DL Personal Project [곽용하 2014121047]

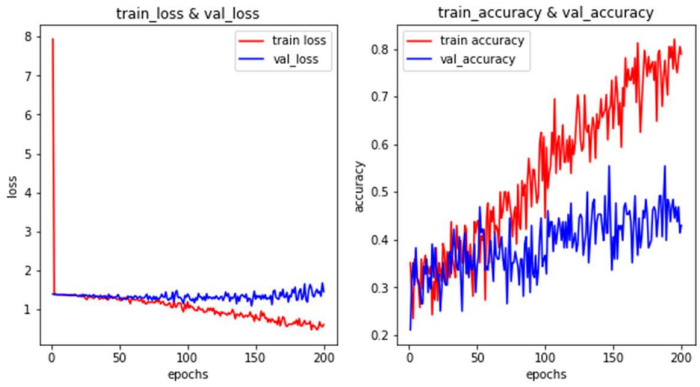
(1) How to collect your data sets (where did you download your data sets/or how to simulate your data if you are doing simulation studies).

1st step: 구글 이미지 크롤링을 통해 각 인물 사진을 다운로드합니다.

2nd step: OpenCV 를 활용하여 얼굴 부분만 cropping 합니다.

3rd step: 모든 사진이 균일한 사이즈를 갖도록 변경합니다.

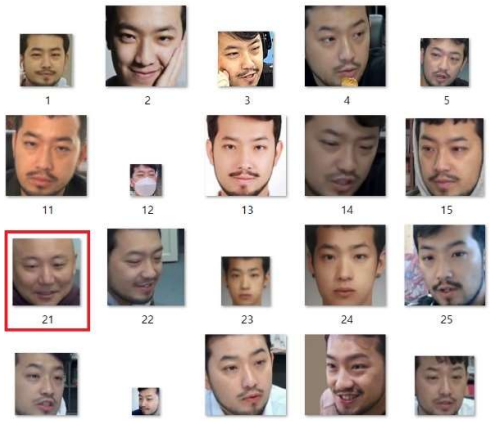
처음 모델을 구성하고 학습시킬 때에는 1st step에서 취합한 자료만을 이용했는데, 아래의 그림처럼 overfitting 문제가 해결되지 않았습니다. 오히려 validation loss가 상승하는 모습을 보였습니다. 이는 hidden layer을 아무리 많이 추가해도, dropout 비율을 아무리 높여도 이와 같은 현상은 사라지지 않았습니다.



그 이유를 분석한 결과 이미지 내에 얼굴 외에도 사진을 분별하는데 특징으로 분류될 수 있는 영역이 존재하는 것이 가장 주된 원인이 될 것이라 판단했습니다. 예를 들면 아래 사진처럼,

같은 침착맨이더라도, 배경이 다를 경우 이를 다른 범주로 분류하는 오류를 발생할 수 있는 것입니다.

따라서 2nd step을 통해서 분류학습을 보다 명확히 할 수 있도록 하는 것이 필요했습니다. 해당 절차를 거치면 아래와 같은 결과를 얻습니다.



그럼에도 21번 사진처럼 분류를 하는데 있어 혼란을 주는 요소들이 남아있어, 직접 확인하여 해당 사진을 제거했습니다.

3rd step: 이미지 파일의 크기가 제각각 이므로, 균일한 사이즈 이하가 될 수 있도록 조정했습니다.

각 단계에서 사용한 코드는 다음과 같습니다.

[1st step]

In [3]:
```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import time
import urllib.request
import os
```

In [5]:
```
def createDirectory(directory):
    try:
        if not os.path.exists(directory):
            os.makedirs(directory)
    except OSError:
        print("Error: Failed to create the directory.")
```

