МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО»

ЗВІТ

з лабораторної роботи №4

з навчальної дисципліни «Проектування та реалізація програмних систем з

нейронними мережами»

|  |  |
| --- | --- |
| Виконав:  студент групи ІП-15  Мєшков Андрій Ігорович | Перевірив:  Шимкович В.М. |

Київ 2024

**ЛАБОРАТОРНА РОБОТА №4**

**Тема:** Згорткові нейронні мережі.

**Завдання** – Написати програму що реалізує згорткову нейронну мережу AlexNet для розпізнавання об’єктів з датасету ImageNet

**Хід роботи**

* Завантажуємо датасет

(train\_ds, test\_ds), ds\_info = tfds.load('imagenette/160px-v2', split=['train', 'validation'], as\_supervised=True, with\_info=True)

* Передобробка даних

train\_ds = train\_ds.map(lambda img, label: (tf.image.resize(img, (128, 128)), label))

test\_ds = test\_ds.map(lambda img, label: (tf.image.resize(img, (128, 128)), label))

* Завантаження класів

img\_size, img\_labels = ds\_info.features['label'].num\_classes, ds\_info.features['label'].int2str

label\_names = [img\_labels(id) *for* id *in* range(img\_size)]

print(f'Усього {img\_size} класів')

*for* i, label *in* enumerate(label\_names):

print(i, '-', label)

Усього 10 класів

0 - n01440764

1 - n02102040

2 - n02979186

3 - n03000684

4 - n03028079

5 - n03394916

6 - n03417042

7 - n03425413

8 - n03445777

9 - n03888257

class\_names = ['bench', 'English springer', 'cassette player', 'chain saw', 'church', 'French horn', 'garbage truck', 'gas pump', 'golf ball', 'parachute']

plt.figure(figsize=(10, 10))

*for* i, example *in* enumerate (test\_ds.take(9)):

image, label = example

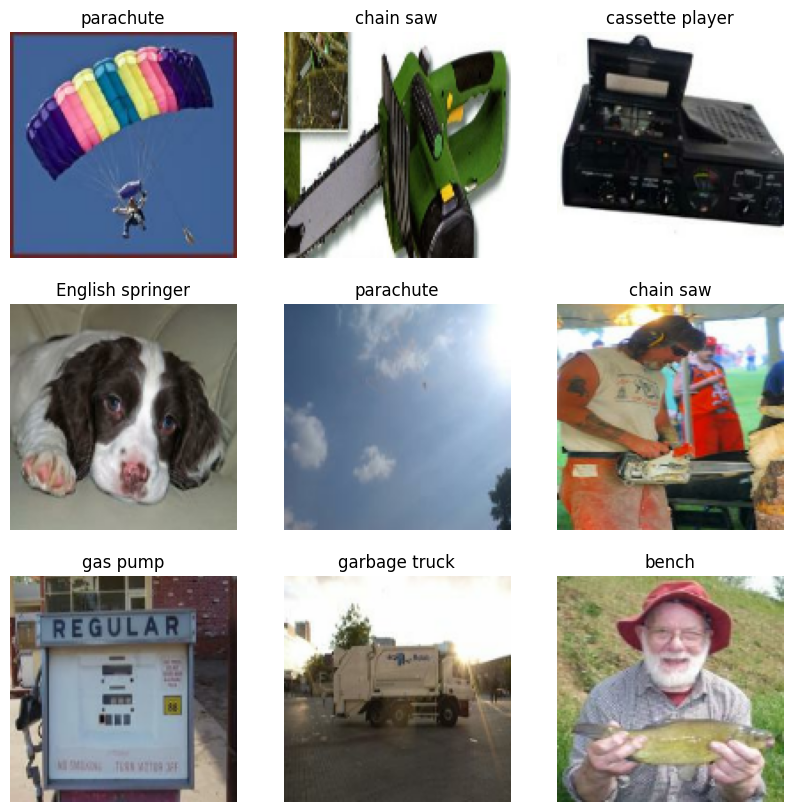
ax = plt.subplot(3, 3, i + 1)

plt.imshow(image. numpy().astype("uint8"))

plt.title(class\_names [label.numpy()])

plt.axis ("off")

plt.show()

****

* Підготовка даних

X\_train, y\_train = list(map(lambda img: img[0], train\_ds)), list(map(lambda img: img[1], train\_ds))

