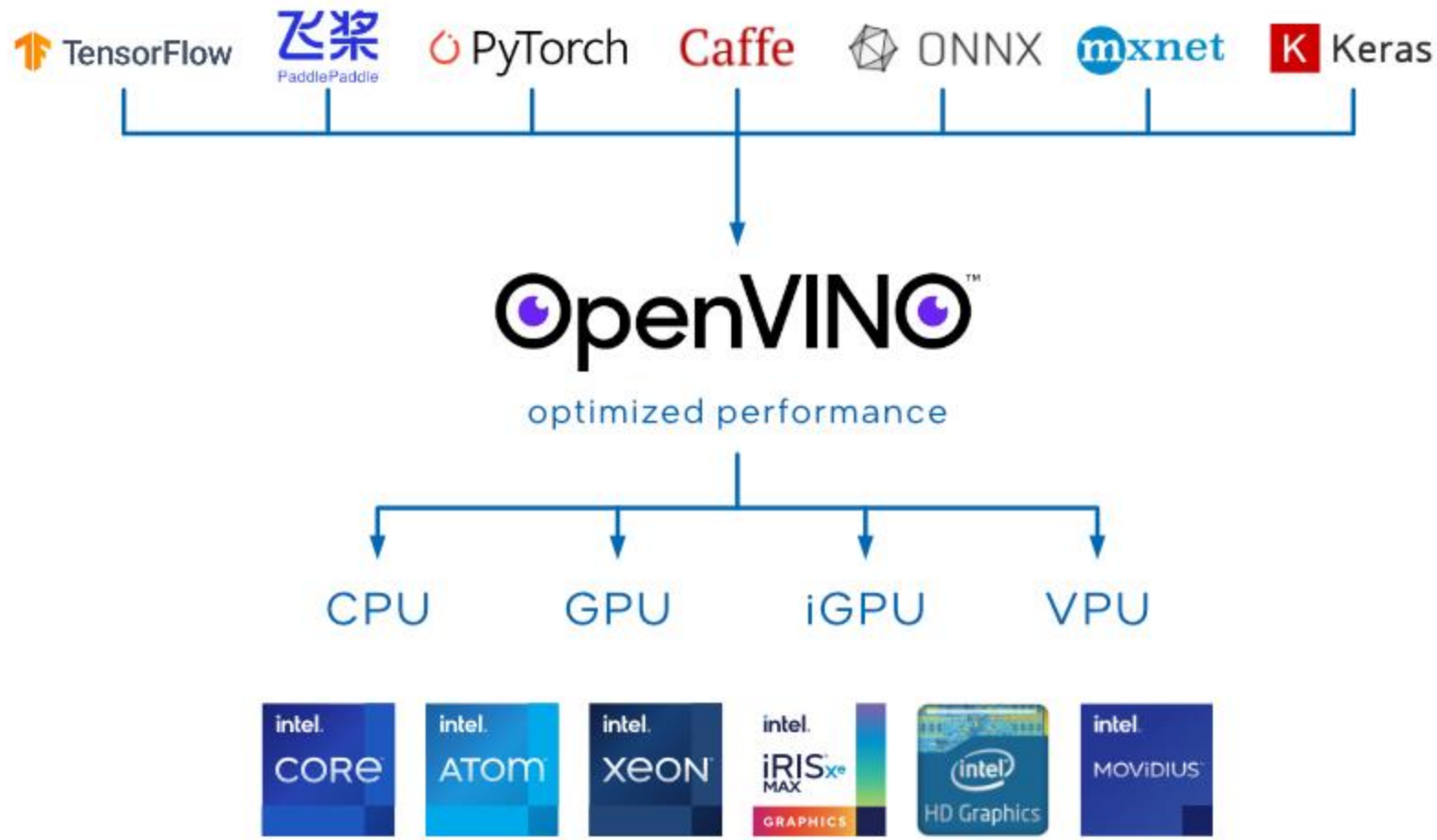
AI PC Image Recognition Training & Inferencing

Intel, HP, G-Market, CURIOSITY PROJECT, NUMP

AI Hackaton


TensorFlow
飞桨
PaddlePaddle
PyTorch
Caffe
ONNX
mxnet
Keras
OpenVINO™
optimized performance
CPU
GPU
iGPU
VPU
intel. CORE
intel. ATOM
intel. XEON
intel. iRIS Xe MAX GRAPHICS
intel. HD Graphics
intel. MOVIDIUS

MODIN · SciPy · dmlc XGBoost · PyTorch · ONNX RUNTIME · OpenVINO™ Write Once Deploy Anywhere

pandas · NumPy · scikit-learn · TensorFlow · Direct ML

Apache Spark · Numba · SigOpt Automate Model Tuning AutoML · Intel® Neural Compressor

Data Analytics at Scale†

Machine & Deep Learning Frameworks, Optimization and Deployment Tools†

oneAPI

Intel® oneAPI Deep Neural Network Library (oneDNN)

Intel® oneAPI Collective Communications Library (oneCCL)

Intel® oneAPI Math Kernel Library (oneMKL)

Intel® oneAPI Data Analytics Library (oneDAL)

Open, cross-architecture programming model for CPUs, GPUs, and other accelerators

Cloud & Enterprise

Client & Workstation

Edge

intel AI TOOLS

Accelerate End-to-End Data Science and AI

Intel® Developer Cloud

Intel® Developer Catalog

Try Latest Intel Tools and Hardware, and access optimized AI Models

cnvrg.io

Full stack ML Operating System

Hugging Face

Intel optimizations and fine-tuning recipes, optimized inference models, and model serving

Note: components at each layer of the stack are optimized for targeted components at other layers based on expected AI usage models, and not every component is utilized by the solutions in the rightmost column

