Subject: 19CSE456

Lab Session: 04

Notes:

1. Please read the assignment notes carefully and comply to the guidelines provided.
2. A report should be prepared for questions available in “Report Section” of this assignment & submitted to TurnItIn. Only one member of the team should upload the report to TurnItIn. The filename should start with the team name / number.
3. The report should not have any content (text, figures, photos etc.) copied from any source outside your work. Please provide results obtained from your experiment to support your statements.
4. Any content retrieved from any external source (papers, bogs, websites, chat sites etc.) should be duly acknowledged.

Please use your project data as mentioned below:

- **For images use images reshaped as a vector**
- **For videos, split the video to frames and them treat them as images; ignore the sound in video.**
- **For audio, use the audio as a single dimensional vector; for different length, you may use zero padding.**
- **For text, you may vectorize the text using Bag of words or other such techniques.**
- **Any other form of input, please discuss with your mentor / teacher.**

References:

- https://towardsdatascience.com/convolutional-neural-network-feature-map-and-filter-visualization-f75012a5a49c
- https://www.kaggle.com/code/arpitjain007/guide-to-visualize-filters-and-feature-maps-in-cnn
- https://www.kaggle.com/code/amarjeet007/visualize-cnn-with-keras

Main Section (Mandatory):

A1. Design and train a convolutional neural network (CNN). Inspect the CNN architecture. Sample code provided for help.

```
import keras,os
from keras.datasets import fashion_mnist, cifar100
from keras.layers import Dense, Activation, Flatten, Conv2D, MaxPooling
2D, Dropout, BatchNormalization
from keras.models import Sequential
from keras.utils import to_categorical
from keras.preprocessing.image import ImageDataGenerator
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
import cv2
from keras import optimizers
from keras.layers.core import Lambda
from keras import backend as K
from keras import regularizers
from sklearn import datasets # load dataset
from sklearn.model_selection import train_test_split # split dataset
from sklearn.preprocessing import StandardScaler # standard scaler
from sklearn.metrics import accuracy_score # check accuracy
```

```
(train_X,train_Y), (test_X,test_Y) = fashion_mnist.load_data()
train_X = train_X.reshape(-1, 28,28, 1)
test_X = test_X.reshape(-1, 28,28, 1)

train_X = train_X.astype('float32')
test_X = test_X.astype('float32')
train_X = train_X / 255
test_X = test_X / 255

train_X1 = train_X[0:100]
train_Y1 = train_Y[:100]

val_X = train_X[100:150]
val_Y_Onehot = to_categorical(train_Y[100:150])

train_Y_one_hot = to_categorical(train_Y1)
test_Y_one_hot = to_categorical(test_Y)
```

```
model = Sequential()
model.add(Conv2D(64, (3,3), input_shape=(28, 28, 1)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(64, (3,3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
model.add(Dense(64))
model.add(Dense(10))
model.add(Activation('softmax'))
model.compile(loss=keras.losses.categorical_crossentropy, optimizer=keras.optimizers.Adam(),metrics=['accuracy'])

model.summary()
```

A2. Train the model with the training data. Plot the training and validation losses for the training session. Observe the graphs and interpret. Code below for help.

```
history = model.fit(train_X1, train_Y_one_hot, batch_size=64, epochs=100, validation_data=(val_X,val_Y_Onehot))

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.show()
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.show()
```

A3. Evaluate the model accuracy with below code.

```
#Evaluate the model on the test data after training your model
score = model.evaluate(test_X[0:100],test_Y_one_hot[0:100], verbose=1)
print('\nKeras CNN binary accuracy:', score[1],'\n')
```

A4. Inspect the filters (at least a few of them) for the first convolution layer. Below code for help.

```
filters, biases = model.layers[0].get_weights()

for i in range(5):
  plt.imshow(filters[:,:,0,i], cmap='gray')
  plt.show()
```

A5. Inspect the impact a filter creates on an input image with 2D convolution. Below code for help.

```
from scipy import signal

im = train_X[10]
plt.imshow(im,cmap='gray')
plt.show()

ot = signal.convolve2d(im.reshape(28,28),filt[:,:,1].reshape(3,3),boundary='symm',mode='same')
plt.imshow(ot,cmap='gray')

#print(im.reshape(28,28),'\n',filt[:,:,1].reshape(3,3))
#print(ot)
```

A6. Design and implement a fully connected and dense network to perform classification on your dataset. Train the network with training & validation sets.

Note:

1. If your images are of size different from 28 x 28 x 1, you need to resize them to this size for using the above network. Alternately, redesign the network to suit your image size.
2. If your images are of variable size, resize them to some standard size before using them.

A7. Make a plot of training loss and validation loss to check for the regular fit of the trained network.

A8. Test the network with your test set and observe the metrics.

Report Assignment:

1. Update your last week’s report with the study of classification using CNN. The report should contain the following details. [4]
   - CNN architecture as a figure
   - Plot of training and validation losses with interpretations of the plots
   - Data Description section should be updated with description of the project data
   - Results should be provided in results section of the document with through discussion of the obtained results.