# Passport Photos

이중분류와 다중분류를 사용하여 여권사진 분류하기

11조

2019240026 김서현, 2018150030 김유경, 2019190241 주민, 2017130533 박희원, 2015170063 오원석

01 **주제 소개**
인공지능을 활용하여 여권사진 분류하기

02 **다중분류 모델**
여권사진이 어떤 기준을 위반하였는지 찾아내기

03 **이중분류 후 다중분류 모델**
잘못된 여권사진 분류 후 위반한 기준에 따라 분류하기

04 **결론**
개선할 점과 아쉬운 점
현실에서 프로젝트 이용가능성 소감

# 주제 소개

# 01

# 인공지능을 학습시켜 여권사진 이미지를 분류

여권사진이 규정에
부합하는가

**이중분류 모델**

여권사진이 어떤 기준을
위반하였는가

**다중분류 모델**

# 다중분류 모델

# 02

# 여권사진이 어떤 기준을 위반하였는지 찾아내기

Google Colab에
다중분류 코드
작성

**모델 준비**

**새로운 이미지
테스트**

**데이터 수집**

구글링을 통해 찾은 여권사진을 편집하여
5개의 규정을 위반한 여권사진 각 120장 준비.
Train 파일 100개와 Test 파일 20개로 나눔.

**인공지능 학습**

**결과 도출**

# Data set 구성

*사진의 크기는 413X531로 통일

| 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
|  |  |  |  |  |
| background | light | location | quality | size |
| 배경이 흰색이 아닌 이미지 | 지나치게 밝거나 어두운 이미지 | 얼굴의 위치가 정중앙이 아닌 이미지 | 화질이 좋지 않은 이미지 | 얼굴 크기가 크거나 작은 이미지 |

# 기준 모델

# 다중분류 기준 모델

```
#1. 구글드라이브 연동
from google.colab import drive
drive.mount('/content/drive')


#2. zip 파일 압축 풀기
import zipfile
with zipfile.ZipFile('/content/drive/My Drive/증명사진/final_data_set.zip','r')as zip_ref:
  zip_ref.extractall('/content/drive/My Drive/증명사진/final_data_set')
```

# 다중분류 기준 모델

```
#3. Data set 불러오기
from tensorflow import keras
#훈련셋
train_datagen=keras.preprocessing.image.ImageDataGenerator(rescale=1/255)
train_generator=train_datagen.flow_from_directory('/content/drive/My Drive/자료/data_set/
train',target_size=(28,28),batch_size=10,class_mode='categorical')
#평가셋
test_datagen=keras.preprocessing.image.ImageDataGenerator(rescale=1/255)
test_generator=test_datagen.flow_from_directory('/content/drive/My Drive/자료/data_set/t
est',target_size=(28,28),batch_size=10,class_mode='categorical')

print(train_generator.image_shape)
print(train_generator.class_indices)

#3-Data set 결과
Found 500 images belonging to 5 classes. Found 100 images belonging to 5 classes.
(28, 28, 3) {'background': 0, 'light': 1, 'location': 2, 'quality': 3, 'size': 4}
```

# 다중분류 기준 모델

```
#4. data 살펴보기
print(test_generator[0][0].shape)

#4-data 결과
(10, 28, 28, 3)
```

# 다중분류 기준 모델

```
#5. 모델 생성: cnn
from tensorflow import keras
from keras.layers import Dense, Conv2D, MaxPooling2D, Flatten
model=keras.Sequential()
model.add(Conv2D(filters=32, kernel_size=(3,3),activation='relu',
                 input_shape=test_generator.image_shape))
model.add(Conv2D(32,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(filters=64, kernel_size=(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(filters=64, kernel_size=(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Flatten())
model.add(Dense(units=32,activation='relu'))
model.add(Dense(5,activation='softmax'))
model.summary()
```

# 다중분류 기준 모델

#5-CNN 결과

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 26, 26, 32)        896
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 24, 24, 32)        9248
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 12, 12, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 10, 10, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 5, 5, 64)          0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 3, 3, 64)          36928
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 1, 1, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 64)                0
_________________________________________________________________
dense (Dense)                (None, 32)                2080
_________________________________________________________________
dense_1 (Dense)              (None, 5)                 165
=================================================================
Total params: 67,813
Trainable params: 67,813
Non-trainable params: 0
_________________________________________________________________
```

# 다중분류 기준 모델

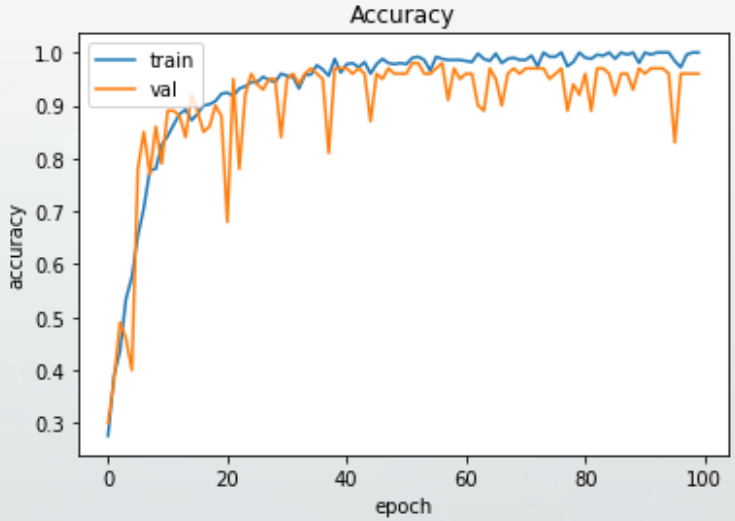
```
#6. COMPILE
model.compile(optimizer='sgd',loss='categorical_crossentropy',metrics=['accuracy'])


#7.모델 학습시키기
epochs=100
history=model.fit(train_generator,steps_per_epoch=50,
                  epochs=epochs,validation_data=test_generator,validation_steps=10)

#8. 모델 평가하기
result=model.evaluate(test_generator,steps=1)

#8-평가 결과
loss: 0.0429
accuracy: 1.0000
```

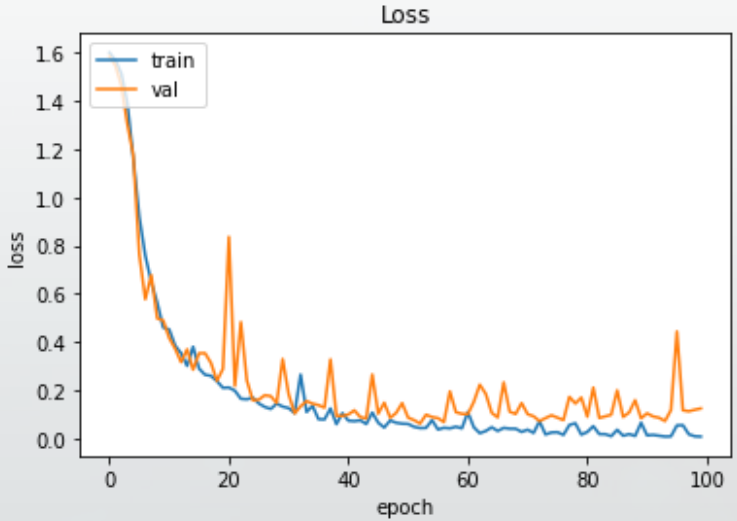
# 다중분류 기준 모델

```
#9. 시각화
import matplotlib.pyplot as plt
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc = 'upper left')
plt.show()
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])

plt.title('Loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc = 'upper left')
plt.show()
```

# 다중분류 기준 모델

#9-시각화 결과



# 모델의 정확도를 높이기 위한 노력

1. 기준모델 + dropout
2. 기준모델 + Epoch 50 + dropout
3. 기준모델 + Optimizer Adam
4. 기준모델 + Optimizer Adam + Epoch 50

기준모델 + dropout

# 1번째 시도

# 1. 기준모델 + dropout

#5. 모델구성

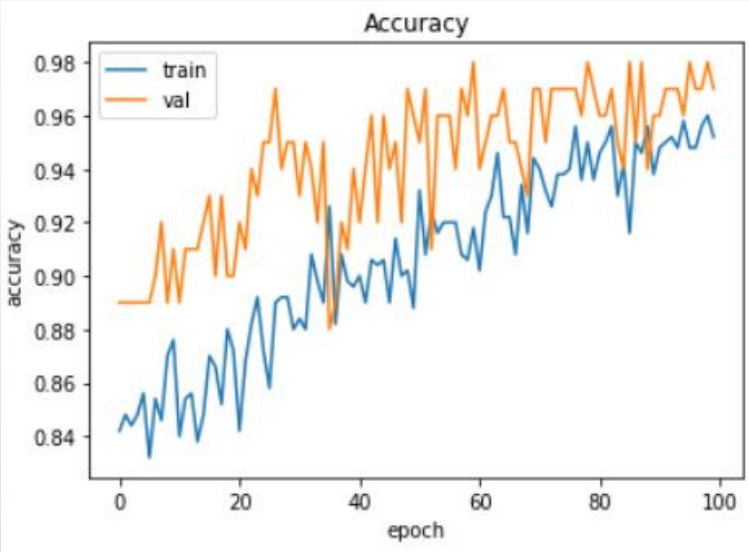
```
from tensorflow import keras
from keras.layers import Dense, Conv2D, MaxPooling2D, Flatten, Dropout

model=keras.Sequential()
model.add(Conv2D(filters=32, kernel_size=(3,3),activation='relu',input_shape=test_generator.image_shape))
model.add(Conv2D(32,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(keras.layers.Dropout(0.5))
model.add(Conv2D(filters=64, kernel_size=(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))


model.add(Conv2D(filters=64, kernel_size=(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(keras.layers.Dropout(0.5))
model.add(Flatten())
model.add(Dense(units=32,activation='relu'))
model.add(Dense(5,activation='softmax'))

model.summary()
```

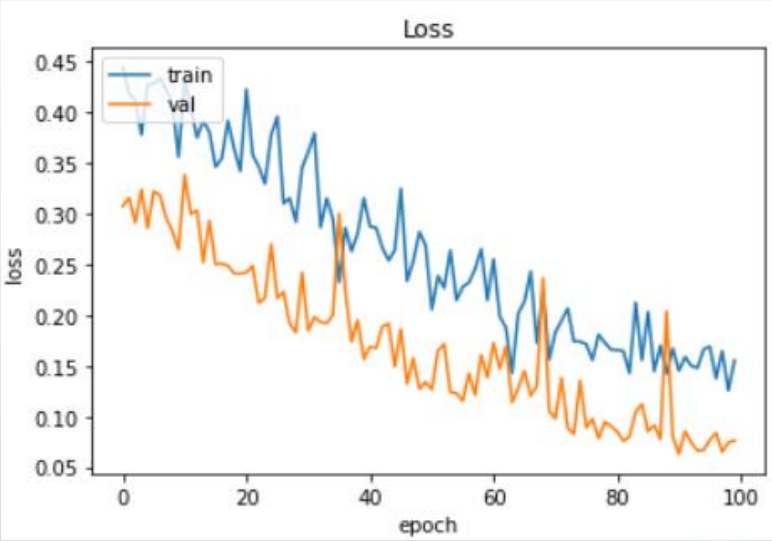
# 1. 기준모델 + dropout

#9-시각화 결과

accuracy: 1.0000

loss: 0.0696



기준모델 + Epoch 50 + dropout

# 2번째 시도

# 1. 기준모델 + dropout

#5. 모델구성