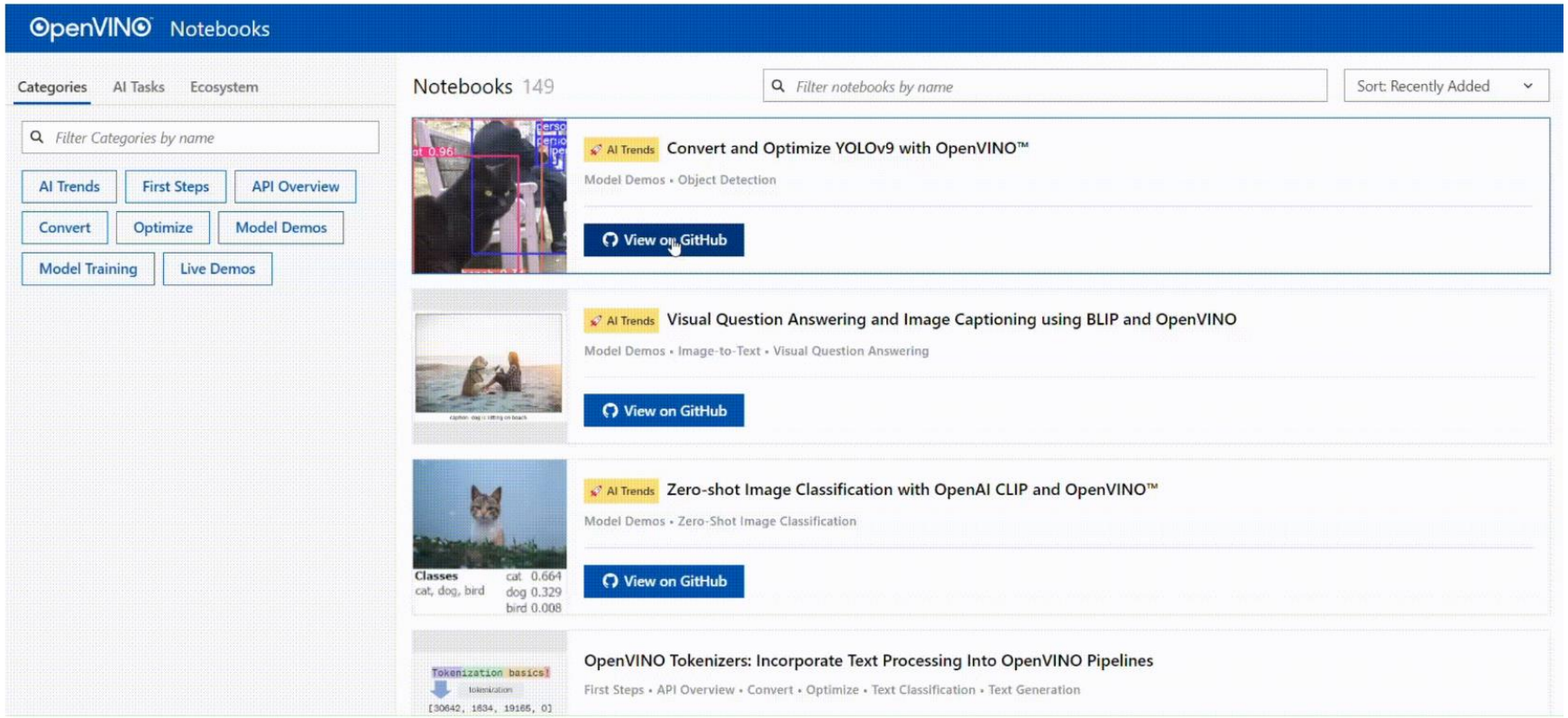
† This list includes popular open source frameworks that are optimized for Intel hardware

GitHub - openvinotoolkit/openvino_notebooks: 📚 Jupyter notebook tutorials for OpenVINO™

📚 OpenVINO™ Notebooks

A collection of ready-to-run Jupyter notebooks for learning and experimenting with the OpenVINO™ Toolkit. The notebooks provide an introduction to OpenVINO basics and teach developers how to leverage our API for optimized deep learning inference.

🚀 Checkout interactive GitHub pages application for navigation between OpenVINO™ Notebooks content: OpenVINO™ Notebooks at GitHub Pages



OpenVINO Notebooks

Categories | AI Tasks (1) | Ecosystem

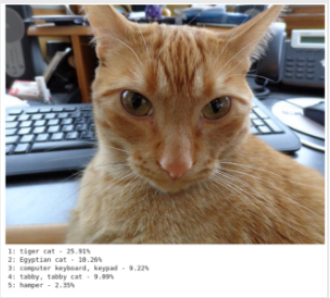
Filter AI Tasks by name

Multimodal

- Text-to-Image
- Image-to-Text
- Text-to-Video
- Video-to-Text
- Text-to-Audio
- Audio-to-Text
- Visual Question Answering
- Image Captioning
- Feature Extraction
- Text-to-Image Retrieval
- Image-to-Text Retrieval

Computer Vision

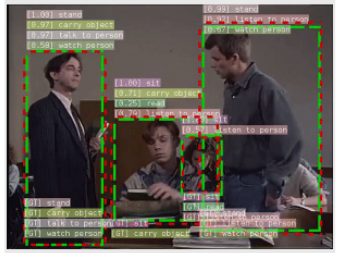
- **Image Classification**
- Image Segmentation
- Image Inpainting
- Image-to-Image
- Object Detection
- Salient Object Detection
- Depth Estimation
- Super Resolution



Convert a PyTorch Model to OpenVINO™ IR

Convert • Image Classification

View on GitHub | Open in Colab



Video Recognition using SlowFast and OpenVINO™

Model Demos • Object Detection • Image Classification

View on GitHub

Convert a Tensorflow Lite Model to OpenVINO™

Convert • Image Classification

View on GitHub | Open in Colab

OpenVINO Notebooks

Optimize Preprocessing

API Overview • Optimize • Image Classification

View on GitHub | Open in Colab

The PASCAL Visual Object Classes Challenge 2012 (VOC2012) (ox.ac.uk)

Visual Object Classes Challenge 2012 (VOC2012)

PASCAL2
Pattern Analysis, Statistical Modelling and Computational Learning

[click on an image to see the annotation]

For news and updates, see the PASCAL Visual Object Classes Homepage

Mark Everingham

It is with great sadness that we report that Mark Everingham died in 2012. Mark was the key member of the VOC project, and it would have been impossible without his selfless contributions. The VOC workshop at ECCV 2012 was dedicated to Mark's memory. A tribute web page has been set up, and an appreciation of Mark's life and work published.

Development Kit

The development kit consists of the training/validation data, MATLAB code for reading the annotation data, support files, and example implementations for each competition.

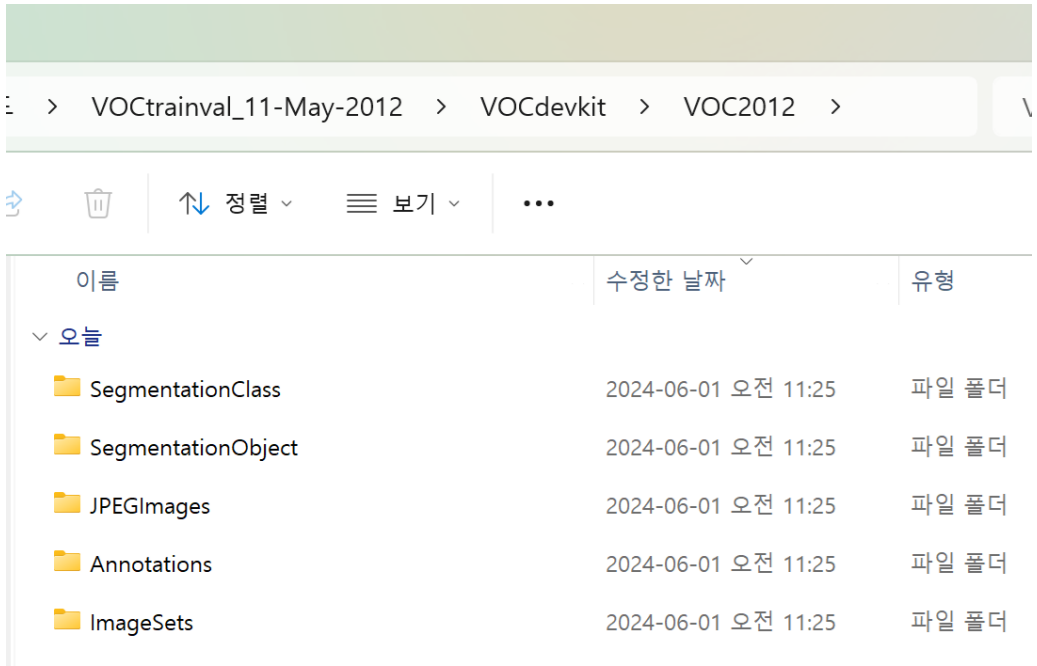
The development kit is now available:

