COEN 240 MACHINE LEARNING

HOMEWROK THREE

NAME: BOSEN YANG **STUDENT ID: 1589880**

PROBLEM ONE

```
RECOGNITION ACCURACY RATE: 0.9262
THE CONFUSION MATRIX:
[[ 957    0    0    4    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    8    2    6   15  913    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    6  928]]
```

PROBLEM TWO:

```
RECOGNITION ACCURACY RATE: 0.9793
THE CONFUSION MATRIX:
[[ 968    1    1    0    2    1    3    1    1    2]
 [   0 1130    1    0    0    0    2    0    2    0]
 [   4    1 1010    2    1    0    2    4    8    0]
 [   0    0    0  988    1    2    0    6    3   10]
 [   1    0    3    0  967    0    3    1    1    6]
 [   3    2    0    9    1  859   10    2    5    1]
 [   2    2    1    1    2    1  949    0    0    0]
 [   1   10    8    1    1    0    0  993    1   13]
 [   4    0    1    5    6    2    2    4  939   11]
 [   2    2    0    2    6    2    1    4    0  990]]
```

PROBLEM THREE

PROBLEM 3a.

$$\delta_k = \frac{\partial E_n}{\partial a_{nk}} = \frac{\partial E_n}{\partial y_{nk}} \cdot \frac{\partial y_{nk}}{\partial a_{nk}}$$

$$= \frac{\partial \frac{1}{2} \sum_{t=1}^{K} (y_{nt} - t_{nt})^2}{\partial y_{nk}} \cdot \frac{\partial \sigma(a_{nk})}{\partial a_{nk}}$$

$$= \frac{\partial \frac{1}{2} (y_{nk} - t_{nk})^2}{\partial y_{nk}} \cdot \left(\sigma(a_{nk})(1 - \sigma(a_{nk})) \right)$$

$$= (y_{nk} - t_{nk}) \cdot y_{nk} \cdot (1 - y_{nk})$$

PROBLEM 3b.

$$\delta_j = \frac{\partial E_n}{\partial a_j} = \sum_k \frac{\partial E_n}{\partial a_k} \cdot \frac{\partial a_k}{\partial a_j}$$

$$= \sum_k \delta_k \cdot w_{kj} \cdot h'(a_j)$$

$$= h'(a_j) \cdot \sum_k \delta_k \cdot w_{kj}$$

$$= (1 - h^2(a_j)) \cdot \sum_k \delta_k \cdot w_{kj}$$

$$a_k = \sum_j w_{kj} \cdot z_j$$

$$= \sum_j w_{kj} \cdot h(a_j)$$

$$\frac{\partial a_k}{\partial a_j} = w_{kj} \cdot h'(a_j)$$

$h(a)$ is the tanh activation function

take the expression of çk into the final expression

ATTACHMENTS

PROBLEM ONE CODE

```
"""
Created on Sun Jan 26 17:12:27 2020

@author: burson
"""


import tensorflow as tf
import numpy as np
import time
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix

mnist = tf.keras.datasets.mnist

# DATASET ACQUISITION
(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_test = x_train / 255.0, x_test / 255.0

# MODEL CREATION
logreg = LogisticRegression(solver='saga', multi_class='multinomial', max_iter = 100, verbose=2)

# DATA CHECKING
#import matplotlib.pyplot as plt
#plt.figure(figsize=(20,4))
#for index, (image, label) in enumerate(zip(x_train[30:35], y_train[30:35])):
#    plt.subplot(1, 5, index + 1)
#    plt.imshow(np.reshape(image, (28,28)), cmap=plt.cm.gray)
#    plt.title('Training: %i\n' % label, fontsize = 20)

# MODEL FITTING
logreg.fit(x_train, y_train)

# MODEL EVALUATION
time.sleep(0.2)
predictions = logreg.predict(x_test).reshape(-1, 1)
y_test = y_test.reshape(-1, 1)
num_test    = x_test.shape[0]
num_match   = np.count_nonzero(np.equal(predictions, y_test))
```

```
score      = num_match/num_test
print("\n\nRECOGNITION ACCURACY RATE: %.4f" % (score))

print("THE CONFUSION MATRIX: ")
cm = confusion_matrix(y_test, predictions)
print(cm)
```

PROBLEM TWO CODE

```
"""
Created on Sun Jan 26 17:41:40 2020

@author: Burson
"""

import numpy as np
import tensorflow as tf
from keras.models import Sequential
from keras.layers import Dense
from sklearn.metrics import confusion_matrix

mnist = tf.keras.datasets.mnist

# DATASET ACQUISITION
(x_train, y_train),(x_test, y_test) = mnist.load_data()
x_train = x_train.reshape((60000, 28*28))
x_test  = x_test.reshape((10000, 28*28))
x_train, x_test = x_train / 255.0, x_test / 255.0

# MODEL CREATION
model = Sequential()
model.add(Dense(512, activation="relu", input_dim=28*28))
model.add(Dense(10, activation="softmax"))
model.summary()

# MODEL COMPILATION
model.compile(optimizer="adam",
          loss="sparse_categorical_crossentropy",
          metrics=["accuracy"])

# MODEL FITTING
model.fit(x_train, y_train, epochs=5, batch_size=50, verbose=2)
```

```
test_loss, test_acc = model.evaluate(x_train, y_train)
print("\n\nTRAINING SET ACCURACY RATE: %.4f" % (test_acc))

# MODEL EVALUATION
predictions_mat = model.predict(x_test)
predictions     = np.argmax(predictions_mat, axis=1)
num_test        = x_test.shape[0]
num_match       = np.count_nonzero(np.equal(predictions, y_test))
score           = num_match/num_test
print("\n\nRECOGNITION ACCURACY RATE: %.4f" % (score))
print("THE CONFUSION MATRIX: ")
cm = confusion_matrix(y_test, predictions)
print(cm)
```