Input

```
'''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])
```

Output

```
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 [==============================] - 6s 11ms/step - loss: 0.4385 - accuracy: 0.8614 - val_loss: 0.1140 - val_accuracy: 0.9634
Epoch 2/20
469/469 [==============================] - 4s 10ms/step - loss: 0.1055 - accuracy: 0.9667 - val_loss: 0.0799 - val_accuracy: 0.9753
Epoch 3/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0738 - accuracy: 0.9767 - val_loss: 0.0751 - val_accuracy: 0.9776
Epoch 4/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0567 - accuracy: 0.9816 - val_loss: 0.0653 - val_accuracy: 0.9821
Epoch 5/20
```

```
469/469 [==============================] - 4s 9ms/step - loss: 0.0454 - ac
curacy: 0.9854 - val_loss: 0.0733 - val_accuracy: 0.9794
Epoch 6/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0422 - ac
curacy: 0.9870 - val_loss: 0.0837 - val_accuracy: 0.9804
Epoch 7/20
469/469 [==============================] - 5s 10ms/step - loss: 0.0370 - a
ccuracy: 0.9893 - val_loss: 0.0760 - val_accuracy: 0.9834
Epoch 8/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0323 - ac
curacy: 0.9900 - val_loss: 0.0864 - val_accuracy: 0.9796
Epoch 9/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0285 - ac
curacy: 0.9911 - val_loss: 0.0941 - val_accuracy: 0.9817
Epoch 10/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0264 - ac
curacy: 0.9925 - val_loss: 0.1027 - val_accuracy: 0.9803
Epoch 11/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0220 - ac
curacy: 0.9937 - val_loss: 0.1106 - val_accuracy: 0.9817
Epoch 12/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0246 - ac
curacy: 0.9930 - val_loss: 0.1086 - val_accuracy: 0.9817
Epoch 13/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0216 - ac
curacy: 0.9932 - val_loss: 0.1060 - val_accuracy: 0.9838
Epoch 14/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0210 - ac
curacy: 0.9941 - val_loss: 0.1046 - val_accuracy: 0.9827
Epoch 15/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0182 - ac
curacy: 0.9950 - val_loss: 0.1030 - val_accuracy: 0.9837
Epoch 16/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0183 - ac
curacy: 0.9946 - val_loss: 0.0974 - val_accuracy: 0.9834
Epoch 17/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0159 - ac
curacy: 0.9954 - val_loss: 0.1124 - val_accuracy: 0.9842
Epoch 18/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0168 - ac
curacy: 0.9950 - val_loss: 0.1217 - val_accuracy: 0.9838
Epoch 19/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0164 - ac
curacy: 0.9956 - val_loss: 0.1174 - val_accuracy: 0.9829
Epoch 20/20
469/469 [==============================] - 4s 9ms/step - loss: 0.0181 - ac
curacy: 0.9954 - val_loss: 0.1175 - val_accuracy: 0.9832
Test loss: 0.11749401688575745
Test accuracy: 0.9832000136375427
```