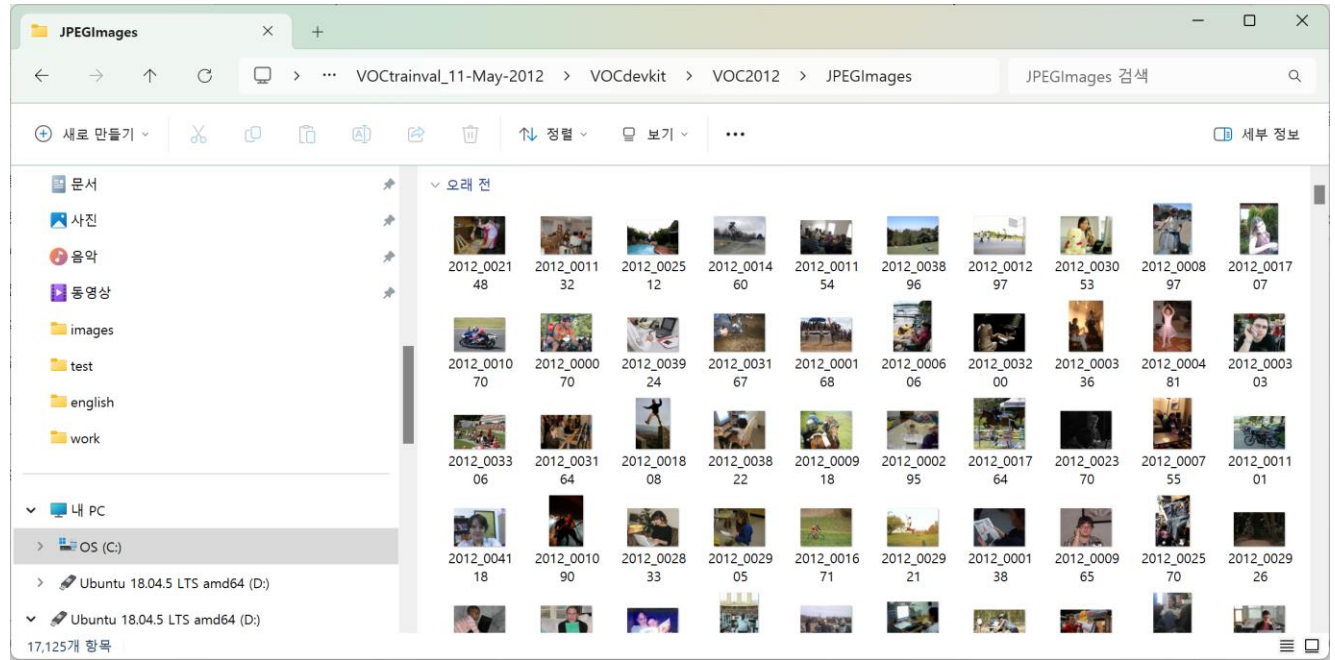
- Download the training/validation data (2GB tar file)
- Download the development kit code and documentation (500KB tar file)
- Download the PDF documentation (500KB PDF)
- Browse the HTML documentation
- View the guidelines used for annotating the database (VOC2011)
- View the action guidelines used for annotating the action task images

Test Data

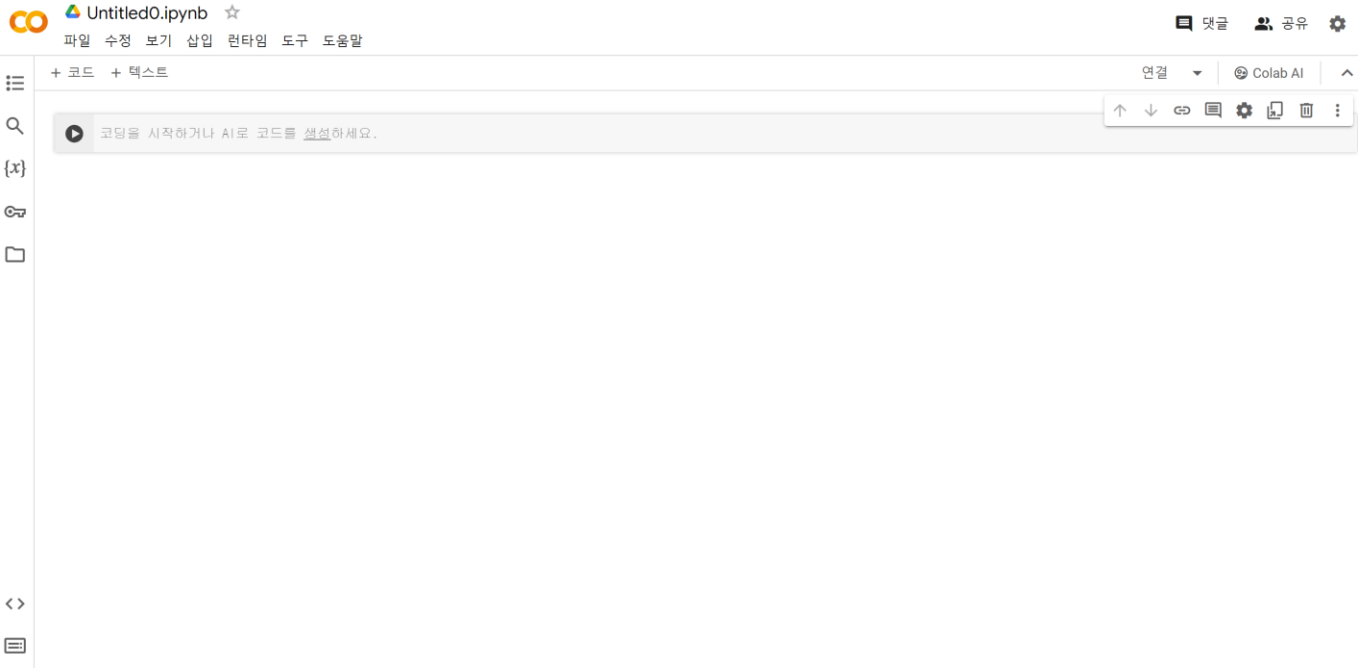
The test data will be made available according to the challenge timetable. Note that the only annotation in the data is for the action task and layout taster. As in 2008-2011, there are no current plans to release full annotation - evaluation of results will be provided by the organizers.