```
from tensorflow import keras
from keras.layers import Dense, Conv2D, MaxPooling2D, Flatten, Dropout

model=keras.Sequential()
model.add(Conv2D(filters=32, kernel_size=(3,3),activation='relu',input_shape=test_generator.image_shape))
model.add(Conv2D(32,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(keras.layers.Dropout(0.5))
model.add(Conv2D(filters=64, kernel_size=(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))


model.add(Conv2D(filters=64, kernel_size=(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(keras.layers.Dropout(0.5))
model.add(Flatten())
model.add(Dense(units=32,activation='relu'))
model.add(Dense(5,activation='softmax'))

model.summary()
```

# 1. 기준모델 + dropout

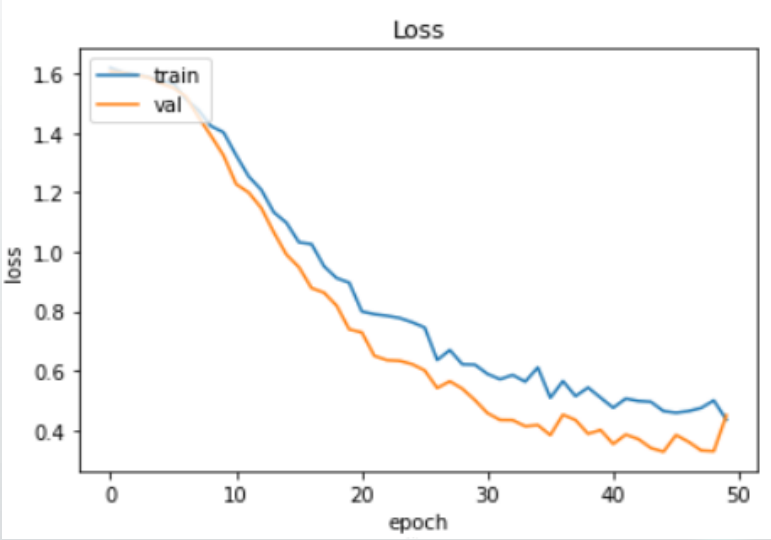
```
#7.모델 학습시키기
epochs=50

history=model.fit(train_generator,steps_per_epoch=50,
                  epochs=epochs,
                  validation_data=test_generator,
                  validation_steps=10)
```

# 2. 기준모델 + Epoch 50 + dropout

#9-시각화 결과

accuracy: 0.9000

loss: 0.3197

기준모델 + Optimizer Adam

# 3번째 시도

# 3. 기준모델 + Optimizer Adam

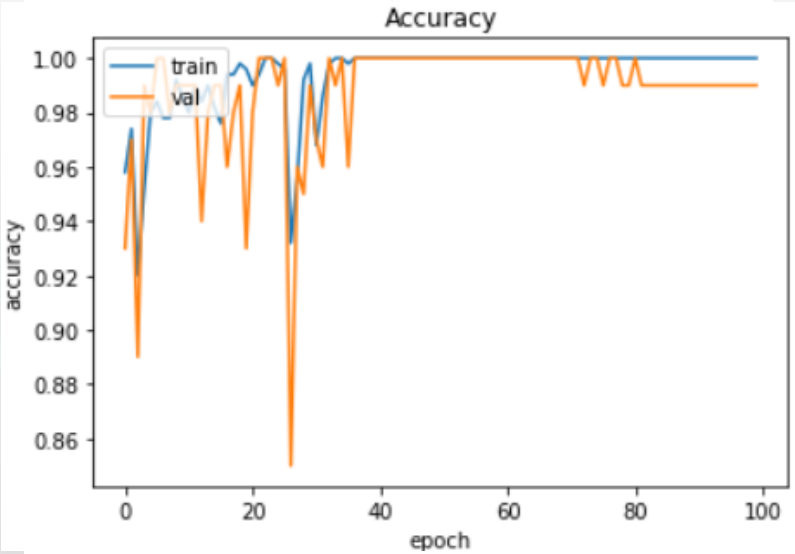
```
# 6. compile