X\_test, y\_test = list(map(lambda img: img[0], test\_ds)), list (map (lambda img: img[1], test\_ds))

y\_train, y\_test = to\_categorical(y\_train, img\_size), to\_categorical(y\_test, img\_size)

train\_size, test\_size = ds\_info.splits['train'].num\_examples, ds\_info.splits['validation'].num\_examples

print(f'Навчальна вибірка: {train\_size}, Тестувальна вибірка {test\_size}')

Навчальна вибірка: 9469, Тестувальна вибірка 3925

train\_datagen = ImageDataGenerator(rescale=1./255, rotation\_range=40, height\_shift\_range=0.2)

test\_datagen = ImageDataGenerator(rescale=1./255)

train\_ds = NumpyArrayIterator(x=np.array(X\_train), y=np.array(y\_train), image\_data\_generator=train\_datagen, batch\_size=16)

test\_ds = NumpyArrayIterator(x=np.array(X\_test), y=np.array(y\_test), image\_data\_generator=test\_datagen, batch\_size=32)

* Опреділення моделі AlexNet

model = Sequential([

Conv2D(96, kernel\_size=(11,11), strides=4, activation='relu', input\_shape=(128,128,3), padding='valid'),

BatchNormalization(),

MaxPool2D(pool\_size=(3,3), strides=(2,2)),

Conv2D(256, kernel\_size=(5,5), padding='same', activation='relu', strides=(1,1)),

BatchNormalization(),

MaxPool2D(pool\_size=(3,3), strides=(2,2)),

Conv2D(384, kernel\_size=(3,3), padding='same', activation='relu', strides=(1,1)),

BatchNormalization(),

Conv2D(192, kernel\_size=(3,3), padding='same', activation='relu', strides=(1,1)),

BatchNormalization(),

Conv2D(256, kernel\_size=(3,3), padding='same', activation='relu', strides=(1,1)),

BatchNormalization(),

MaxPool2D(pool\_size=(3,3), strides=(2,2)),

Flatten(),

Dense(2048, activation='relu'),

Dropout(0.5),

Dense(2048, activation='relu'),

Dropout(0.5),

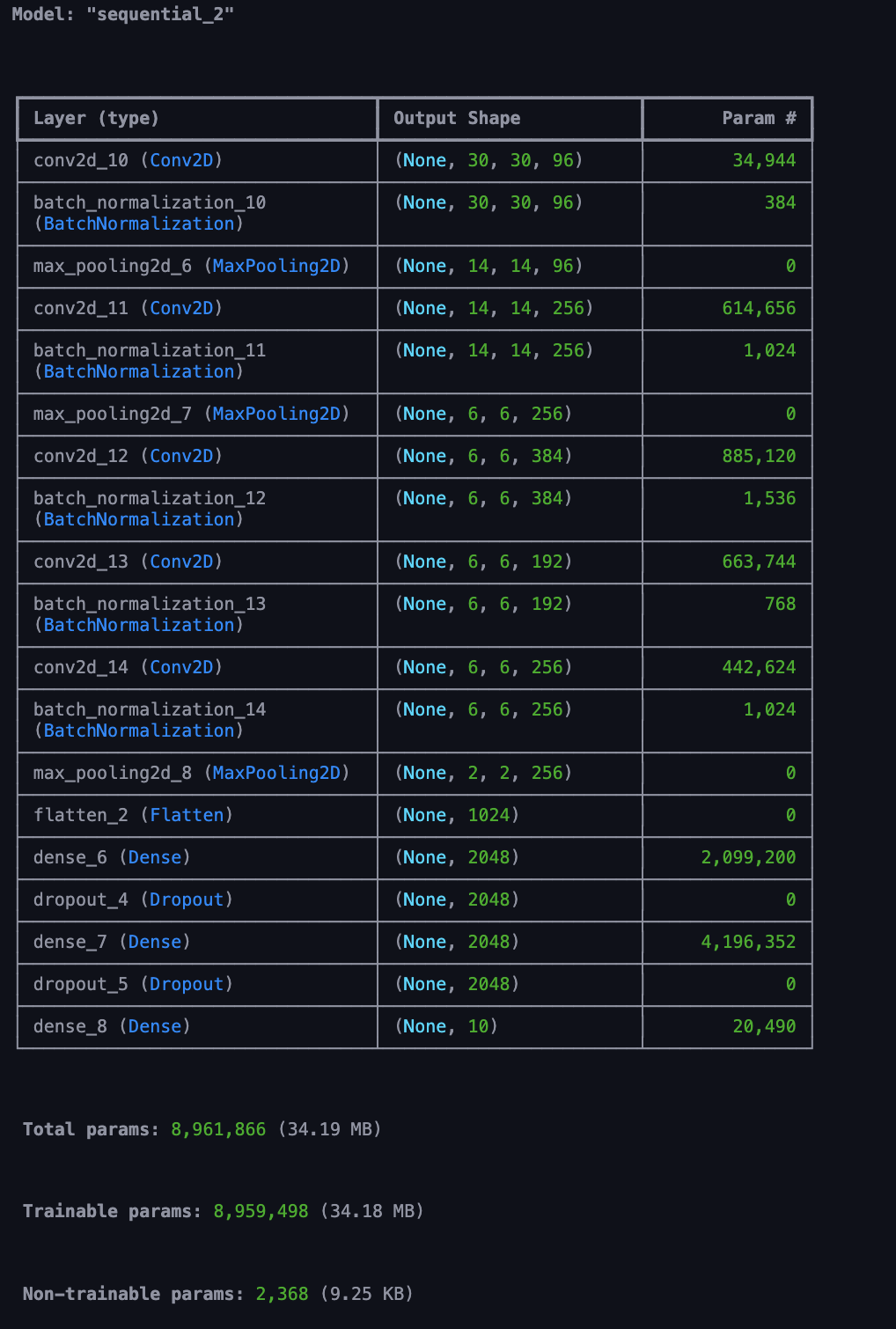
Dense(10, activation='softmax')