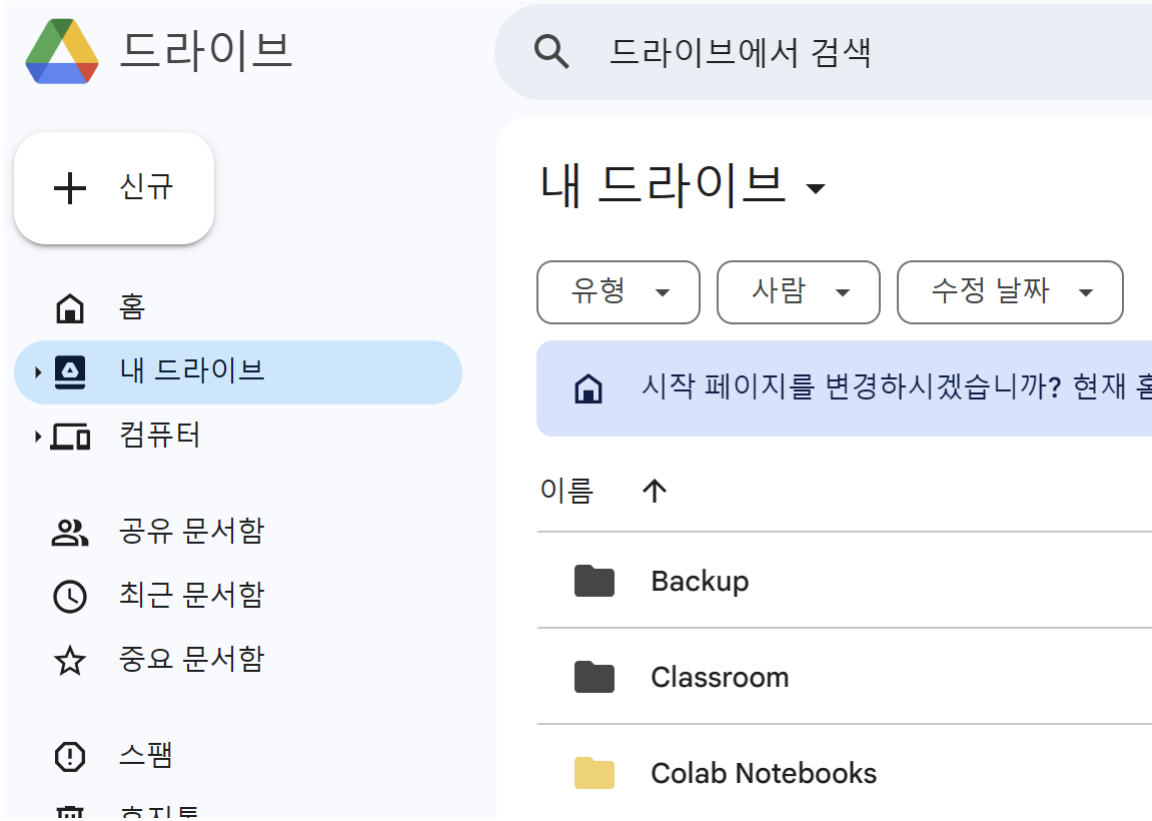
The test data can be downloaded from the evaluation server. You can also use the evaluation server to evaluate your method on the test data.





https://colab.research.google.com

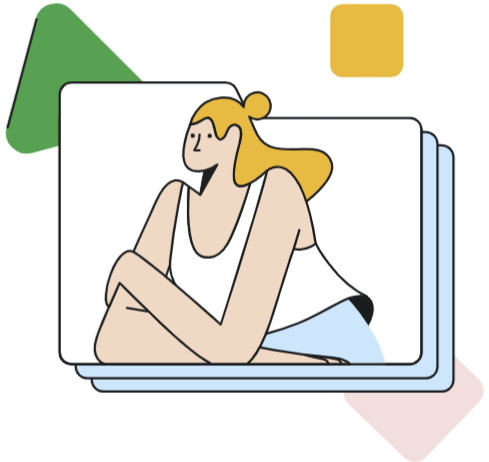




내 드라이브 > Colab Notebooks > data

유형 사람 수정 날짜

단축키 업데이트 2024년 8월 1일 목요일부터 첫 글자를 탐색할 수 있도록 Drive 단축키가 업데이트됩니다. 자세히 알아보기 단축키 보기 지금 업데이트



파일을 여기 끌어다 놓거나

'새로 만들기' 버튼을 사용하세요.

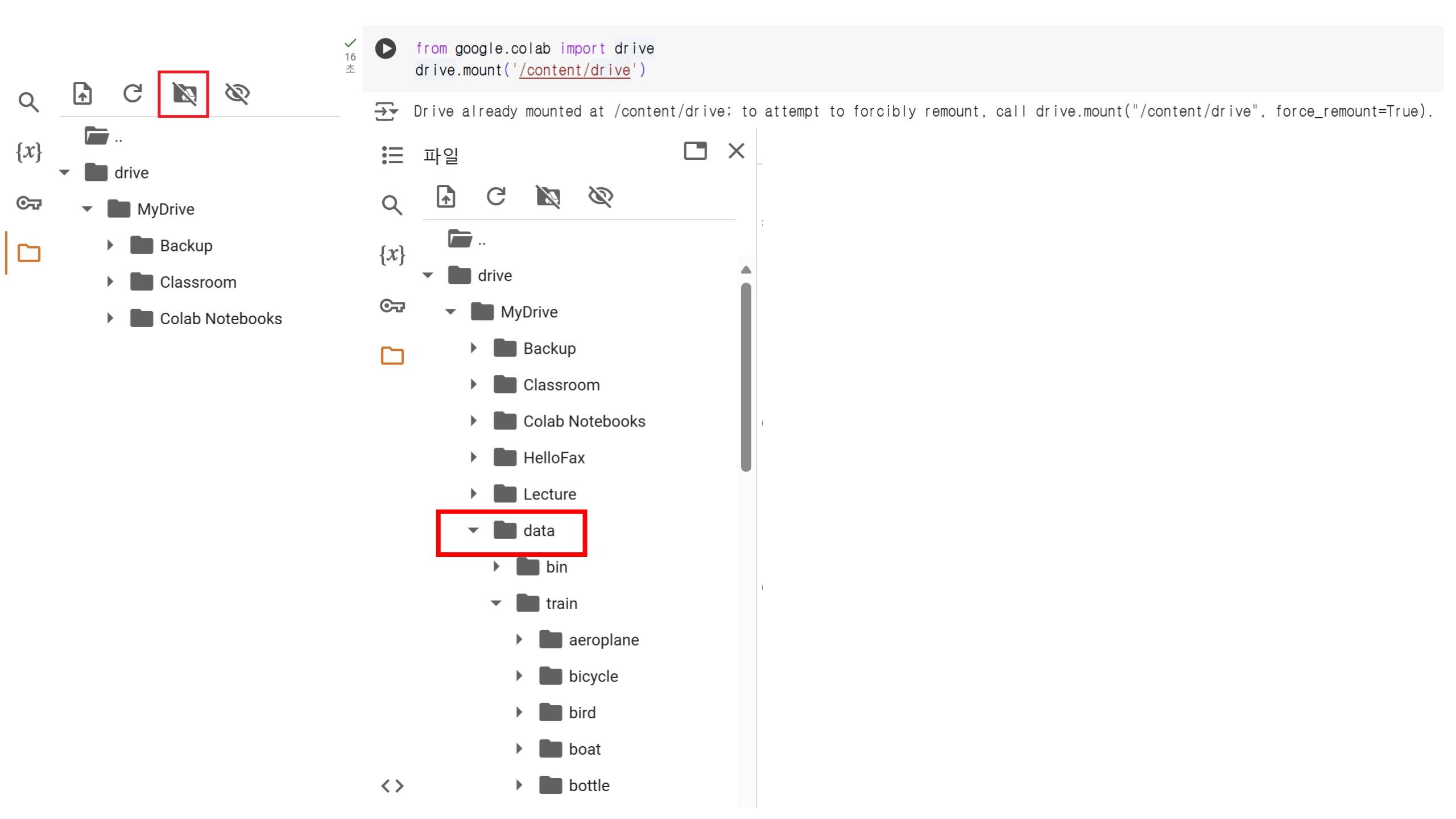
2개 항목 업로드 중

업로드 시작 중... 취소

train 0/3059개

val 0/808개

이벤트에 대한 알림을 받습니다. 사용


16
초
from google.colab import drive
drive.mount('/content/drive')
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
..
drive
MyDrive
Backup
Classroom
Colab Notebooks
파일
..
drive
MyDrive
Backup
Classroom
Colab Notebooks
HelloFax
Lecture
data
bin
train
aeroplane
bicycle
bird
boat
bottle