model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

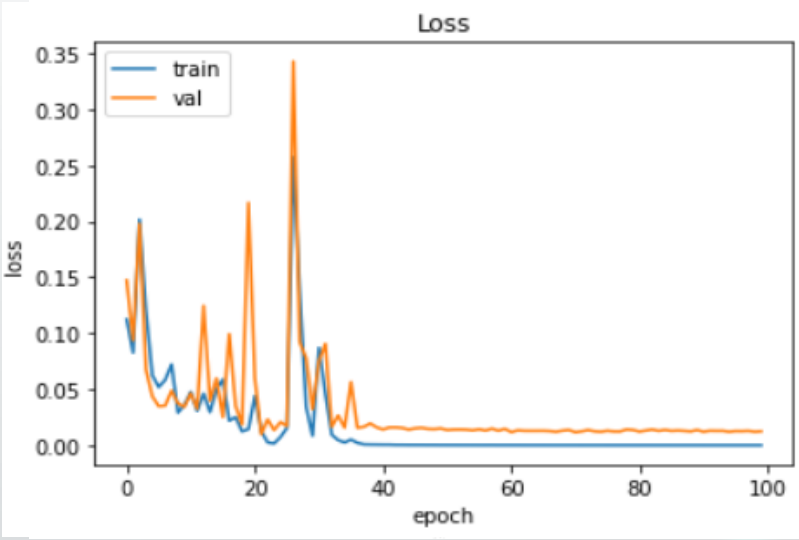
# 3. 기준모델 + Optimizer Adam

#9-시각화 결과

accuracy: 1.0000

loss: 0.0155



기준모델 + Optimizer Adam
+ Epoch 50

# 4번째 시도

# 4. 기준모델 + Optimizer Adam + Epoch 50

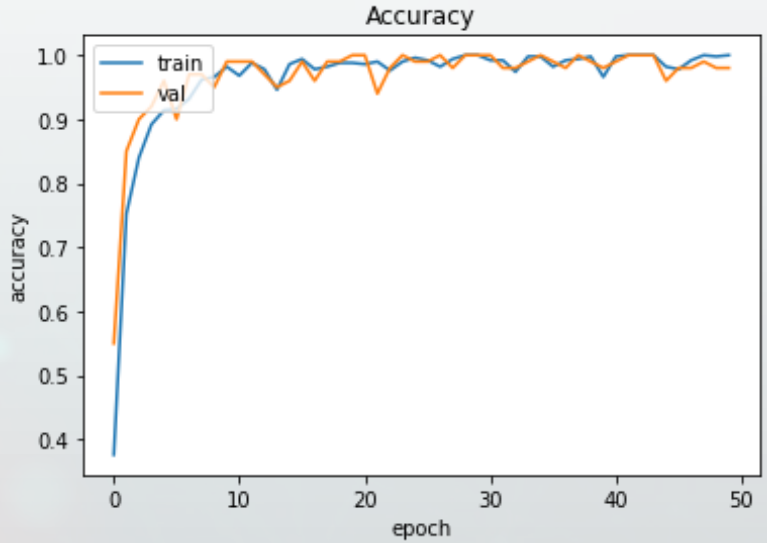
```
# 6. compile

model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])


#7.모델 학습시키기
epochs=50

history=model.fit(train_generator,steps_per_epoch=50,
                  epochs=epochs,
                  validation_data=test_generator,
                  validation_steps=10)
```

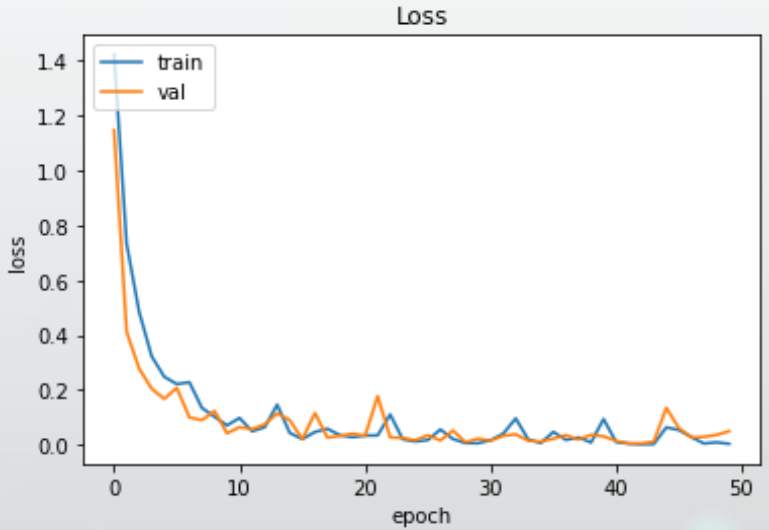
# 4. 기준모델 + Optimizer Adam + Epoch 50

#9-시각화 결과

accuracy: 1.0000

loss: 0.0030



# 이중분류 후 다중분류 모델

# 02

# 잘못된 여권사진 분류 후 위반한 기준에 따라 분류하기

**데이터 수집**

구글링을 통해
규정에 부합하는 여권사진 600장과
규정을 위반한 여권사진 600장 준비
Test 파일은 정상사진 100개와 비정상사진 100개,
Train 파일은 정상사진 500개와 비정상사진 500개로 분류

**이중분류 모델 준비 및 인공지능 학습**

Google Colab에
이중분류 코드
작성

**다중분류 모델 준비 및 인공지능 학습**

Google Colab에
다중분류 코드 작성

**전체 모델 테스트**

**결과 도출**

# 기준 모델

# 이중분류+다중분류 기준 모델

1. 이중분류 모델 학습 및 평가

```
# 1. 구글 드라이브 연동

from google.colab import drive

drive.mount('/content/drive')


Mounted at /content/drive


# 2. zip 파일 압축 풀기

import zipfile
with zipfile.ZipFile('/content/drive/My Drive/data_set2.zip', 'r') as zip_ref:
  zip_ref.extractall('/content/drive/My Drive/data_set2')
```

# 이중분류+다중분류 기준 모델

```
# 3. Data set 불러오기

from tensorflow import keras

# 훈련셋 + data exaggeration
train_datagen = keras.preprocessing.image.ImageDataGenerator(rescale = 1/255,
                                                              horizontal_flip = True)
train_generator = train_datagen.flow_from_directory('/content/drive/My Drive/data_set2/train', target_size = (413, 531),
batch_size = 10, class_mode = 'binary', shuffle = True)

# 평가셋
Test_datagen = keras.preprocessing.image.ImageDataGenerator(rescale = 1/255)
Test_generator = test_datagen.flow_from_directory('/content/drive/My Drive/data_set2/test', target_size = (413, 531),
batch_size = 10, class_mode = 'binary')

print(train_generator.image_shape)
print(train_generator.class_indices)

#3-Data set 결과
Found 1000 images belonging to 2 classes. Found 200 images belonging to 2 classes. (413, 531, 3) {'abnormal': 0, 'normal': 1}
```

# 이중분류+다중분류 기준 모델

```
# 4. data 살펴보기

print(test_generator[0][0].shape)


#4-Data 살펴보기 결과

(10, 413, 531, 3)
```

# 이중분류+다중분류 기준 모델

```
# 5. 모델 구성
from tensorflow import keras
from keras.layers import Dense, Conv2D, MaxPooling2D, Flatten, Dropout

model=keras.Sequential()
model.add(Conv2D(filters = 32, kernel_size = (3,3), activation = 'relu',input_shape = test_generator.image_shape))
model.add(Conv2D(32,(3,3), activation = 'relu'))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Dropout(0.2))

model.add(Conv2D(64, (3,3),activation = 'relu'))
model.add(MaxPooling2D(pool_size = (2,2)))

model.add(Conv2D(128, (3,3), activation = 'relu'))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Flatten())

model.add(Dense(64, activation = 'relu'))
model.add(Dense(32, activation = 'relu'))
model.add(Dropout(0.2))

model.add(Dense(1, activation = 'sigmoid'))

model.summary()
```