In [7]:
```
def crawling_img(name):
    driver = webdriver.Chrome("C:/Users/david/OneDrive/바탕 화면/DL_final/chromedriver.exe")
    driver.get("https://www.google.co.kr/imghp?hl=ko&tab=ri&authuser=0&ogbl")
    elem = driver.find_element_by_name("q")
    elem.send_keys(name)
    elem.send_keys(Keys.RETURN)

    #
    SCROLL_PAUSE_TIME = 1
    # Get scroll height
    last_height = driver.execute_script("return document.body.scrollHeight")  # 브라우저의 높이를 자바스크립트로 찾음
    while True:
        # Scroll down to bottom
        driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")  # 브라우저 끝까지 스크롤을 내림
        # Wait to load page
        time.sleep(SCROLL_PAUSE_TIME)
        # Calculate new scroll height and compare with last scroll height
        new_height = driver.execute_script("return document.body.scrollHeight")
        if new_height == last_height:
            try:
                driver.find_element_by_css_selector(".mye4qd").click()
            except:
                break
        last_height = new_height

    imgs = driver.find_elements_by_css_selector(".rg_i.Q4LuWd")
    dir = "./idols" + "/" + name
```

```
    createDirectory(dir) #폴더 생성
    count = 1
    for img in imgs:
        try:
            img.click()
            time.sleep(2)
            imgUrl = driver.find_element_by_xpath('/html/body/div[2]/c-wiz/div[3]/div[2]/div[3]/div/div/div[3]/div[2]/c-wiz/div/di
                "src")
            path = "C:/Users/david/Downloads/idols/" + name + "/"
            urllib.request.urlretrieve(imgUrl, path + "PP" + str(count) + ".png")
            count = count + 1
            if count >= 50:
                break
        except:
            pass
    driver.close()
```

In [8]:
```
idols = ["김풍"]

for idol in idols:
    crawling_img(idol)
```

[2nd step]

In [2]:
```
pip install opencv-python
```

```
...
```

In [1]:
```
import cv2
import glob
import os
import numpy as np
```

In [9]:
```
face_cascade = cv2.CascadeClassifier("C:/Users/david/anaconda3/Lib/site-packages/cv2/data/haarcascade_frontalface_alt.xml")
eye_cascade = cv2.CascadeClassifier("C:/Users/david/anaconda3/Lib/site-packages/cv2/data/haarcascade_eye.xml")
```

In [25]:
```
images = sorted(glob.glob('C:/Users/david/Downloads/idols/JHM/*.png'), key=os.path.getctime)
for fname in images:
    pass
images

len(images)
```

Out[25]: 295

In [26]:
```
for i in range(1,len(images)):
    img = cv2.imread(images[i-1])
    faces = face_cascade.detectMultiScale(img, 1.1,4)
    imgNum = 1
    for (x,y,w,h) in faces:
        cropped = img[y:y + h, x:x + w]
        cv2.imwrite(str(i) + str(imgNum) + ".png", cropped)
        imgNum += 1
```

[3rd step]

파일 사이즈 일괄 조정하기

In [39]:
```
files = glob.glob("C:/Users/david/Downloads/cpp/JHM/*")

for f in files:
    title, ext = os.path.splitext(f)
    if ext in ['.png']:
        img = Image.open(f)
        img_resize = img.resize((128, 128))
        img_resize.save(title+ext)
```

(2) What kind of methods (algorithms) did you applied. You should try various directions to provide a better fitting. (e.g. changing number of hidden layers, tuning parameter settings, and so on.)

Convolutional Neural Network를 사용했습니다. 기존 과제였던 '세포의 감염 여부 binary classification'의 경우 데이터도 풍부하고 차이점이 확연하고 간결하여 그다지 복잡한 모델이 필요하지 않다는 생각이 들었습니다. 이에 사람 얼굴의 차이를 CNN을 통해서도 구분할 수 있는지 확인하고 싶었고, 이번 개인 과제에서는 이말년, 주호민, 김풍, 그리고 기안84 네 명의 사람을 정확히 구분할 수 있는 모델을 성공적으로 만드는 것을 목표로 하게 되었습니다.

모델링 코드와 결과물은 아래와 같습니다.

CNN Modeling

In [1]:
```
import tensorflow as tf
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Conv2D,MaxPool2D,Dropout,Flatten,Dense,BatchNormalization
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import EarlyStopping

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
# Dividing Dataset into two folders train and test
width = 128
height = 128
datagen = ImageDataGenerator(rescale = 1/255.0, horizontal_flip=True, zoom_range=0.2, rotation_range=0.1, width_shift_range=0.2,
                             height_shift_range = 0.2, shear_range = 0.2,
                             fill_mode='nearest', validation_split=0.2)

trainDatagen = datagen.flow_from_directory(directory='C:/Users/david/Downloads/ccp/',
                                           target_size=(width,height),
                                           class_mode = 'categorical',
                                           batch_size = 16,
                                           subset='training')

valDatagen = datagen.flow_from_directory(directory='C:/Users/david/Downloads/ccp/',
                                           target_size=(width,height),
                                           class_mode = 'categorical',
                                           batch_size = 16,
                                           subset='validation')
```

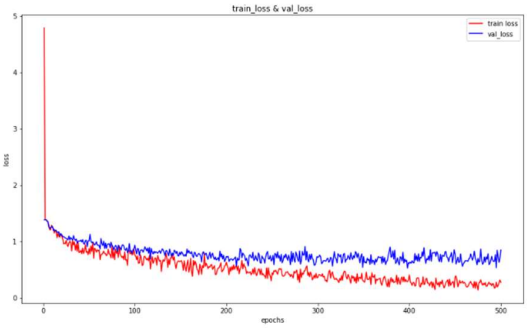
```
Found 759 images belonging to 4 classes.
Found 187 images belonging to 4 classes.
```

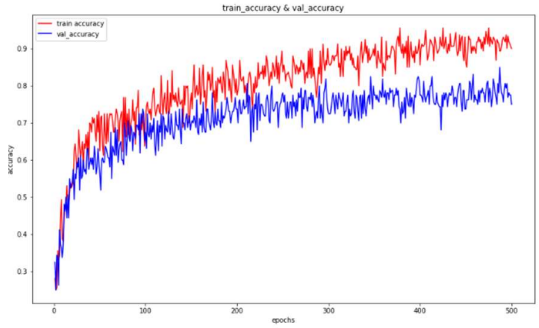
In [3]:
```
# Model
#1
model = Sequential()
model.add(Conv2D(16, (3, 3), activation='relu',padding='same',input_shape = (width,height,3)))
model.add(MaxPool2D(pool_size=(2, 2),strides=2,padding='same'))
model.add(Dropout(0.2))
#2
model.add(Conv2D(32, (3, 3), activation='relu',padding='same'))
model.add(MaxPool2D(pool_size=(2, 2),strides=2,padding='same'))
model.add(Dropout(0.2))
#3
model.add(Conv2D(64, (3, 3), activation='relu',padding='same'))
model.add(MaxPool2D(pool_size=(2, 2),strides=2,padding='same'))
model.add(Dropout(0.2))
#5
model.add(Flatten())
model.add(Dense(1024, activation='relu'))
model.add(Dense(4, activation='softmax'))
```

In [4]:
```
#model compile & fit
model.compile(optimizer='adam',
              loss = 'categorical_crossentropy',
              metrics=['accuracy'])

history = model.fit(trainDatagen,
                    steps_per_epoch=10,
                    epochs=500,
                    verbose=1,
                    validation_data = valDatagen,
                    validation_steps=10)
```

```
7937
Epoch 495/500
10/10 [==============================] - 3s 295ms/step - loss: 0.2752 - accuracy: 0.9007 - val_loss: 0.5295 - val_accuracy: 0.
8062
Epoch 496/500
10/10 [==============================] - 3s 294ms/step - loss: 0.1907 - accuracy: 0.9338 - val_loss: 0.8425 - val_accuracy: 0.
7688
Epoch 497/500
10/10 [==============================] - 3s 290ms/step - loss: 0.2160 - accuracy: 0.9187 - val_loss: 0.7209 - val_accuracy: 0.
7812
Epoch 498/500
10/10 [==============================] - 3s 293ms/step - loss: 0.2689 - accuracy: 0.9139 - val_loss: 0.6253 - val_accuracy: 0.
7750
Epoch 499/500
10/10 [==============================] - 3s 287ms/step - loss: 0.3143 - accuracy: 0.9062 - val_loss: 0.6936 - val_accuracy: 0.
7750
Epoch 500/500
10/10 [==============================] - 3s 295ms/step - loss: 0.2710 - accuracy: 0.9007 - val_loss: 0.8460 - val_accuracy: 0.
7500
```





(3) What is the conclusion based on your analysis.

구글 이미지 검색에서 찾을 수 없는 유튜브 화면을 프린트-스크린하여 실험했습니다.