0초

```
[2]  # 우리가 분류할 8개의 클래스
     #CLASSES = [ 'car', 'cat', 'bottle', 'bus', 'bicycle', 'bird', 'boat', 'aeroplane' ]
     classes = []
```

학습에 필요한 상수 정의

0초

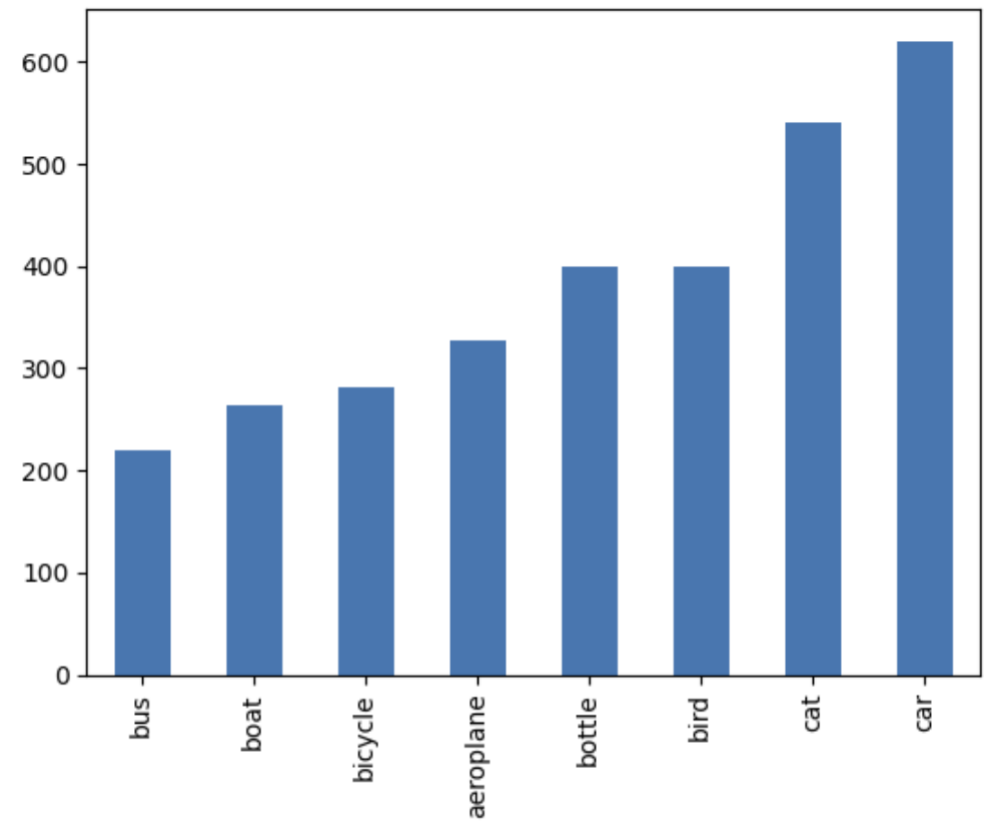
```
from pathlib import Path

# 학습에 필요한 상수들을 정의 합니다.
IMG_SHAPE = (224, 224, 3)
LEARNING_RATE = 2e-5
BATCH_SIZE = 16
EPOCHS = 10

# 각 이미지의 기본 주소
BASE_PATH = './drive/MyDrive/data/'
images_dir = Path(BASE_PATH).expanduser()
print(images_dir)
```

```
drive/MyDrive/data
```

내 드라이브 파일 마운트하고, 폴더 정보 읽어오기

13초

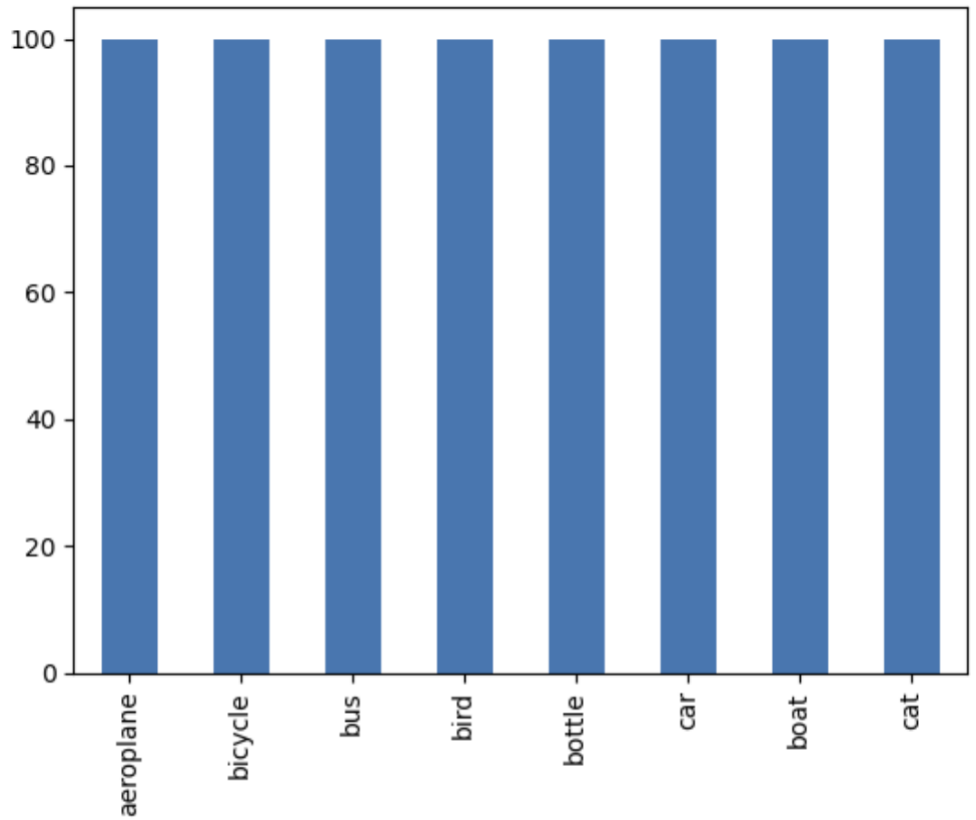
```
[3]  import os
     import matplotlib.pyplot as plt
     import pandas as pd

     # 모델 학습에 사용할 데이터 정보를 설정합니다.
     class_list = []
     num_list = []

     # IMAGE_BASE_PATH = './data/'
     train_path = BASE_PATH + 'train/'
     for folder in os.listdir(train_path):
         classes.append(folder)
         folder_size = len(os.listdir(train_path+folder))
         class_list.append(folder)
         num_list.append(folder_size)

     voc_s = pd.Series(num_list,index=class_list)
     voc_s.sort_values().plot(kind='bar')
     plt.show()

     print(voc_s.sort_values())
```




```
bus          219
boat         264
bicycle      281
aeroplane    328
bottle       399
bird         399
cat          540
car          621
dtype: int64
```

0초

```
# 모델 학습 검증에 사용할 데이터 내용 입니다.

class_list = []
num_list = []

#IMAGE_BASE_PATH = './data/'
valid_path = BASE_PATH + 'val/'
for folder in os.listdir(valid_path):
    folder_size = len(os.listdir(valid_path+folder))
#     print('{:<15} : {}'.format(folder,folder_size))
    class_list.append(folder)
    num_list.append(folder_size)

voc_s = pd.Series(num_list,index=class_list)
voc_s.sort_values().plot(kind='bar')
plt.show()

print(voc_s.sort_values())
```




```
aeroplane     100
bicycle       100
bus           100
bird          100
bottle        100
car           100
boat          100
cat           100
dtype: int64
```

Transfer Learning 을 통해 적은 데이터를 가지고 빠르게 학습시키기

여기서는 기존 weight(가중치) 값을 이용하여 데이터를 학습시킵니다.

✓ 5초

```
[8]  import keras
     import tensorflow as tf
     from tensorflow.keras.applications import MobileNet, MobileNetV2

     # base model 의 input shape, 그리고  trainable 을 false 로 합니다.

     base_model = MobileNetV2(input_shape=(224,224,3),
                              include_top=False,
                              weights='imagenet')
     base_model.trainable = False
```

```
Downloading data from https://storage.googleapis.com/tensorflow/kera
9406464/9406464 [==============================] - 0s 0us/step
```

Transfer Learning 에서 정확성을 높이기 위해 마지막 3개 block 에서 모델 트레이닝합니다.

0초

```
from tqdm import tqdm
# tqdm: progress bar를 표시해주는 패키지

# transfer learning에서 마지막 3개 block 을 사용하여 모델을 트레이닝 합니다.
set_trainable = False
for layer in tqdm(base_model.layers):
    if layer.name in ['block_14_expand','block_15_expand', 'block_16_expand']:
        set_trainable = True
    if set_trainable:
        layer.trainable = True
    else:
        layer.trainable = False
```

```
100%|██████████| 154/154 [00:00<00:00, 19171.97it/s]
```

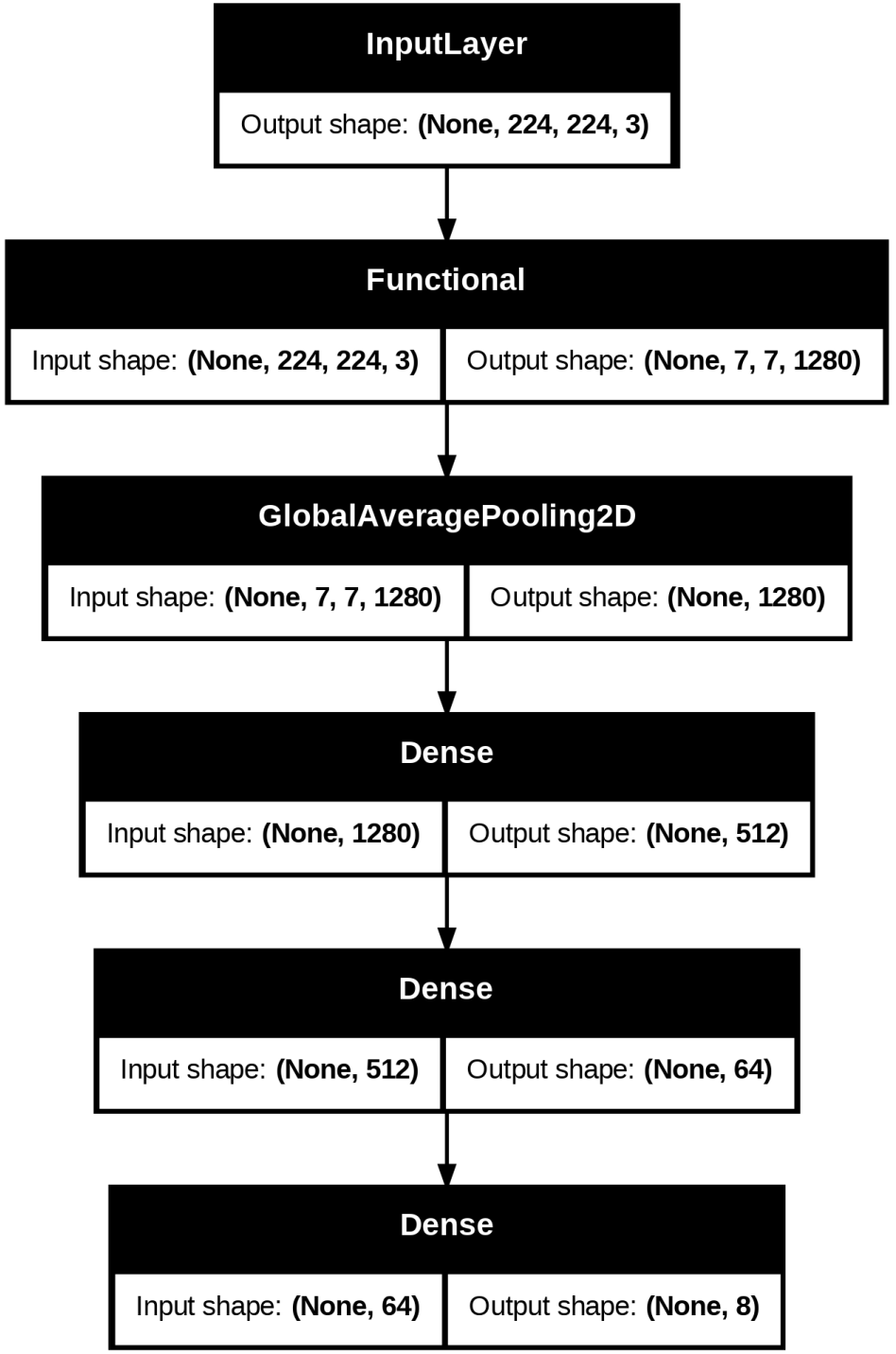
0초

```
# 레이어 구성을 살펴 봅니다.

layers = [(layer, layer.name, layer.trainable) for layer in base_model.layers]

pd.DataFrame(layers, columns=['Layer Type', 'Layer Name', 'Layer Trainable'])
```

