|  |  |
| --- | --- |
| 교육 제목 | **딥러닝 cnn을 활용한 이미지 분류 연습** |
| 교육 일시 | 2021년 12월 9일 목요일 |
| 교육 장소 | 영우글로벌러닝 2층 |
| **교육 내용** | |
| 오전 | 1. 이미지 분류 2. Kaggle Dogs Vs. Cats 데이터셋 준비하기    1. !wget --no-check-certificate \    2. https://storage.googleapis.com/mledu-datasets/cats\_and\_dogs\_filtered.zip \    3. -O /tmp/cats\_and\_dogs\_filtered.zip    4. 리눅스 기본명령어 - wget 'Web Get'의 약어로 웹 상의 파일을 다운로드 받을 때 사용하는 명령어 3. Kaggle Dogs Vs. Cats 데이터셋 살펴보기    1. 이미지 확인 코드    2. import matplotlib.image as mpimg    3. import matplotlib.pyplot as plt    4. nrows, ncols = 4, 4    5. pic\_index = 0    6. fig = plt.gcf()    7. fig.set\_size\_inches(ncols\*3, nrows\*3)    8. pic\_index+=8    9. next\_cat\_pix = [os.path.join(train\_cats\_dir, fname)    10. for fname in train\_cat\_fnames[ pic\_index-8:pic\_index]]    11. next\_dog\_pix = [os.path.join(train\_dogs\_dir, fname)    12. for fname in train\_dog\_fnames[ pic\_index-8:pic\_index]]    13. for i, img\_path in enumerate(next\_cat\_pix+next\_dog\_pix):    14. sp = plt.subplot(nrows, ncols, i + 1)    15. sp.axis('Off')    16. img = mpimg.imread(img\_path)    17. plt.imshow(img)    18. plt.show() 4. 모델 구성하기    1. import tensorflow as tf    2. model = tf.keras.models.Sequential([    3. tf.keras.layers.Conv2D(16, (3,3), activation='relu', input\_shape=(150, 150, 3)),    4. tf.keras.layers.MaxPooling2D(2,2),    5. tf.keras.layers.Conv2D(32, (3,3), activation='relu'),    6. tf.keras.layers.MaxPooling2D(2,2),    7. tf.keras.layers.Conv2D(64, (3,3), activation='relu'),    8. tf.keras.layers.MaxPooling2D(2,2),    9. tf.keras.layers.Flatten(),    10. tf.keras.layers.Dense(512, activation='relu'),    11. tf.keras.layers.Dense(1, activation='sigmoid')    12. ])    13. model.summary() 5. 모델 컴파일하기    1. from tensorflow.keras.optimizers import RMSprop    2. model.compile(optimizer=RMSprop(lr=0.001),    3. loss='binary\_crossentropy',    4. metrics = ['accuracy'])   . |
| 오후 | 1. 이미지 데이터 전처리하기    1. from tensorflow.keras.preprocessing.image import ImageDataGenerator    2. train\_datagen = ImageDataGenerator( rescale = 1.0/255. )    3. test\_datagen = ImageDataGenerator( rescale = 1.0/255. )    4. train\_generator = train\_datagen.flow\_from\_directory(train\_dir,    5. batch\_size=20, class\_mode='binary', target\_size=(150, 150))    6. validation\_generator = test\_datagen.flow\_from\_directory(validation\_dir, batch\_size=20, class\_mode = 'binary', target\_size = (150, 150)) 2. 모델 훈련하기    1. history = model.fit(train\_generator, validation\_data=validation\_generator,    2. steps\_per\_epoch=100, epochs=100,    3. validation\_steps=50, verbose=2) 3. 정확도와 손실 확인하기    1. import matplotlib.pyplot as plt    2. acc = history.history['accuracy']    3. val\_acc = history.history['val\_accuracy']    4. loss = history.history['loss']    5. val\_loss = history.history['val\_loss']    6. epochs = range(len(acc))    7. plt.plot(epochs, acc, 'bo', label='Training accuracy')    8. plt.plot(epochs, val\_acc, 'b', label='Validation accuracy')    9. plt.title('Training and validation accuracy')    10. plt.legend()    11. plt.figure()    12. plt.plot(epochs, loss, 'go', label='Training Loss')    13. plt.plot(epochs, val\_loss, 'g', label='Validation Loss')    14. plt.title('Training and validation loss')    15. plt.legend()    16. plt.show() 4. 테스트 이미지 분류하기    1. import numpy as np    2. from google.colab import files    3. from keras.preprocessing import image    4. uploaded=files.upload()    5. for fn in uploaded.keys():    6. path='/content/' + fn    7. img=image.load\_img(path, target\_size=(150, 150))    8. x=image.img\_to\_array(img)    9. x=np.expand\_dims(x, axis=0)    10. images = np.vstack([x])    11. classes = model.predict(images, batch\_size=10)    12. print(classes[0])    13. if classes[0]>0:    14. print(fn + " is a dog")    15. else:    16. print(fn + " is a cat") |