# 이중분류+다중분류 기준 모델

#5-모델구성 결과

```
Model: "sequential"
_________________________________________________________________
Layer (type) Output Shape Param #
=================================================================
conv2d (Conv2D) (None, 411, 529, 32) 896
_________________________________________________________________
conv2d_1 (Conv2D) (None, 409, 527, 32) 9248
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 204, 263, 32) 0
_________________________________________________________________
dropout (Dropout) (None, 204, 263, 32) 0
_________________________________________________________________
conv2d_2 (Conv2D) (None, 202, 261, 64) 18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 101, 130, 64) 0
_________________________________________________________________
conv2d_3 (Conv2D) (None, 99, 128, 128) 73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 49, 64, 128) 0
_________________________________________________________________
flatten (Flatten) (None, 401408) 0
_________________________________________________________________
dense (Dense) (None, 64) 25690176
_________________________________________________________________
dense_1 (Dense) (None, 32) 2080
_________________________________________________________________
dropout_1 (Dropout) (None, 32) 0
_________________________________________________________________
dense_2 (Dense) (None, 1) 33
=================================================================
Total params: 25,794,785 Trainable params: 25,794,785 Non-trainable
params: 0
_________________________________________________________________
```

# 이중분류+다중분류 기준 모델

```
# 6. compile

model.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accuracy'])


# 7. train

from keras.callbacks import EarlyStopping

early_stop = EarlyStopping(patience = 15)

epochs = 100

history = model.fit(train_generator,
                    steps_per_epoch = 100,
                    epochs = epochs,
                    validation_data = test_generator,
                    validation_steps = 20,
                    callbacks = [early_stop])
```

# 이중분류+다중분류 기준 모델

```
# 8. Evaluate

result = model.evaluate(test_generator, steps = 1)
print(result)
```

#8-Evaluate 결과

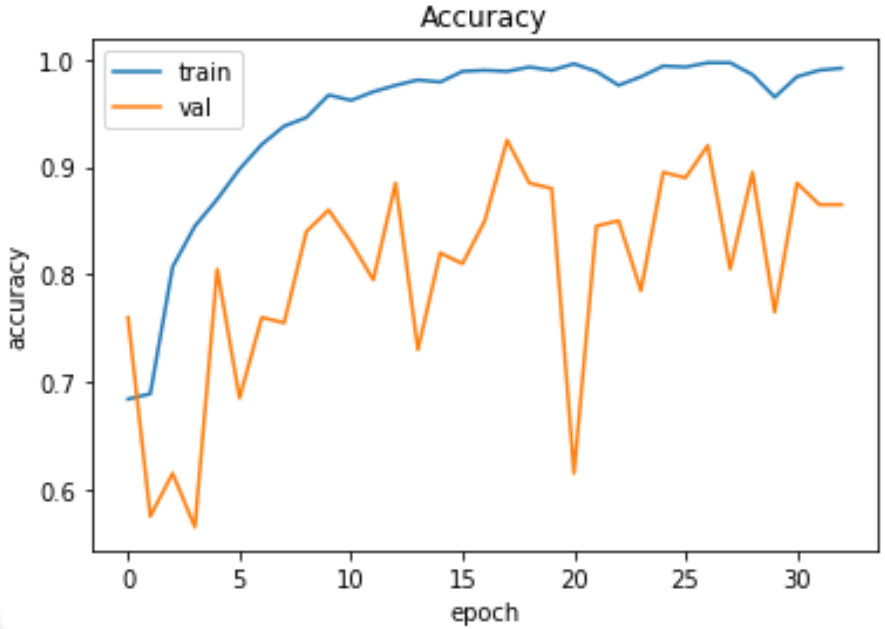
```
loss: 0.0710
accuracy: 1.0000
```

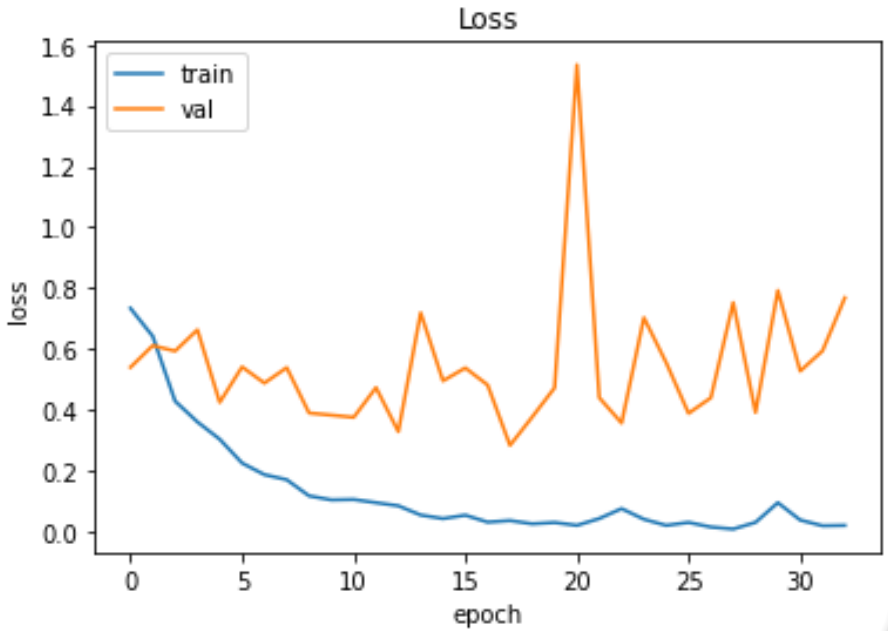
# 이중분류+다중분류 기준 모델

```
# 9. 시각화

import matplotlib.pyplot as plt

plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])

plt.title('Accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc = 'upper left')
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])

plt.title('Loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc = 'upper left')
plt.show()
```

