
469/469 [==============================] - 5s 10ms/step - loss: 0.0386 -
accuracy: 0.9887 - val_loss: 0.0789 - val_accuracy: 0.9833
Epoch 8/20
469/469 [==============================] - 4s 10ms/step - loss: 0.0335 -
accuracy: 0.9897 - val_loss: 0.0871 - val_accuracy: 0.9821
Epoch 9/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0304 -
accuracy: 0.9906 - val_loss: 0.0885 - val_accuracy: 0.9834
Epoch 10/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0329 -
accuracy: 0.9912 - val_loss: 0.0797 - val_accuracy: 0.9836
Epoch 11/20
469/469 [==============================] - 4s 10ms/step - loss: 0.0257 -
accuracy: 0.9922 - val_loss: 0.0898 - val_accuracy: 0.9830
Epoch 12/20
469/469 [==============================] - 5s 10ms/step - loss: 0.0259 -
accuracy: 0.9925 - val_loss: 0.1038 - val_accuracy: 0.9825
Epoch 13/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0232 -
accuracy: 0.9931 - val_loss: 0.0994 - val_accuracy: 0.9825
Epoch 14/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0222 -
accuracy: 0.9938 - val_loss: 0.0995 - val_accuracy: 0.9849
Epoch 15/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0226 -
accuracy: 0.9936 - val_loss: 0.1026 - val_accuracy: 0.9821
Epoch 16/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0207 -
accuracy: 0.9941 - val_loss: 0.0991 - val_accuracy: 0.9850
Epoch 17/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0195 -
accuracy: 0.9950 - val_loss: 0.1154 - val_accuracy: 0.9834
Epoch 18/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0183 -
accuracy: 0.9951 - val_loss: 0.1172 - val_accuracy: 0.9845
Epoch 19/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0164 -
accuracy: 0.9954 - val_loss: 0.1244 - val_accuracy: 0.9838
```

```
Epoch 20/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0169 -
accuracy: 0.9951 - val_loss: 0.1170 - val_accuracy: 0.9856
Test loss: 0.11704535782337189
Test accuracy: 0.9855999946594238
```

[ ]: