**딥러닝/클라우드**

2차과제 레포트

32183164 이석현

Dankook University

2020 Fall

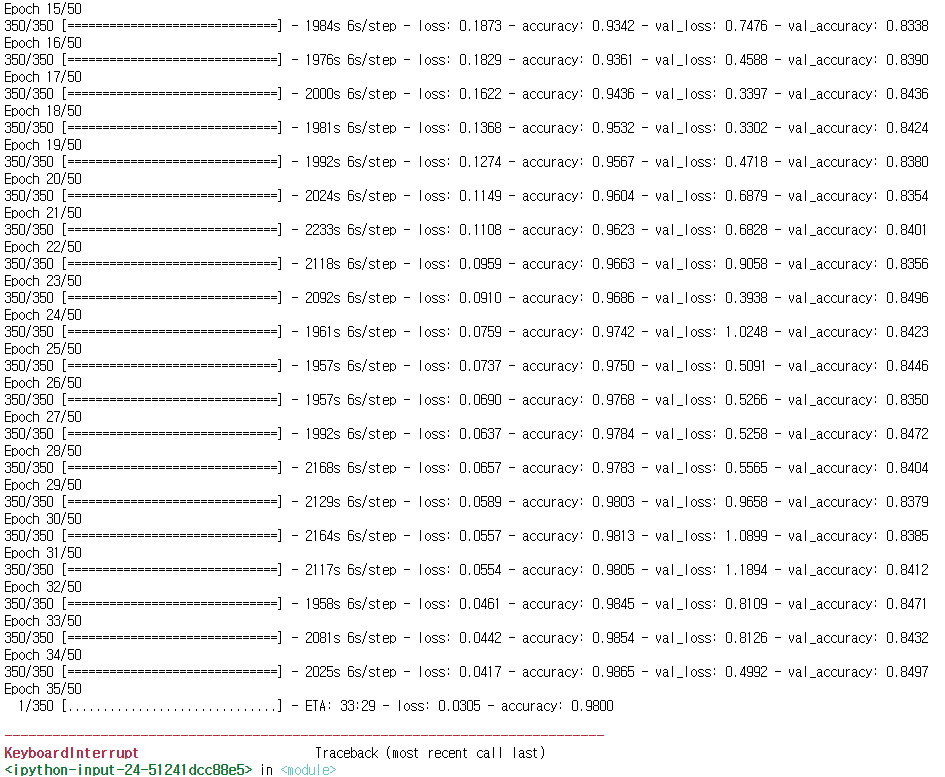
목차

1. **실행 화면 캡처 2**

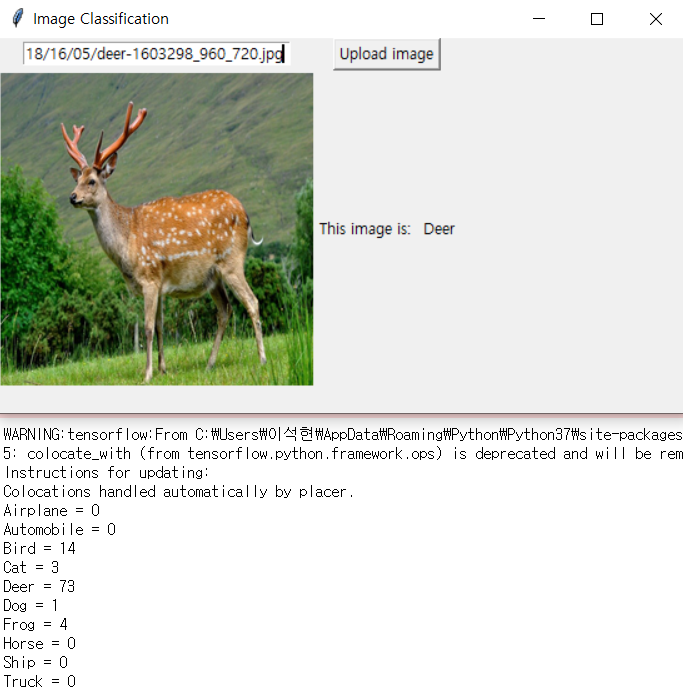
**2. 소감 4**

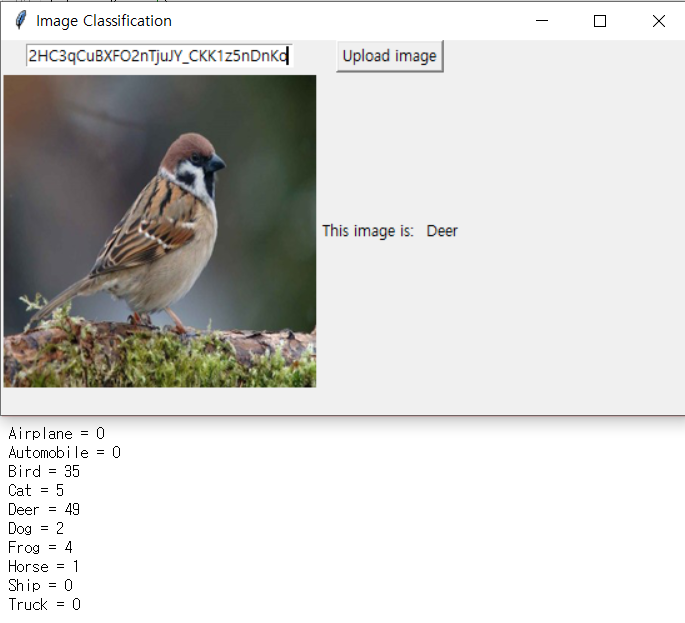
**3. 모델 개발용 소스코드 5**

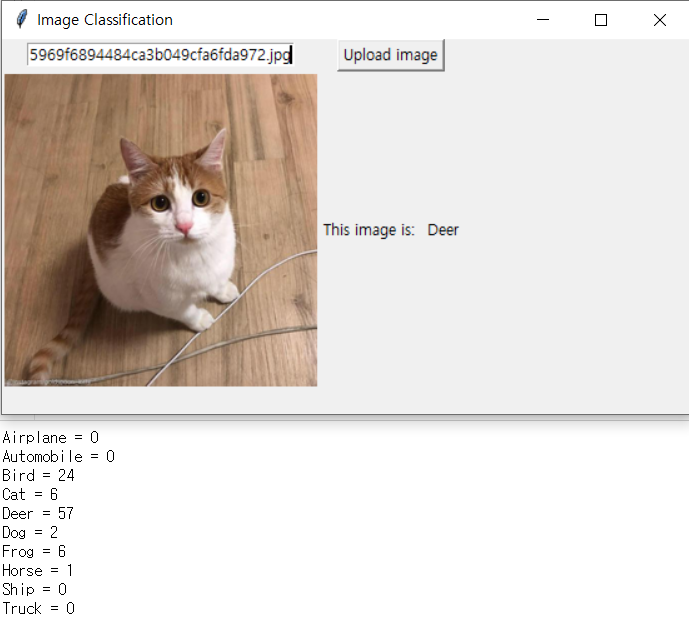
**4. 이미지 분류 SW 소스 코드 7**

**1. 실행 화면 캡처**

34번째 epoch 종료 후 keyboard interrupt 했습니다.







**2. 소감**

처음에 epoch를 50번으로 설정하고 시행하다가 34번째 epoch가 되어서야 validation accuracy가 18번째부터 줄어들고 있었다는 것을 확인하고 중단했습니다. Epoch를 17로 하고 딥러닝을 돌리면 최적의 값이 나올 것이지만 한 epoch당 한 시간이 넘게 소요되기에 시간관계상 시도하지 못하였습니다. 결과적으로 이미지 분류에서도 overfitting되어 deer의 값이 계속 높게 나오는 문제가 발생하여 아쉬움이 남습니다.

Transfer learning으로 이전에 개발되어 있는 모델을 이용해 새로운 학습 모델을 만들면서 이런 방식도 존재한다는 것에 흥미를 느꼈고, 그래서인지 재미있게 모델을 설계할 수 있었습니다. 다만 아쉬운 점이 있다면, 한 모델을 개발하는데 걸리는 러닝타임이 길어, 다양한 시도를 해보지 못한 아쉬움이 남습니다. 또한 tkinter를 이용해 GUI를 구현했는데 이 패키지를 오랜만에 사용하다 보니 낯선 부분이 있어 시간이 꽤 걸렸고, 하이퍼링크를 입력 받아 사진을 다운받고 업로드 하는데 생각보다 많은 오류가 발생해 이를 수정하는데 어려움이 있었습니다. 그래도 결국 오류를 해결해 프로그램을 동작 시킬 수 있어 뿌듯합니다.

**3. 모델 개발용 소스코드**

|  |  |
| --- | --- |
| import numpy as np  import pandas as pd  from sklearn.utils.multiclass import unique\_labels  import os  import matplotlib.pyplot as plt  import matplotlib.image as mpimg  import seaborn as sns  %matplotlib inline  import itertools  from sklearn.model\_selection import train\_test\_split  from sklearn.metrics import confusion\_matrix  from keras import Sequential  from keras.applications import VGG19  from keras.preprocessing.image import ImageDataGenerator  from keras.optimizers import SGD,Adam  from keras.callbacks import ReduceLROnPlateau  from keras.layers import Flatten,Dense,BatchNormalization,Activation,Dropout  from keras.utils import to\_categorical  from keras.datasets import cifar10  #develop model  (train\_X,train\_y),(test\_X,test\_y)=cifar10.load\_data()  train\_X,val\_X,train\_y,val\_y=train\_test\_split(train\_X,train\_y,test\_size=.3)  train\_y=to\_categorical(train\_y)  val\_y=to\_categorical(val\_y)  test\_y=to\_categorical(test\_y)  train\_generator = ImageDataGenerator(rotation\_range=2, horizontal\_flip=True, zoom\_range=.1 )  val\_generator = ImageDataGenerator(rotation\_range=2, horizontal\_flip=True, zoom\_range=.1)  test\_generator = ImageDataGenerator(rotation\_range=2, horizontal\_flip= True, zoom\_range=.1)  train\_generator.fit(train\_X)  val\_generator.fit(val\_X)  test\_generator.fit(test\_X)  lrr= ReduceLROnPlateau(monitor='val\_acc', factor=.01, patience=3, min\_lr=1e-5)  base\_model\_1 = VGG19(include\_top=False,weights='imagenet',input\_shape=(32,32,3),classes=train\_y.shape[1])  model\_1= Sequential()  model\_1.add(base\_model\_1)  model\_1.add(Flatten())  model\_1.add(Dense(1024,activation=('relu'),input\_dim=512))  model\_1.add(Dense(512,activation=('relu')))  model\_1.add(Dense(256,activation=('relu')))  #model\_1.add(Dropout(.3))  model\_1.add(Dense(128,activation=('relu')))  #model\_1.add(Dropout(.2))  model\_1.add(Dense(10,activation=('softmax')))  batch\_size= 100  epochs=50  learn\_rate=.001  sgd=SGD(lr=learn\_rate,momentum=.9,nesterov=False)  adam=Adam(lr=learn\_rate, beta\_1=0.9, beta\_2=0.999, epsilon=None, decay=0.0, amsgrad=False)  model\_1.compile(optimizer=sgd,loss='categorical\_crossentropy',metrics=['accuracy'])  model\_1.fit\_generator(train\_generator.flow(train\_X,train\_y,batch\_size=batch\_size), epochs=epochs, steps\_per\_epoch=train\_X.shape[0]//batch\_size, validation\_data=val\_generator.flow(val\_X,val\_y,batch\_size=batch\_size),validation\_steps=250, callbacks=[lrr], verbose=1)  #save model  model\_1.save('cifar10\_model.h5')  model\_json = model\_1.to\_json()  with open("model.json", "w") as json\_file :  json\_file.write(model\_json)  model\_1.save\_weights("model.h5") |  |

**4. 이미지 분류 SW 소스 코드**

|  |
| --- |
| **from tkinter import \***  **import urllib.request**  **import os**    **from PIL import ImageTk, Image**  **def img\_processing():**  **from keras.models import model\_from\_json**  **json\_file = open("model.json", "r")**  **loaded\_model\_json = json\_file.read()**  **json\_file.close()**  **loaded\_model = model\_from\_json(loaded\_model\_json)**  **loaded\_model.load\_weights("model.h5")**  **loaded\_model.compile(loss="binary\_crossentropy", optimizer="rmsprop", metrics=['accuracy'])**    **import cv2**  **im = cv2.imread('test.jpg')**    **im = cv2.cvtColor(im, cv2.COLOR\_BGR2RGB)**  **im = cv2.resize(im, (32,32))**    **in\_size = 32\*32\*3**  **import numpy as np**  **im = im.reshape(-1,32,32,3)/ 255**    **r = loaded\_model.predict(im)**    **res = r[0]**  **labels = ['Airplane', 'Automobile', 'Bird', 'Cat', 'Deer', 'Dog', 'Frog', 'Horse', 'Ship', 'Truck']**  **for i, acc in enumerate(res) :**  **print(labels[i], "=", int(acc\*100))**  **return labels[res.argmax()]**  **def download\_img():**  **url = txt.get()**  **#print(url)**  **os.system("curl " + url + " > test.jpg")**  **open\_img()**  **def open\_img():**  **x = 'C:/Users/이석현/딥러닝/test.jpg'**  **img = Image.open(x)**  **img = img.resize((250, 250), Image.ANTIALIAS)**  **img = ImageTk.PhotoImage(img)**  **panel = Label(root, image = img)**  **panel.image = img**  **panel.grid(row = 2)**    **result = img\_processing()**  **lbl3 = Label(root, text=result)**  **lbl3.grid(row=2,column=2)**    **root = Tk()**  **root.title("Image Classification")**  **root.geometry("550x300+300+150")**  **root.resizable(width = True, height = True)**  **txt = Entry(root, width=30)**  **txt.grid(row=1)**  **btn = Button(root, text ='Upload image', command = download\_img).grid(row = 1, column=1, columnspan = 4)**  **lbl2 = Label(root, text="This image is: ")**  **lbl2.grid(row=2,column=1)**  **root.mainloop()** |