# 이중분류+다중분류 기준 모델

#9-시각화 결과



# 이중분류+다중분류 기준 모델

```
# 10. 모델 저장

model_path1 = 'passport_wrongornot.h5'
model.save(model_path1)
```

# 이중분류+다중분류 기준 모델

2. 다중분류 모델 학습 및 평가

```
# 11. zip 파일 압축 풀기
import zipfile
with zipfile.ZipFile('/content/drive/My Drive/data_set.zip', 'r') as zip_ref:
zip_ref.extractall('/content/drive/My Drive/data_set')

# 12. 데이터셋 불러오기

from tensorflow import keras
# 훈련셋 + data exaggeration
train_datagen2 = keras.preprocessing.image.ImageDataGenerator(rescale = 1/255, horizontal_flip = True)
train_generator2 = train_datagen2.flow_from_directory('/content/drive/My Drive/data_set/train', target_size = (413, 531), b
atch_size = 20, class_mode = 'categorical', shuffle = True)

# 평가셋
Test_datagen2 = keras.preprocessing.image.ImageDataGenerator(rescale = 1/255)
Test_generator2 = test_datagen2.flow_from_directory('/content/drive/My Drive/data_set/test', target_size = (413, 531), bat
ch_size = 20, class_mode = 'categorical')

print(train_generator2.image_shape)
print(train_generator2.class_indices)
```

# 이중분류+다중분류 기준 모델

# 12-데이터셋 불러오기 결과

Found 500 images belonging to 5 classes. Found 100 images belonging to 5 classes. (413, 531, 3) {'background': 0, 'light': 1, 'location': 2, 'quality': 3, 'size': 4}

# 13. data 살펴보기

```
print(test_generator2[0][0].shape)
```

# 13-data 살펴보기 결과

(10, 413, 531, 3)

# 이중분류+다중분류 기준 모델

```
# 14. 모델 구성
model2 = keras.Sequential()
model2.add(Conv2D(filters = 32, kernel_size = (3,3), activation = 'relu',
            input_shape = test_generator.image_shape, padding = 'same'))
model2.add(Conv2D(32,(3,3),activation = 'relu', padding = 'same'))
model2.add(MaxPooling2D(pool_size=(2,2)))
model2.add(Dropout(0.2))

model2.add(Conv2D(64, (3,3), activation = 'relu', padding = 'same'))
model2.add(MaxPooling2D(pool_size=(2,2)))

model2.add(Conv2D(128, (3,3),activation = 'relu', padding = 'same'))
model2.add(MaxPooling2D(pool_size=(2,2)))
model2.add(Flatten())

model2.add(Dense(64, activation = 'relu'))
model2.add(Dense(32, activation = 'relu'))
model2.add(Dropout(0.5))

model2.add(Dense(5, activation = 'softmax'))

model2.summary()
```

# 이중분류+다중분류 기준 모델

#14-모델구성 결과

```
Model: "sequential"
_________________________________________________________________
Layer (type) Output Shape Param #
=================================================================
conv2d (Conv2D) (None, 413, 531, 32) 896
_________________________________________________________________
conv2d_1 (Conv2D) (None, 413, 531, 32) 9248
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 206, 265, 32) 0
_________________________________________________________________
dropout (Dropout) (None, 206, 265, 32) 0
_________________________________________________________________
conv2d_2 (Conv2D) (None, 206, 265, 64) 18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 103, 132, 64) 0
_________________________________________________________________
conv2d_3 (Conv2D) (None, 103, 132, 128) 73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 51, 66, 128) 0
_________________________________________________________________
flatten (Flatten) (None, 430848) 0
_________________________________________________________________
dense (Dense) (None, 64) 27574336
_________________________________________________________________
dense_1 (Dense) (None, 32) 2080
_________________________________________________________________
dropout_1 (Dropout) (None, 32) 0
_________________________________________________________________
dense_2 (Dense) (None, 5) 165
=================================================================
Total params: 27,679,077 Trainable params: 27,679,077 Non-
trainable params: 0
_________________________________________________________________
```

# 이중분류+다중분류 기준 모델

```
# 15. compile

model2.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])

# 16. train
from keras.callbacks import EarlyStopping

early_stop = EarlyStopping(patience = 15)

epochs = 100
history2 = model2.fit(train_generator2,
                      steps_per_epoch = 50,
                      epochs = epochs,
                      validation_data = test_generator2,
                      validation_steps = 10,
                      callbacks = [early_stop])
```

# 이중분류+다중분류 기준 모델

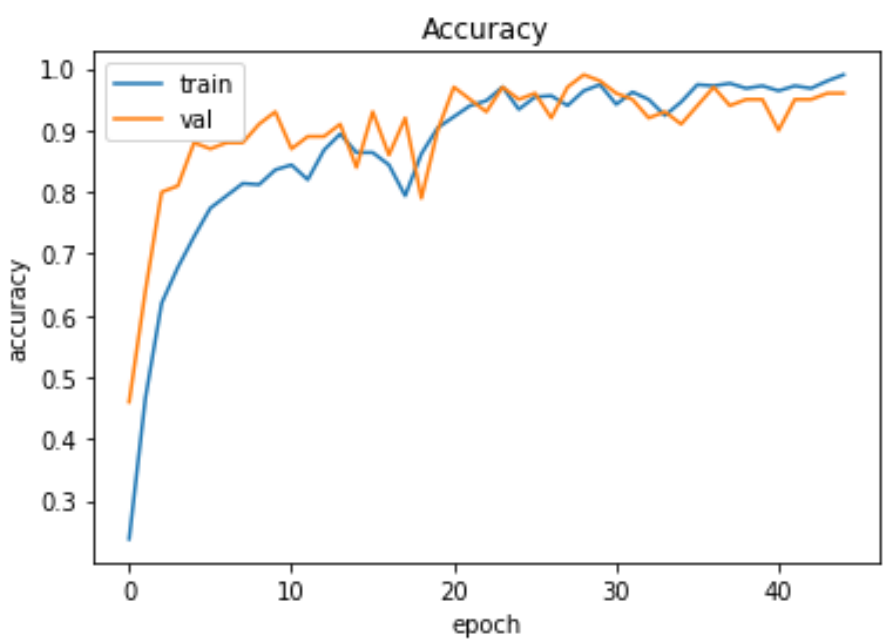
```
# 17. evaluate

result2 = model2.evaluate(test_generator2, steps = 1)
print(result2)


#17-Evaluate 결과

loss: 0.1671
accuracy: 0.9000
```