	Layer Type	Layer Name	Layer Trainable
0	<keras.src.engine.input_layer.InputLayer objec...	input_1	False
1	<keras.src.layers.convolutional.conv2d.Conv2D ...	Conv1	False
2	<keras.src.layers.normalization.batch_normaliz...	bn_Conv1	False
3	<keras.src.layers.activation.relu.ReLU object ...	Conv1_relu	False
4	<keras.src.layers.convolutional.depthwise_conv...	expanded_conv_depthwise	False
...	...	...	...
149	<keras.src.layers.convolutional.conv2d.Conv2D ...	block_16_project	True
150	<keras.src.layers.normalization.batch_normaliz...	block_16_project_BN	True
151	<keras.src.layers.convolutional.conv2d.Conv2D ...	Conv_1	True
152	<keras.src.layers.normalization.batch_normaliz...	Conv_1_bn	True
153	<keras.src.layers.activation.relu.ReLU object ...	out_relu	True

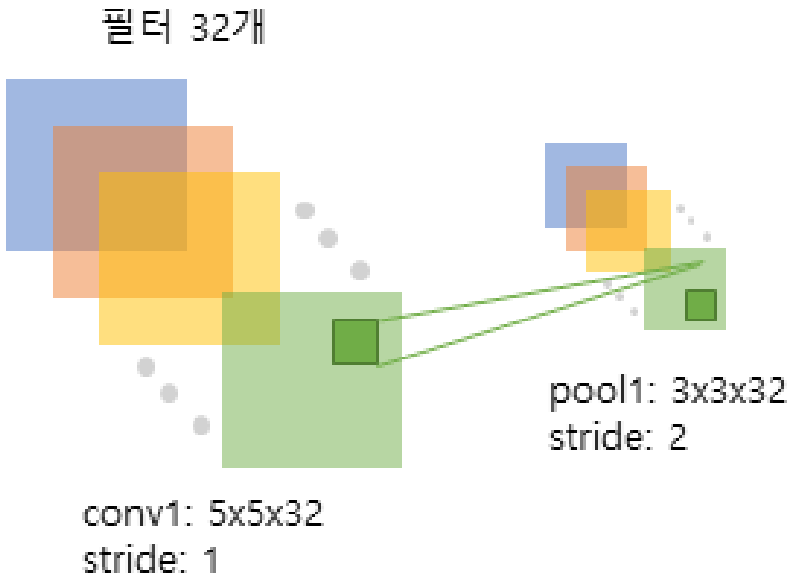
154 rows × 3 columns

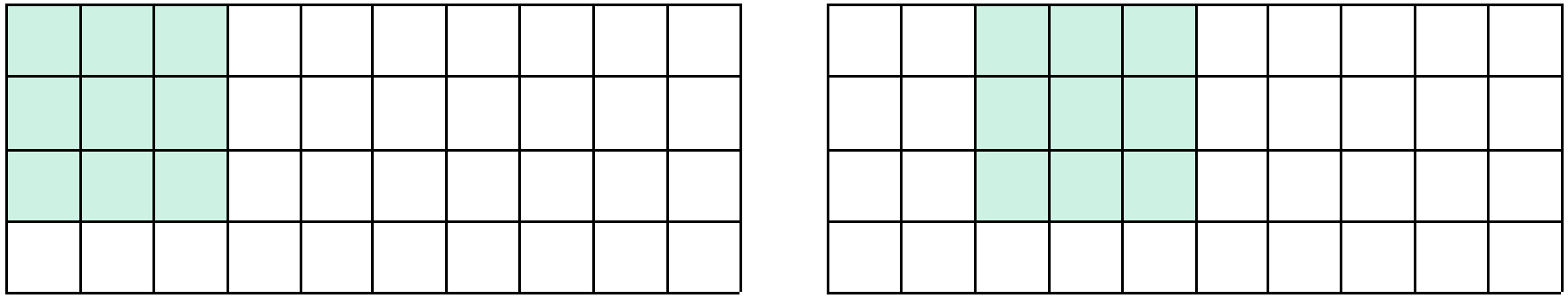
```python
from tensorflow.keras.models import Model, Sequential
from tensorflow.keras.layers import Dense, Conv2D,GlobalAveragePooling2D, Input
from tensorflow.keras.utils import plot_model

# Define input shape explicitly
input_tensor = Input(shape=(224, 224, 3))

# Connect base_model to input tensor
x = base_model(input_tensor)

# Build the rest of the model
x = GlobalAveragePooling2D()(x)
x = Dense(512,activation='relu')(x)
x = Dense(64,activation='relu')(x)
output_tensor = Dense(len(classes), activation='softmax')(x)

# Create the model
model = Model(inputs=input_tensor, outputs=output_tensor)

model.summary()

png_file = BASE_PATH+'mobilenet_model.png'
plot_model(model,to_file=png_file,show_shapes=True)
```



