Assignment01_RanjanVikas

March 21, 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 10ms/step - loss: 0.4345 -
accuracy: 0.8622 - val_loss: 0.1076 - val_accuracy: 0.9667
Epoch 2/20
469/469 [==============================] - 5s 10ms/step - loss: 0.1055 -
accuracy: 0.9677 - val_loss: 0.0872 - val_accuracy: 0.9740
Epoch 3/20
```

```
469/469 [==============================] - 4s 9ms/step - loss: 0.0767 -
accuracy: 0.9766 - val_loss: 0.0746 - val_accuracy: 0.9773
Epoch 4/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0579 -
accuracy: 0.9821 - val_loss: 0.0776 - val_accuracy: 0.9792
Epoch 5/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0520 -
accuracy: 0.9838 - val_loss: 0.0809 - val_accuracy: 0.9819
Epoch 6/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0408 -
accuracy: 0.9878 - val_loss: 0.0758 - val_accuracy: 0.9827
Epoch 7/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0345 -
accuracy: 0.9897 - val_loss: 0.0727 - val_accuracy: 0.9831
Epoch 8/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0309 -
accuracy: 0.9908 - val_loss: 0.0964 - val_accuracy: 0.9807
Epoch 9/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0282 -
accuracy: 0.9916 - val_loss: 0.0883 - val_accuracy: 0.9818
Epoch 10/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0285 -
accuracy: 0.9918 - val_loss: 0.0958 - val_accuracy: 0.9832
Epoch 11/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0249 -
accuracy: 0.9925 - val_loss: 0.0943 - val_accuracy: 0.9821
Epoch 12/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0237 -
accuracy: 0.9933 - val_loss: 0.1014 - val_accuracy: 0.9826
Epoch 13/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0211 -
accuracy: 0.9939 - val_loss: 0.0938 - val_accuracy: 0.9842
Epoch 14/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0209 -
accuracy: 0.9942 - val_loss: 0.1021 - val_accuracy: 0.9837
Epoch 15/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0174 -
accuracy: 0.9950 - val_loss: 0.1150 - val_accuracy: 0.9814
Epoch 16/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0186 -
accuracy: 0.9945 - val_loss: 0.1140 - val_accuracy: 0.9849
Epoch 17/20
469/469 [==============================] - 4s 8ms/step - loss: 0.0218 -
accuracy: 0.9935 - val_loss: 0.1035 - val_accuracy: 0.9830
Epoch 18/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0152 -
accuracy: 0.9958 - val_loss: 0.1021 - val_accuracy: 0.9857
Epoch 19/20
```

```
469/469 [==============================] - 4s 9ms/step - loss: 0.0156 -
accuracy: 0.9958 - val_loss: 0.1189 - val_accuracy: 0.9841
Epoch 20/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0221 -
accuracy: 0.9947 - val_loss: 0.1262 - val_accuracy: 0.9827
Test loss: 0.12620897591114044
Test accuracy: 0.982699990272522
```

[ ]: