
1: The First Problem

(a) Algorithm:

```
import numpy as np
import os
import sys
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dropout, Flatten, Dense
from tensorflow.keras import backend as K
from tensorflow.keras.models import Sequential
from tensorflow.keras.datasets import mnist
import matplotlib.pyplot as plt
import pandas as pd
import warnings
warnings.filterwarnings("ignore")

batch_size = 128
num_classes = 10
epochs = 10

img_rows, img_cols = 28, 28

(x_train, y_train), (x_test, y_test) = mnist.load_data()

if K.image_data_format() == 'channels_first':
    x_train = x_train.reshape(x_train.shape[0], 1, img_rows, img_cols)
    x_test = x_test.reshape(x_test.shape[0], 1, img_rows, img_cols)
    input_shape = (1, img_rows, img_cols)
else:
    x_train = x_train.reshape(x_train.shape[0], img_rows, img_cols, 1)
    x_test = x_test.reshape(x_test.shape[0], img_rows, img_cols, 1)
    input_shape = (img_rows, img_cols, 1)

x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /= 255
x_test /= 255

print('train samples: ', x_train.shape[0])
print('test samples: ', x_test.shape[0])

y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)
```

```

model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3),
                 activation='relu',
                 input_shape=input_shape))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))

model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adadelta(),
              metrics=['accuracy'])

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 acc:', score[1])

```

(b) Output:

```

jupyter lab9 Last checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Help Trusted Python 3.7.6 64-bit
In [23]: epochs=epochs,
         validation_data=(x_test, y_test))

Train on 40000 samples, validate on 10000 samples
Epoch 1/10 [-----] - 230s 4ms/sample - loss: 0.2690 - acc: 0.9166 - val_loss: 0.0572 - val_
acc: 0.9813
Epoch 2/10 [-----] - 219s 4ms/sample - loss: 0.0895 - acc: 0.9736 - val_loss: 0.0403 - val_
acc: 0.9868
Epoch 3/10 [-----] - 242s 4ms/sample - loss: 0.0663 - acc: 0.9803 - val_loss: 0.0372 - val_
acc: 0.9869
Epoch 4/10 [-----] - 181s 3ms/sample - loss: 0.0550 - acc: 0.9831 - val_loss: 0.0310 - val_
acc: 0.9890
Epoch 5/10 [-----] - 192s 3ms/sample - loss: 0.0460 - acc: 0.9862 - val_loss: 0.0313 - val_
acc: 0.9904
Epoch 6/10 [-----] - 263s 4ms/sample - loss: 0.0430 - acc: 0.9869 - val_loss: 0.0285 - val_
acc: 0.9905
Epoch 7/10 [-----] - 241s 4ms/sample - loss: 0.0374 - acc: 0.9891 - val_loss: 0.0301 - val_
acc: 0.9908
Epoch 8/10 [-----] - 160s 3ms/sample - loss: 0.0344 - acc: 0.9897 - val_loss: 0.0258 - val_
acc: 0.9917
Epoch 9/10 [-----] - 162s 3ms/sample - loss: 0.0333 - acc: 0.9901 - val_loss: 0.0255 - val_
acc: 0.9914
Epoch 10/10 [-----] - 158s 3ms/sample - loss: 0.0315 - acc: 0.9906 - val_loss: 0.0260 - val_
acc: 0.9912

Out[23]: <tensorflow.python.keras.callbacks.History at 0x12d4c8bd0>

In [24]: score = model.evaluate(x_test, y_test, verbose=0)
         print('Test loss:', score[0])
         print('Test acc:', score[1])

Test loss: 0.025975444087418326
Test acc: 0.9912

```