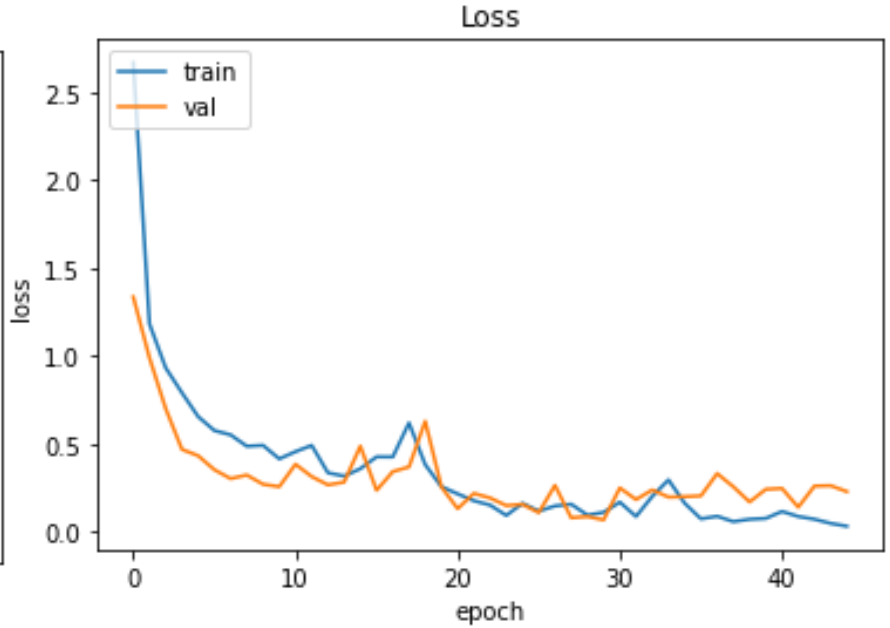
# 이중분류+다중분류 기준 모델

```
# 18. 시각화

import matplotlib.pyplot as plt

plt.plot(history2.history['accuracy'])
plt.plot(history2.history['val_accuracy'])

plt.title('Accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc = 'upper left')
plt.show()

plt.plot(history2.history['loss'])
plt.plot(history2.history['val_loss'])

plt.title('Loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc = 'upper left')
plt.show()
```

# 이중분류+다중분류 기준 모델

# 18-시각화 결과



# 이중분류+다중분류 기준 모델

```
# 19. 모델 저장

model_path2 = 'passport_wrongtype.h5'
model2.save(model_path2)
```

# 이중분류+다중분류 기준 모델

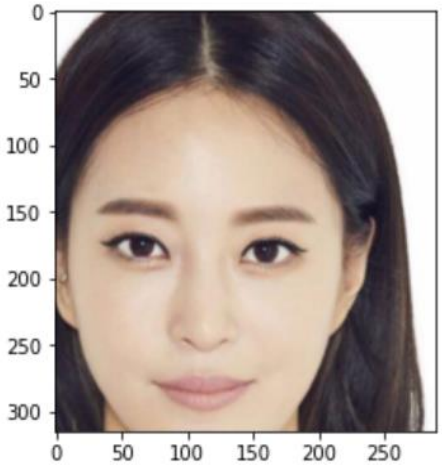
3. 전체 모델 test

```
# 20. 모델 불러오기

from keras.models import load_model

model_1 = load_model(model_path1)
model_2 = load_model(model_path2)
print(model_1.summary())
print(model_2.summary())


# 21. 새로운 이미지 테스트

from google.colab import files

upload = files.upload()
```

# 이중분류+다중분류 기준 모델

```
# 22. 이미지 확인

import cv2

test_image = cv2.imread('KakaoTalk_20201210_124224293.png')
test_image = cv2.cvtColor(test_image, cv2.COLOR_BGR2RGB)
plt.imshow(test_image)
```

```
<matplotlib.image.AxesImage at 0x7fc7fa931ba8>
```




```
[ ]  print(test_image.shape)

     (315, 290, 3)
```

# 이중분류+다중분류 기준 모델

```
# 23. resize -> (413, 531, 3)

image_resized = cv2.resize(test_image, dsize=(531,413),
                           interpolation=cv2.INTER_AREA)

print(image_resized.shape)

# 23- resize 결과
(413, 531, 3)
```

# 이중분류+다중분류 기준 모델

```
# 24. test

import numpy as np
result = model_1.predict(image_resized.reshape(1, 413, 531, 3))

if result == 0:
  result2 = model_2.predict(image_resized.reshape(1, 413, 531, 3))
  score = result2[0]
  print("abnormal:")
  print(print(np.where(result2 == np.amax(result2))[1]))
else:
  print("normal: ")
  print(print(np.where(result == np.amax(result))[1]))

# 24- test 결과
abnormal:
[1]
None
```

# 모델의 정확도를 높이기 위한 노력

지금까지 작성한 여러 코드를 바꾸어 적용

- loss, accuracy값이 지나치게 초반에 빠르게 좋아진 모델을 학습시켰더니 실제 정확도는 매우 떨어졌다.
- loss, accuracy값이 계속해서 변화하지 않는 모델은 코드가 잘못되었음을 확인하고 코드를 수정하였다.
- data exaggeration을 추가하였더니 기존의 loss, accuracy값보다 훨씬 높거나 낮게 나타나는 등 이상한 결과를 보이기도 하였다.
- 모델을 초기화하지 않고 계속 코드를 수정하여 돌렸더니 모델이 초반 batch부터 loss, accuracy값이 너무 낮게 또는 너무 높게 나왔다.

