Міністерство освіти і науки України

НТУУ «Київський політехнічний інститут ім. Ігоря Сікорського»

Фізико-технічний інститут

Лабораторна робота №1

З дисципліни

«Автоматизація обробки ІзОД»

Варіант 6

**Виконав:**

Студент 5 курсу ФТІ

групи ФЕ-91мп

Назаров О.В.

**Перевірив:**

Прогонов Д. О.

Київ-2020

**І. Підготовка**

**Вхідні дані**

Тестовий пакет – MIRFlickr-20k (https://press.liacs.nl/mirflickr/#sec\_download)

Вибірка зображень – 250 зображень;

Формування вибірки зображень – псевдовипадкове, з використанням генератора Мерсена (стартове значення співпадає з номером студента в загальному списку групи) за модулем кількості зображень в тестовому пакеті.

**Завдання**

1. Сформувати тестову вибірку зображень з вихідного пакета;

2. Для кожного каналу кольору кожного зображення з тестового пакета обчислити наступні характеристики:

a. Максимальна / мінімальне значення;

b. Математичне сподівання і дисперсію;

c. Медіану значень, інтерквартільний розмах;

d. Коефіцієнти асиметрії та ексцесу (нормалізований);

3. Для кожного каналу кольору кожного зображення з тестового пакета побудувати гістограму значень яскравості пікселів;

4. Провести апроксимацію отриманих гістограм з використанням відомих імовірнісних розподілів, визначити найкращу апроксимацію;

5. Побудувати розподіл типів використаних імовірнісних розподілів для яких досягається мінімальне значення помилки апроксимації з п.4.

**ІI. Хід роботи**

Роботу виконуватимемо мовою Python за допомогою блокового інтерпритатора Jupyter. Також в роботі будкть використані такі бібліотеки як:

* Os
* Matplotlib
* Numpy
* Scipy
* Pandas
* Sklearn
* Seaborn

1. Генерація індексів зображень відносно варіантів

test\_index = random.sample(range(6,25001),250)

2.Створення Дата фрейму ,заповнення значеннями

a. Максимальна / мінімальне значення;

b. Математичне сподівання і дисперсію;

c. Медіану значень, інтерквартільний розмах;

d. Коефіцієнти асиметрії та ексцесу (нормалізований);

params = ['Image Number','Channel Number','Max','Min','ExpValue(UnifDist)','Variance','Median','Interquartile range','Skew',"Kurtosis",'Img']

def part1(img):

param\_list = []

for i in range(len(img)):

for Channel in range(3):

img\_ch = img[i][:,:,Channel] param\_list.append(list((test\_index[i],Channel,img\_ch.max(),img\_ch.min(),np.mean(img\_ch),np.var(img\_ch)

,np.median(img\_ch),iqr(img\_ch),skew(img\_ch,None),kurtosis(img\_ch,None),img\_ch)))

return pd.DataFrame(param\_list,columns = params)

3. Для кожного каналу кольору кожного зображення з тестового пакета будуємо побудувати гістограму значень яскравості пікселів;

plt.figure(figsize=(12,9))

plt.subplot(231),plt.hist(Data.loc[Data['Channel Number'] == 0]['Img'])

plt.subplot(232),plt.hist(Data.loc[Data['Channel Number'] == 1]['Img'])

plt.subplot(233),plt.hist(Data.loc[Data['Channel Number'] == 2]['Img'])

plt.show()

4. Прводим апроксимацію отриманих гістограм з використанням відомих імовірнісних розподілів, визначити найкращу апроксимацію;

X,Y = sns.distplot(np.ravel(hist\_data[3])).get\_lines()[0].get\_data()

test = np.ravel(test)

xt = plt.xticks()[0]

xmin, xmax = min(xt), max(xt)

lnspc = np.linspace(0, 255, len(X))

m, s = stats.norm.fit((X,Y))# get mean and standard deviation

pdf\_g = stats.norm.pdf(lnspc, m, s) # now get theoretical values in our interval

plt.plot(lnspc, pdf\_g, label="Norm",color = 'grey') # plot it

plt.plot(X,Y,color = 'black',linewidth = 4)

plt.plot(lnspc, pdf\_g, label="Norm") # plot it

ag,bg,cg = stats.gamma.fit((X,Y))

pdf\_gamma = stats.gamma.pdf(lnspc, ag, bg,cg)

plt.plot(lnspc, pdf\_gamma, label="Gamma",color = 'blue')

ab,bb,cb,db = stats.beta.fit((X,Y))

pdf\_beta = stats.beta.pdf(lnspc, ab, bb,cb, db)

plt.plot(lnspc, pdf\_beta, label="Beta",color = 'red')

plt.show()

5. Побудувати розподіл типів використаних імовірнісних розподілів для яких досягається мінімальне значення помилки апроксимації з п.4.

def fi(images):

mse1 = []

mse2 = []

mse3 = []

for i in images:

X,Y = sns.distplot(np.ravel(i)).get\_lines()[0].get\_data()

"""

for s in range(len(X)):

if 0 <= X[s] <=255:

X[s] = X[s]

else :

X[s] = 0

"""

lnspc = np.linspace(0, 255, len(X))

m, s = stats.norm.fit((X,Y))

pdf\_g = stats.norm.pdf(lnspc, m, s)

ag,bg,cg = stats.gamma.fit((X,Y))

pdf\_gamma = stats.gamma.pdf(lnspc, ag, bg,cg)

ab,bb,cb,db = stats.beta.fit((X,Y))

pdf\_beta = stats.beta.pdf(lnspc, ab, bb,cb, db)

mse1.append(sum((Y - pdf\_g)\*\*2) )#+ sum((X -lnspc)\*\*2))

mse2.append(sum((Y - pdf\_gamma)\*\*2))#+ sum((X -lnspc)\*\*2))

mse3.append(sum((Y - pdf\_beta)\*\*2))#+ sum((X -lnspc)\*\*2))

MSE\_norm = 1/250 \* sum(mse1)

MSE\_gamma = 1/250 \* sum(mse2)

MSE\_beta = 1/250 \* sum(mse3)

return MSE\_norm,MSE\_gamma,MSE\_beta

fi(Data['Img'][:25]) 