> # Name: *Omar Rashad Salem*
>
> # Course: *CV - prof.Heba*
>
> # Assignemnt No.: *4*
>
> ## QUESTIONS

1) **What is a Convolutional Neural Network (CNN) and what makes it suitable for image-related tasks?**

ans:

- CNN is type of neural networks that is used for processing a higher dimensional type of data or cluster of data like images e.g.(rgb / gray scale) it posseses the ability to learn patterns and hierarchical

representations

2) **Explain the concept of local connectivity in CNNs.**

ans: output of neuron in a X layer connects only to a small local region in the next layer that mostly share some similar context. this effectively reduces `CNN` complexity and optimizes it!

---

3) **What are the key parameters of a Convolutional Layer?**

ans:

- No. of filters `K`
- Stride `S`
- Filter size `F`
- Zero padding `P`

---

4) **Why is zero-padding used in Convolutional Layers, and what impact does it have on the spatial arrangement?**

ans:

- preserves the input dimensions / makes it managable
- marks the edges of an images which could come very handy in image processing and computer vision

---

5) **What is the purpose of pooling layer in ConvNet?**

ans:

- reduces the dimensions and parameters which helps boosting CNN over all performance and computing time
- produces no extra parameters

---

6) **SOLVE**

given:

```
- filter size = 2x2
- stride = 2
- i/p size = 28x28x64
- it's a pooling layer
```

- Find o/p Volume?

ans:

$$W = (w - f)/s + 1$$
$$W = (16 - 2)/2 + 1$$
$$\therefore W = 14$$
$$o/p - Volume = 14 * 14 * 64$$

---

7) **SOLVE**

given: - (assume padding = 1) - filter size = 3x3 - filters No. = 32 - stride = 1 - i/p size = 32x32x3 - it's a Convolution layer

- Find o/p Volume?

ans:

$$W = (w - f + 2p)/s + 1$$
$$W = (32 - 3 + 2 * 1)/1 + 1$$
$$\therefore W = 32$$
$$o/p - Volume = 30 * 30 * 32$$

---

8) **SOLVE**

given:

```
- filter size = 3x3
- filters No. = 8
```

- Find No. of parameters including biases?

ans:

$$No. parameters = filter' size * depth * biases$$
$$= 3 * 3 * 3 + 1$$
$$o/p - Volume = 28$$
$$\therefore No. parameters = 28 * 8 = 224(216 weight + 8 biases)$$

---

9) **SOLVE**

given:

```
- i/p volume = 256*256*3
- layer neurons no. = 100
```

- Find No. weights and biases needed?

ans:

$$weights = 256 * 256 * 3 * 100$$
$$weights = 19660800$$
$$biases = 1 * 100 = 100$$

Programming Assignment *(challenge task!)*:

Image Classification using Convolutional Neural Networks (ConvNet)

Code for accuracy FOCUSED CNN model

In [ ]:

```
%pip install imgaug

import matplotlib.pyplot as plt
from imgaug import augmenters as iaa
import tensorflow as tf
import numpy as np
from sklearn.model_selection import train_test_split
from tensorflow.keras.optimizers import SGD
from tensorflow.keras.models import Sequential
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.layers import Dense, Flatten, Conv2D, MaxPooling2D, Dropout, BatchNormalization, Global


from tensorflow.keras.datasets import cifar10
(xtr, ytr), (xtst, ytst) = cifar10.load_data()
xeval, xtr, yeval, ytr = train_test_split(  xtr, ytr, test_size= 1000, random_state= 42 )


# augment 25% of data and append it to train set(reform some of image to widen the range of cases that model
num_subset = 0.25 * ytr.shape[0]

rand_subset_indices = np.random.choice(ytr.shape[0], int(num_subset), replace=False)
image_subset = xtr[rand_subset_indices]
lables_subset = ytr[rand_subset_indices]


seq = iaa.Sequential([
   iaa.Fliplr(0.5), #horizental
   iaa.Crop(percent=(0, 0.1)),
   iaa.Sometimes(
      0.5,
      iaa.GaussianBlur(sigma=(0, 0.5))
   ),
   iaa.LinearContrast((0.75, 1.5)),
   iaa.AdditiveGaussianNoise(loc=0, scale=(0.0, 0.05*255), per_channel=0.5),
   iaa.Multiply((0.8, 1.2), per_channel=0.2),
   iaa.Affine(
      scale={"x": (0.8, 1.2), "y": (0.8, 1.2)},
      translate_percent={"x": (-0.2, 0.2), "y": (-0.2, 0.2)},
      rotate=(-10, 10),
      shear=(-8, 8)
   )], random_order= True)

augmented_subset = seq(images= image_subset)

#get the old images with new augmanted images (50k + 12.5k images)
xtr = np.concatenate((xtr, augmented_subset), axis= 0)
ytr = np.concatenate((ytr, lables_subset), axis= 0)


# Build Accuracy focused CNN model
model = Sequential()
model.add(Conv2D(64, (3,3), activation= 'relu', padding= 'same', input_shape= (32, 32, 3)))
model.add(Dropout(0.25))
model.add(BatchNormalization())
model.add(Conv2D(64, (3,3), activation= 'relu', padding='same'))
model.add(Dropout(0.25))
model.add(BatchNormalization())
```

```
model.add(Conv2D(64, (3,3), activation= 'relu', padding='same'))
model.add(Dropout(0.25))
model.add(BatchNormalization())
model.add(Conv2D(128, (5,5), activation= 'relu', padding='same'))
model.add(BatchNormalization())
model.add(GlobalMaxPooling2D()) #only focus on most important features (not mainly used to reduce dimensions
model.add(Flatten())
model.add(Dense(128, activation= 'relu'))
model.add(Dropout(0.25))
model.add(BatchNormalization())
model.add(Dense(10, activation= 'softmax'))

print (f"Model info: \n {model.summary()}")

#run model and get the results
model.compile(optimizer= 'adam', loss= 'sparse_categorical_crossentropy', metrics= ['accuracy'])
model.fit(xtr, ytr, epochs= 100, validation_data=(xeval, yeval))
tst_loss, tst_accuracy = model.evaluate(xtst, ytst)

#print result and save model weights
print(f"Model Performance Result(loss,accuracy): \n {tst_loss,tst_accuracy}")
model.save(r"./assig5_model_data/assig5_model.h5")
```