In [6]:
```
import keras
```

In [7]:
```
img = keras.preprocessing.image.load_img(
    path = "C:/Users/david/OneDrive/바탕 화면/test/chim3.png", target_size=(height, width)
)
```

In [8]:
```
img_array = keras.preprocessing.image.img_to_array(img)
img_array1 = tf.expand_dims(img_array, 0) # Create a batch
```

In [9]:
```
predictions = model.predict(img_array1)
score = tf.nn.softmax(predictions[0])
```

In [10]:
```
img
```

Out[10]:

In [11]:
```
np.argmax(score), trainDatagen.class_indices
```

Out[11]:
```
(1, {'84': 0, 'CM': 1, 'JHM': 2, 'KP': 3})
```

인물 4명의 얼굴 분류를 잘 하고 있습니다. 특히 굳이 layer을 많이 설정하지 않더라도, 그리고 dropout을 지나치게 높게 설정하지 않을 경우 상대적으로 높은 정확도를 보였습니다.

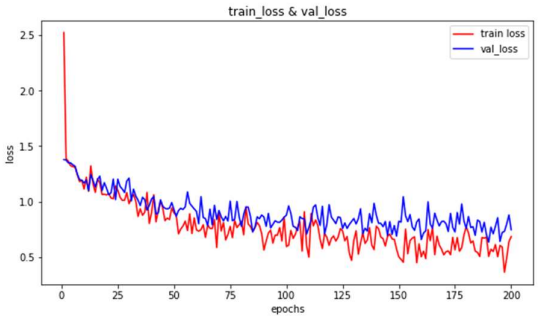
다만, 각 인물 폴더의 이미지 개수가 300여 개씩 정도로 충분하다고는 할 수 없는 수준입니다. Validation accuracy가 75%~80% 사이에서 형성되는데, 이는 데이터의 양 자체를 늘리거나, 분류 기준을 명확히 하는데 도움이 되는 질 높은 데이터를 더 수집하는 방법을 통해 해결할 수 있을 것 같습니다.

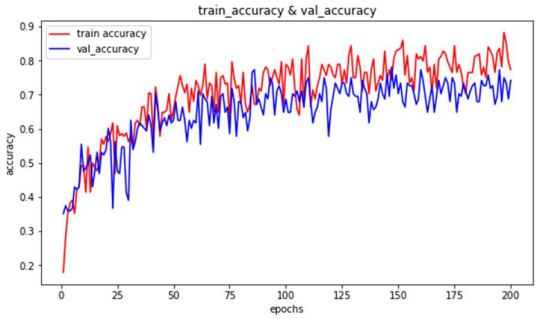
이번 개인 자유과제를 수행하며, 데이터 수집 자체의 중요성, 그리고 수집된 데이터를 분석에 맞게 가공하는 작업이 선행되어야 CNN 등 신경망 모형이 효과적으로 작동한다는 것을 깨닫게 되었습니다.

첨부) 모델을 다양한 방식으로 수정하며 나온 결과들 중 일부입니다.

a) Convolution layer을 두 겹 쌓은 후 max pooling 할 경우

```
# Model
#1
model = Sequential()
model.add(Conv2D(16, (3, 3), activation='relu',padding='same',input_shape = (width,height,3)))
model.add(MaxPool2D(pool_size=(2, 2),strides=2,padding='same'))
model.add(Dropout(0.2))
#2
model.add(Conv2D(32, (3, 3), activation='relu',padding='same'))
model.add(Conv2D(32, (3, 3), activation='relu',padding='same'))
model.add(MaxPool2D(pool_size=(2, 2),strides=2,padding='same'))
model.add(Dropout(0.2))
#3
model.add(Conv2D(64, (3, 3), activation='relu',padding='same'))
model.add(Conv2D(64, (3, 3), activation='relu',padding='same'))
model.add(MaxPool2D(pool_size=(2, 2),strides=2,padding='same'))
model.add(Dropout(0.2))
#4
model.add(Conv2D(128, (3, 3), activation='relu',padding='same'))
model.add(MaxPool2D(pool_size=(2, 2),strides=2,padding='same'))
model.add(Dropout(0.2))
#5
model.add(Flatten())
model.add(Dense(1024, activation='relu'))
model.add(Dense(4, activation='softmax'))
```





b) Hidden layer을 4개로 할 경우