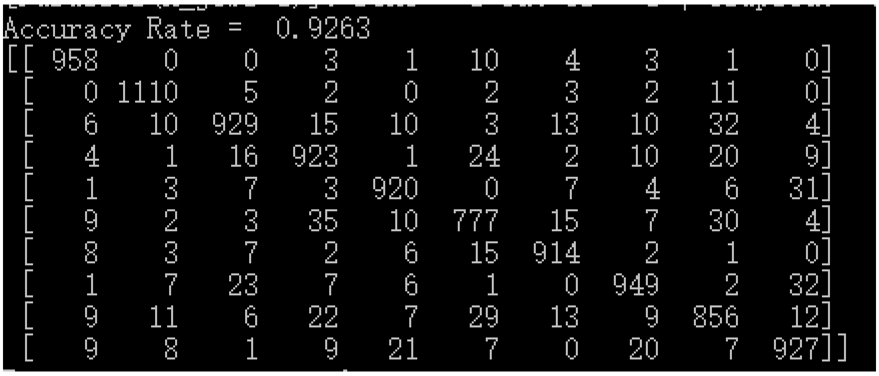
COEN 240 Machine Learning

Homework #3

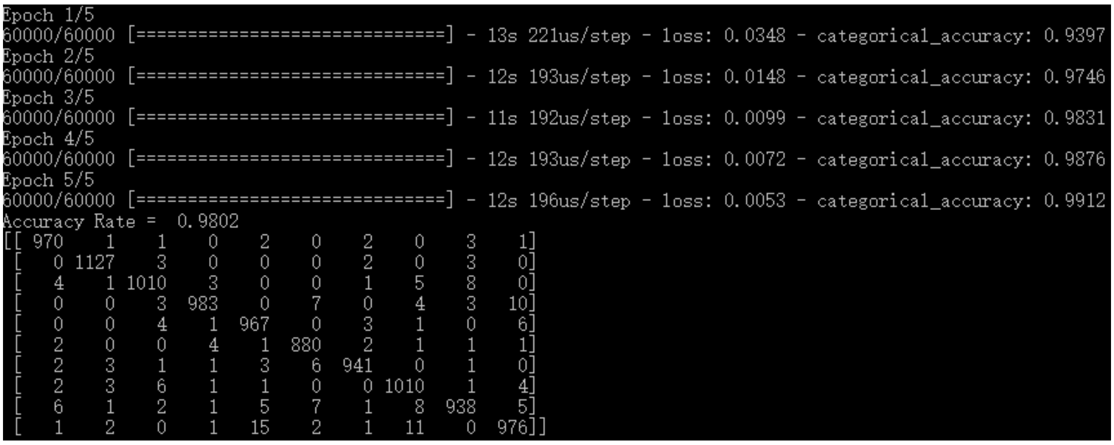
Name: Jinhao Wang   ID: 4302178

Problem1:



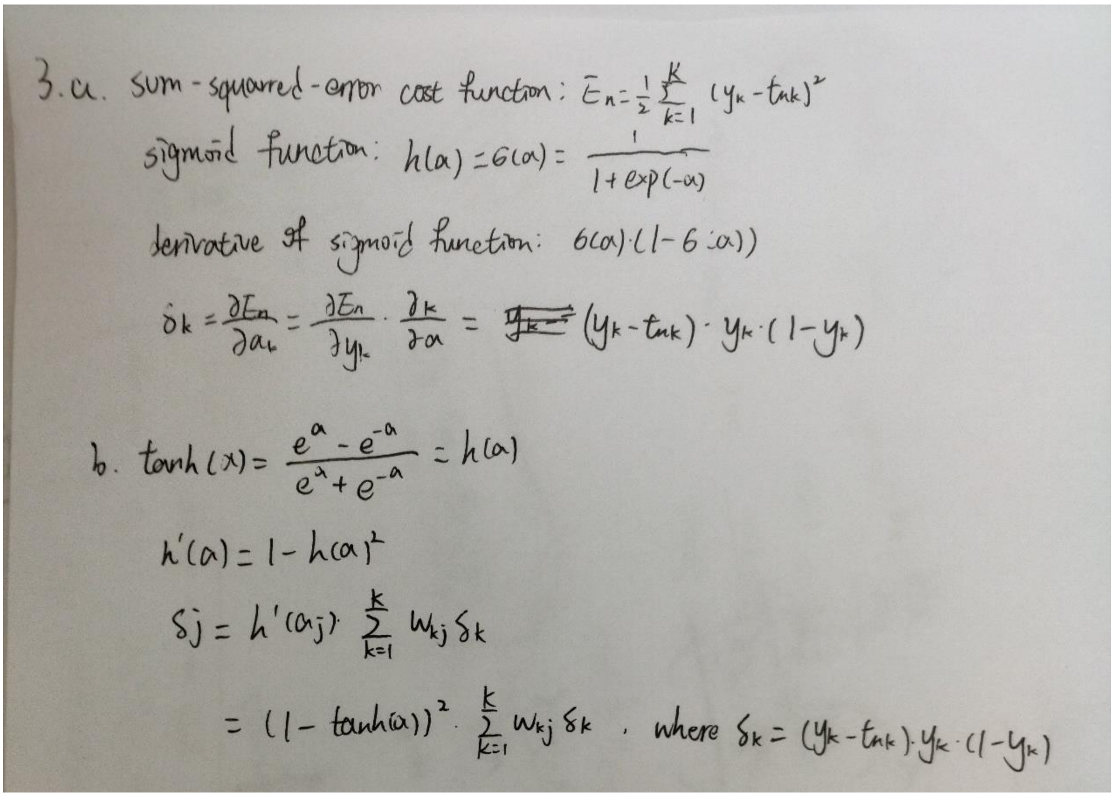
```
Accuracy Rate =  0.9263
[[ 958    0    0    3    1   10    4    3    1    0]
 [   0 1110    5    2    0    2    3    2   11    0]
 [   6   10  929   15   10    3   13   10   32    4]
 [   4    1   16  923    1   24    2   10   20    9]
 [   1    3    7    3  920    0    7    4    6   31]
 [   9    2    3   35   10  777   15    7   30    4]
 [   8    3    7    2    6   15  914    2    1    0]
 [   1    7   23    7    6    1    0  949    2   32]
 [   9   11    6   22    7   29   13    9  856   12]
 [   9    8    1    9   21    7    0   20    7  927]]
```