])

* Компіляція моделі

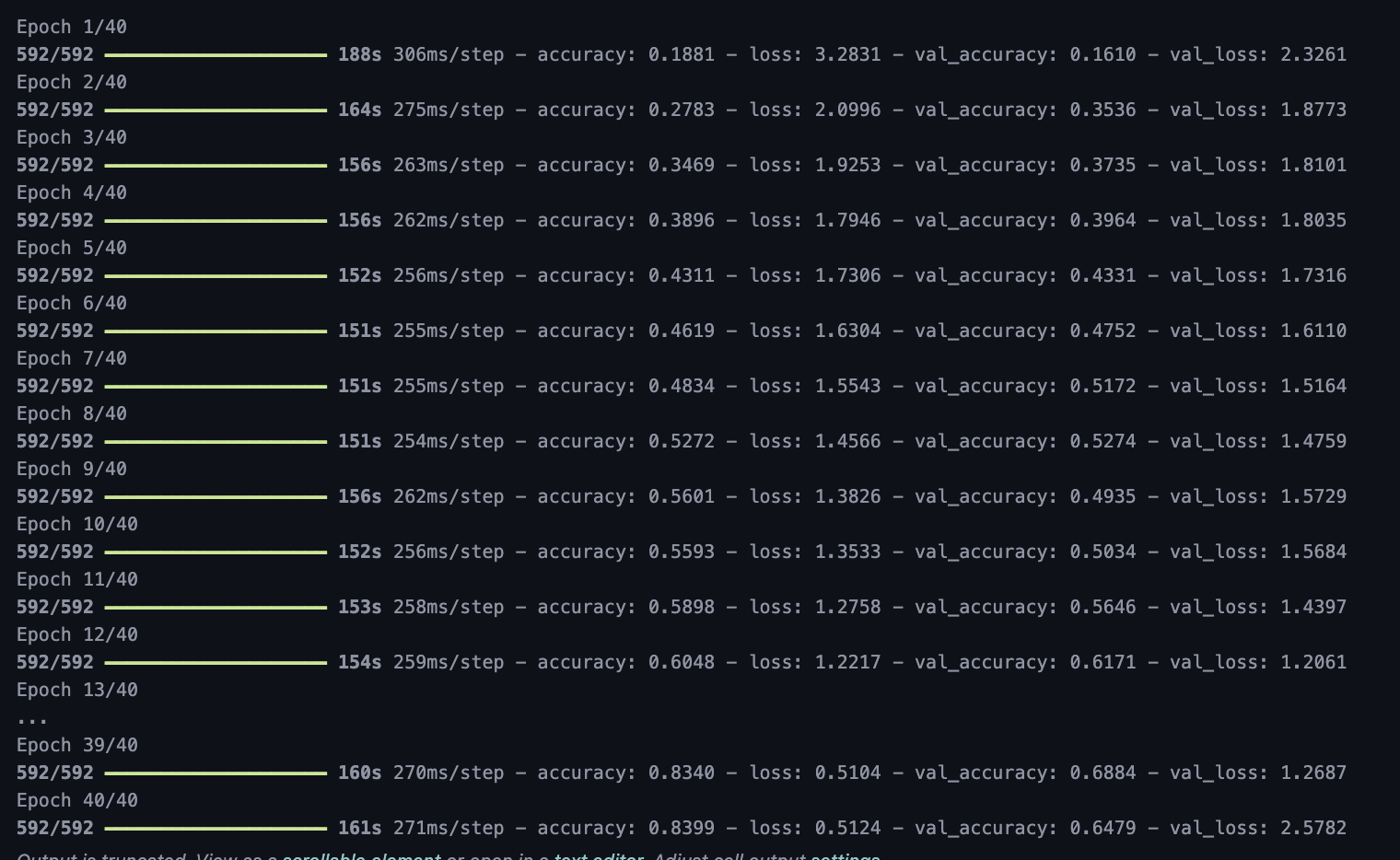
model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])

