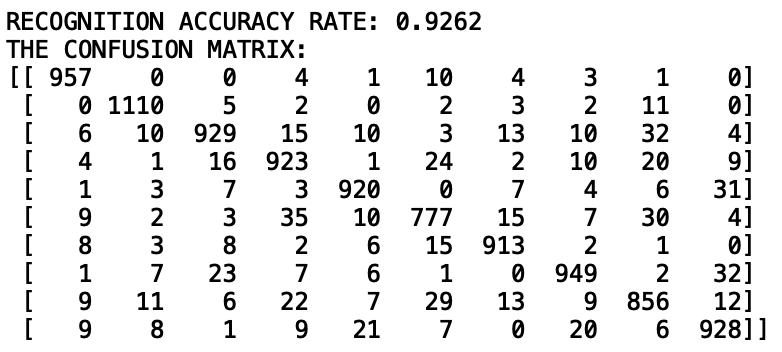
**COEN 240 MACHINE LEARNING**

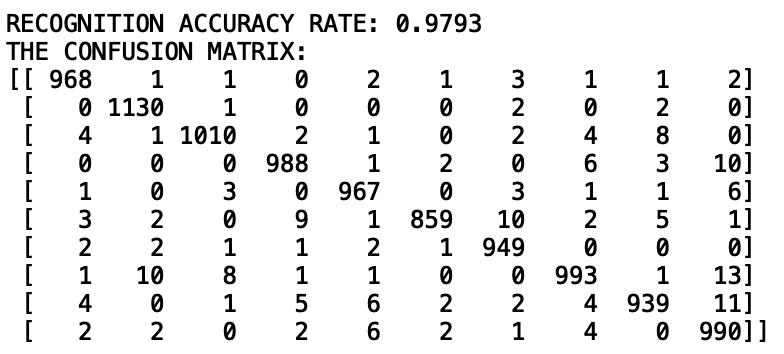
**HOMEWROK THREE**

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

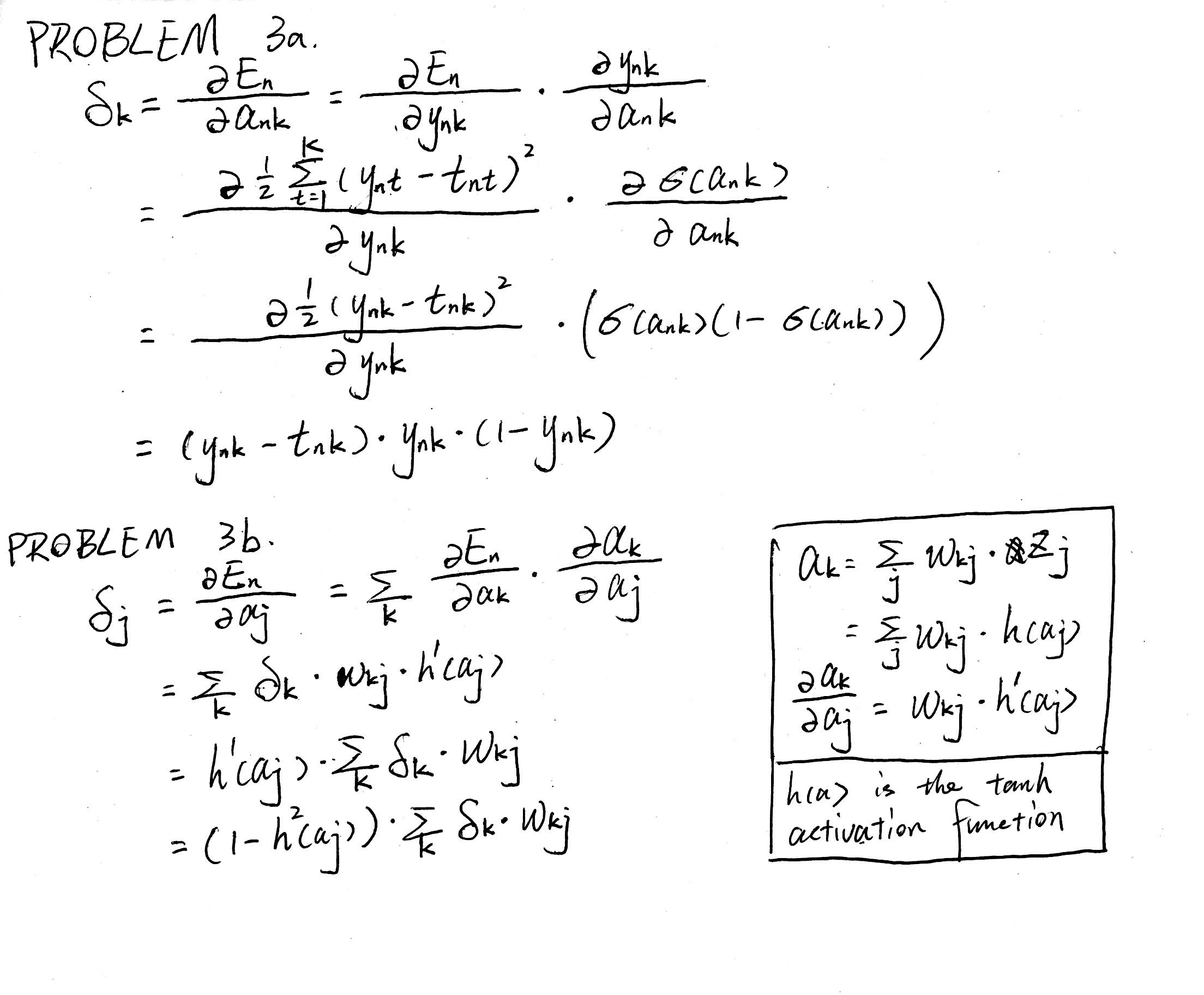
**PROBLEM ONE**



**PROBLEM TWO:**



**PROBLEM THREE**



**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)