# 04

# 결론

# 개선할 점과 아쉬운 점

1. 다중분류에서 시도해볼 수 있을 방법들

   1) 이미지 부풀리기

   2) 마지막 부분에서 예측할 때 몇 %로 예측했는지 나타내도록 코드 추가

# 개선할 점과 아쉬운 점

2. 이중분류 후 다중분류 모델에서 시도해볼 수 있을 방법들

   1) 테스트에서 성공할 수 있도록 loss 값 줄이고 accuracy 값 늘리기

   2) 갑자기 loss가 높아지거나 accuracy가 낮아지는 부분 해결

   ⇒ Epoch 올리기

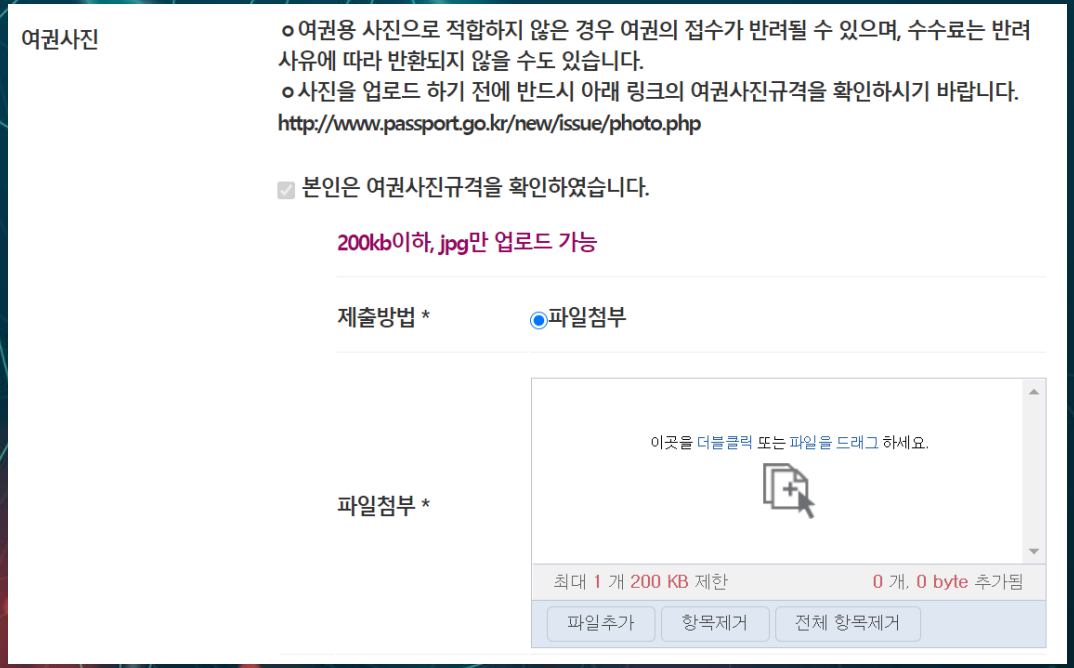
   ⇒ 모델구성: cnn 모델 쌓는 부분 수정

   ⇒ 모델학습: Epoch 올리기, earlystoppoing과 metrics 변화

   ⇒ 적절한 test 이미지 모색

# 현실에서 프로젝트의 이용가능성

## 민원24를 통한 온라인 여권신청 시




현재는 업로드한 여권사진을
사람이 직접 확인하여 승인

이 프로젝트를 통해
구성한 모델을 사용하면,
인공지능을 활용하여
더욱 빠르고 편리하게
여권사진을 판별할 수 있다.

# 프로젝트를 마치며

인공지능이 가깝게 느껴졌고, 직접 해보면서 새로운 주제를 찾아 도전해볼 자신감도 생긴 것 같아 좋았다.

**김서현**

머신러닝과 딥러닝 기법에 대해 구체적으로 배우고 이를 적용하여 모델을 직접 구성할 수 있어 유익한 시간이었다.

**주민**

짧게 배운 인공지능이였지만 실질적으로 모델을 만들고 그 모델을 바탕으로 산업체에서 사용할만한 결과물을 만들었다는게 뿌듯하며 팀원들이랑 같이해서 좋았다.

**오원석**

처음으로 모델을 구성하고 인공지능을 학습시켜보았기에 많은 새로운 지식을 습득할 수 있었다.

**김유경**

생소했던 인공지능에 대해 배울 수 있었고 새로운 팀프로젝트를 경험할 수 있어 좋았다.

**박희원**

# 역할 배분

**주민**: 다중분류 데이터 수집 및 가공(location), 다중분류 모델 구성 및 효율화 작업, 이중분류와 다중분류 모델 작성 및 효율화 작업, 주제 선정 등에 대한 프로젝트 전체 회의 참석하여 아이디어 제공

**박희원**: 다중분류 데이터 수집 및 가공(background), 다중분류 모델 구성 및 정확도 높이기 작업, 다중분류 PPT 수정, 주제 선정 등에 대한 프로젝트 전체 회의 참석하여 아이디어 제공

**김서현**: 다중분류 데이터 수집 및 가공(size), 다중분류 모델 구성 및 정확도 높이기 작업, 이중분류 데이터 수집 및 가공(normal), 주제 선정 등에 대한 프로젝트 전체 회의 참석하여 아이디어 제공

**김유경**: 다중분류 데이터 수집 및 가공(light), 다중분류 모델 구성, 전체 PPT 제작 및 수정, 주제 선정 등에 대한 프로젝트 회의 참석하여 아이디어 제공

**오원석**: 다중분류 데이터 수집 및 가공(quality), 다중분류 모델 구성, 발표대본 집필 및 발표, 데이터 수집 등에 대한 프로젝트 회의 참석하여 아이디어 제공

# Q & A

# Thank You