model.summary()

****

* Навчання моделі

history = model.fit(train\_ds, validation\_data=test\_ds, epochs=40)



* Вивчення результатів

plt.plot(history.history['accuracy'], label='accuracy')

plt.plot(history.history['val\_accuracy'], label = 'val\_accuracy')

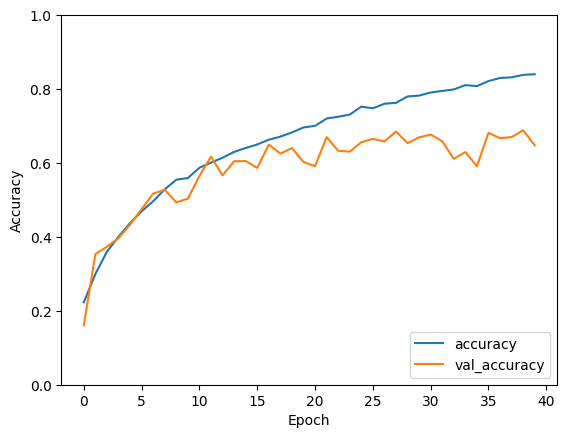
plt.xlabel('Epoch')

plt.ylabel('Accuracy')

plt.ylim([0, 1])

plt.legend(loc='lower right')

plt.show()



plt.plot(history.history['loss'], label='loss')

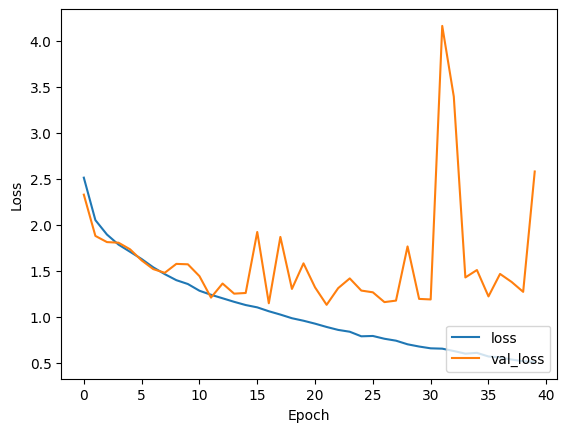
plt.plot(history.history['val\_loss'], label = 'val\_loss')

plt.xlabel('Epoch')

plt.ylabel('Loss')

plt.legend(loc='lower right')

plt.show()



* Приклад

def plot\_predictions(img\_id, predictions, ds):

img, label = ds[img\_id]

img\_pred = predictions[img\_id]

plt.figure(figsize=(6, 3))

plot\_pred\_img(img\_pred, img, label)

def plot\_pred\_img(img\_pred, img, img\_label):

plt.grid(False)

plt.xticks([])

plt.yticks([])

plt.imshow(img[0], cmap=plt.cm.binary)

pred\_label = np.argmax(img\_pred)

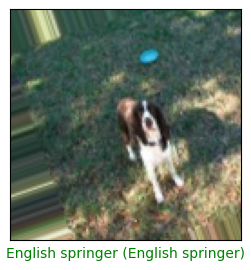
real\_label = np.argmax(img\_label)

color = 'green' *if* pred\_label == real\_label *else* 'red'

plt.xlabel(f"{class\_names[real\_label]} ({class\_names[pred\_label]})", color=color)

predictions = model.predict(test\_ds)

plot\_predictions(149, predictions, test\_ds)



**Висновок:**

Під час виконання даної лабораторної роботи було досліджено структуру та принцип роботи згорткових нейронних мереж, а саме було написано програму яка реалізує нейронну мережу AlexNet для розпізнавання зображень з датасету ImageNet. Мережу було успішно навчено розпізнавати зображеня також її було протестовано та перевірено на тестових даних.