Problem 2:



```
Epoch 1/5
60000/60000 [==============================] - 13s 221us/step - loss: 0.0348 - categorical_accuracy: 0.9397
Epoch 2/5
60000/60000 [==============================] - 12s 193us/step - loss: 0.0148 - categorical_accuracy: 0.9746
Epoch 3/5
60000/60000 [==============================] - 11s 192us/step - loss: 0.0099 - categorical_accuracy: 0.9831
Epoch 4/5
60000/60000 [==============================] - 12s 193us/step - loss: 0.0072 - categorical_accuracy: 0.9876
Epoch 5/5
60000/60000 [==============================] - 12s 196us/step - loss: 0.0053 - categorical_accuracy: 0.9912
Accuracy Rate =  0.9802
[[ 970    1    1    0    2    0    2    0    3    1]
 [   0 1127    3    0    0    0    2    0    3    0]
 [   4    1 1010    3    0    0    1    5    8    0]
 [   0    0    3  983    0    7    0    4    3   10]
 [   0    0    4    1  967    0    3    1    0    6]
 [   2    0    0    4    1  880    2    1    1    1]
 [   2    3    1    1    3    6  941    0    1    0]
 [   2    3    6    1    1    0    0 1010    1    4]
 [   6    1    2    1    5    7    1    8  938    5]
 [   1    2    0    1   15    2    1   11    0  976]]
```

Problem 3:



3.a. sum-squarred-error cost function: $E_n = \frac{1}{2}\sum_{k=1}^{K} (y_k - t_{nk})^2$

sigmoid function: $h(a) = \sigma(a) = \frac{1}{1+\exp(-a)}$

derivative of sigmoid function: $\sigma(a)\cdot(1-\sigma(a))$

$$\delta_k = \frac{\partial E_n}{\partial a_k} = \frac{\partial E_n}{\partial y_k}\cdot\frac{\partial_k}{\partial a} = \cancel{y_k}(y_k - t_{nk})\cdot y_k\cdot(1-y_k)$$

b. $\tanh(x) = \frac{e^a - e^{-a}}{e^a + e^{-a}} = h(a)$

$h'(a) = 1 - h(a)^2$

$$\delta_j = h'(a_j)\cdot\sum_{k=1}^{k} w_{kj}\delta_k$$

$$= (1-\tanh(a))^2\cdot\sum_{k=1}^{k} w_{kj}\delta_k \text{ , where } \delta_k = (y_k - t_{nk})\cdot y_k\cdot(1-y_k)$$

Attachment:

Problem 1 Code:

```
import tensorflow as tf
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix

mnist = tf.keras.datasets.mnist
(x_traino, y_train), (x_testo, y_test) = mnist.load_data()
x_train = np.reshape(x_traino, (60000, 28*28))
x_test = np.reshape(x_testo, (10000, 28*28))
x_train = x_train/255.0
x_test = x_test/255.0
logreg = LogisticRegression(solver='saga', multi_class='multinomial',
max_iter=100,verbose=2)
logreg.fit(x_train, y_train)
y_predict = logreg.predict(x_test)
num_correct = 0
for i in range(len(y_test)):
    if y_predict[i]==y_test[i]:
        num_correct +=1
Accuracy_rate = num_correct/len(y_test)
print("Accuracy Rate = ", Accuracy_rate)

cm = confusion_matrix(y_test, y_predict, labels=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
print(cm)
```

Problem 2 Code:

```
import keras
from keras.models import Sequential
from keras.layers import Dense
from keras.datasets import mnist
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.metrics import confusion_matrix

(x_train, y_train), (x_test, y_test) = mnist.load_data()
image_vector_size = 28*28
x_train = x_train.reshape(x_train.shape[0], image_vector_size)/255.0
x_test = x_test.reshape(x_test.shape[0], image_vector_size)/255.0
y_train = keras.utils.to_categorical(y_train, 10)

# create model
model = Sequential()
model.add(Dense(512, input_dim=28*28,  activation='relu'))
model.add(Dense(10,  activation='softmax'))
# Compile model
from keras import optimizers
adam = optimizers.Adam(lr=0.001, beta_1=0.9, beta_2=0.999, amsgrad=False)
model.compile(loss='binary_crossentropy', optimizer=adam,
metrics=['categorical_accuracy'])

# Fit the model
model.fit(x_train, y_train, epochs=5, batch_size=32)
# calculate predictions
predictions = model.predict(x_test) # y
# round predictions
y_predict = np.argmax(predictions, axis=1)

num_correct = 0
for i in range(len(y_test)):
    if y_predict[i]==y_test[i]:
        num_correct +=1

Accuracy_rate = num_correct/len(y_test)
print("Accuracy Rate = ", Accuracy_rate)

cm = confusion_matrix(y_test, y_predict, labels=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
print(cm)
```