MAX, STOCASTIC, AVE

Pooling

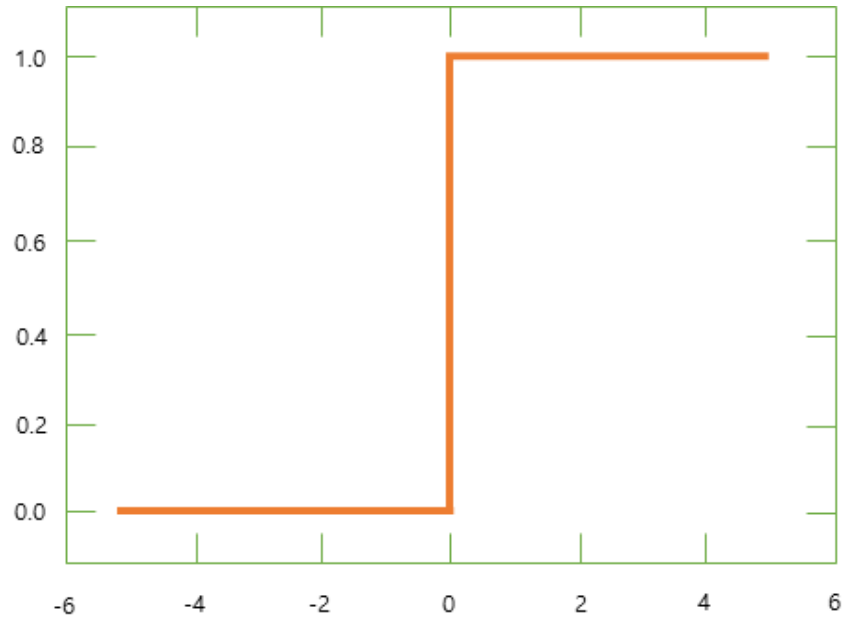




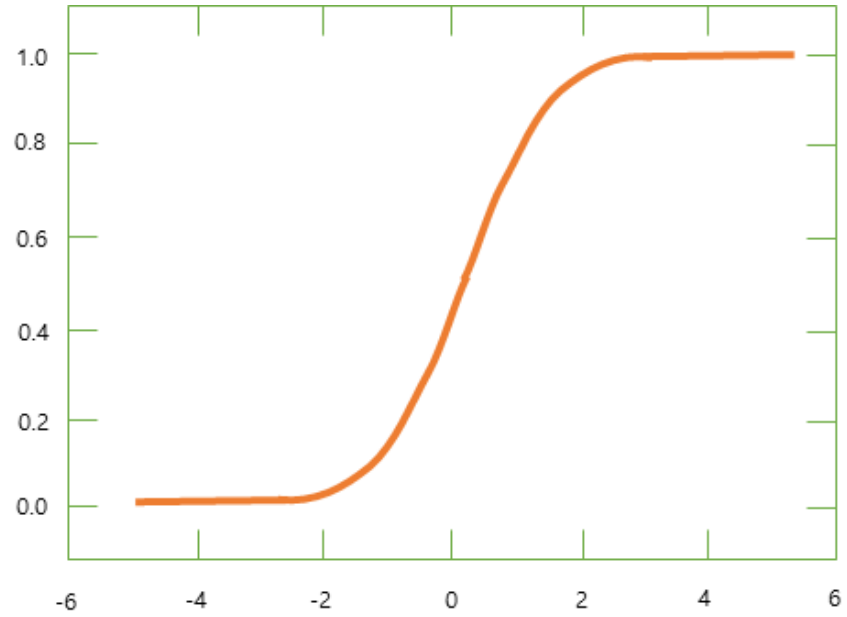
Stride : 2

ReLU

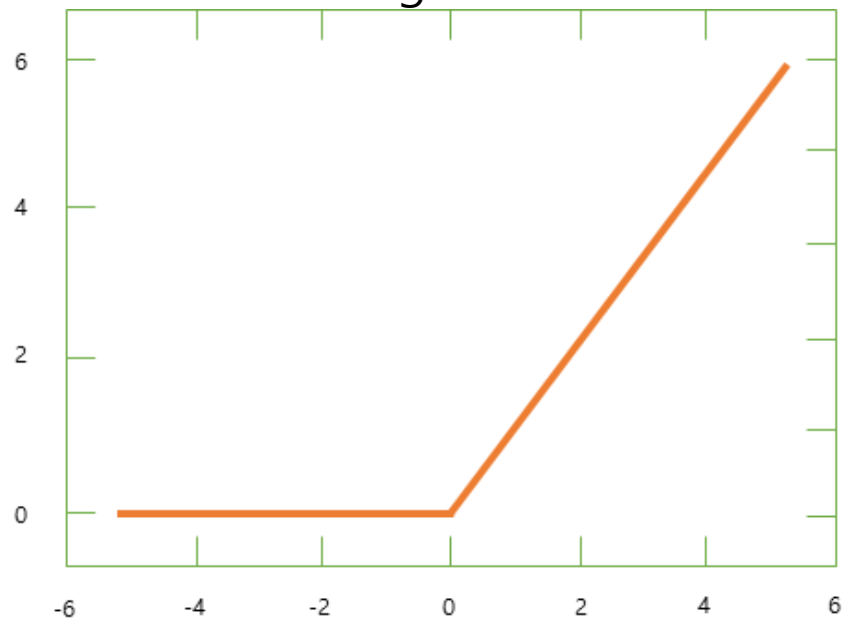
```
layer {
  name: "relu1"
  type: "ReLU"
  bottom: "pool1"
  top: "pool1"
}
```



Step



Sigmoid



ReLU: Rectified Linear Unit

모델 컴파일

[12] 0초

```
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.losses import categorical_crossentropy

model.compile(loss=categorical_crossentropy,
              optimizer=Adam(learning_rate=0.0001), # transfer learning 여기서 학습률을 더 작게
              metrics=['acc'])
```

학습 모델 정보 설정하기

0초

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# 트레인 데이터 augmentation 의로 데이터를 증가 시킵니다.

train_datagen = ImageDataGenerator(rotation_range=30,
                                   rescale=1./255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   width_shift_range=0.2,
                                   height_shift_range=0.2,
                                   horizontal_flip=True,
                                   fill_mode='nearest'
                                   )

train_set = train_datagen.flow_from_directory(BASE_PATH + 'train/',
                                              target_size=(224, 224),
                                              batch_size=16,
                                              shuffle=True,
                                              class_mode='categorical')
```

```
Found 3051 images belonging to 8 classes.
```

0초

```
[14] # 학습중에 validation에 사용할 데이터셋 입니다.

     valid_datagen = ImageDataGenerator(rescale=1./255)

     valid_set = valid_datagen.flow_from_directory(BASE_PATH + 'val/',
                                                   target_size=(224, 224),
                                                   batch_size=16,
                                                   shuffle=True,
                                                   class_mode='categorical')
```

```
Found 800 images belonging to 8 classes.
```

0초

```
# 학습을 완료후에 모델 성능 테스트에 사용할 데이터셋 입니다.

test_datagen = ImageDataGenerator(rescale=1./255)

test_set = test_datagen.flow_from_directory(BASE_PATH + 'val/',
                                            target_size=(224, 224),
                                            batch_size=16,
                                            shuffle=True,
                                            class_mode='categorical')
```

```
Found 800 images belonging to 8 classes.
```

```
# 모델 트레이닝 입니다.

history = model.fit(train_set,
                    steps_per_epoch=train_set.n // train_set.batch_size,
                    epochs=20)
```

```
Epoch 1/20
190/190 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 238s 1s/step - acc: 0.8277 - loss: 0.4850
Epoch 2/20
190/190 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1s 109us/step - acc: 0.6250 - loss: 0.7709
Epoch 3/20
190/190 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 240s 1s/step - acc: 0.8589 - loss: 0.4001
Epoch 4/20
190/190 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1s 97us/step - acc: 0.8750 - loss: 0.3521
Epoch 5/20
190/190 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 242s 1s/step - acc: 0.8619 - loss: 0.3957
Epoch 6/20
190/190 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1s 158us/step - acc: 0.8125 - loss: 0.4186
Epoch 7/20
190/190 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 237s 1s/step - acc: 0.8520 - loss: 0.3978
Epoch 8/20
190/190 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1s 96us/step - acc: 0.8750 - loss: 0.4233
Epoch 9/20
190/190 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 259s 1s/step - acc: 0.8617 - loss: 0.3796
Epoch 10/20
190/190 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1s 125us/step - acc: 0.9375 - loss: 0.2653
Epoch 11/20
190/190 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 258s 1s/step - acc: 0.8807 - loss: 0.3211
Epoch 12/20
```

학습된 모델을 저장

```
model.save(BASE_PATH+'bin/mobilenetv2_class8.h5')

cvt_model = tf.keras.models.load_model(BASE_PATH+'bin/mobilenetv2_class8.h5')
tf.saved_model.save(cvt_model, BASE_PATH+'bin/mobilenetv2_class')
```

```
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `ker
WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to be b
```

test_set.class_indices.items()

```
import pickle

# key 와 value 값을 바꾸어 줍니다.
class8 = dict()
for key,value in test_set.class_indices.items():
    class8[value] = key

with open(BASE_PATH+'bin/class8.pickle', 'wb') as f:
    pickle.dump(class8, f)
```

```
[20] class8
```

```
{0: 'aeroplane',
 1: 'bicycle',
 2: 'bird',
 3: 'boat',
 4: 'bottle',
 5: 'bus',
 6: 'car',
 7: 'cat'}
```

모델 평가하기

```
[28] # 트레인 데이터와 테스트 데이터 셋으로 loss 와 accuracy 측정합니다.
     train_res = model.evaluate(train_set)
     print('Train Loss : {}'.format(train_res[0]))
     print('Train Accuracy : {}'.format(train_res[1]))

     test_res = model.evaluate(test_set)
     print('Test Loss : {}'.format(test_res[0]))
     print('Test Accuracy : {}'.format(test_res[1]))
```

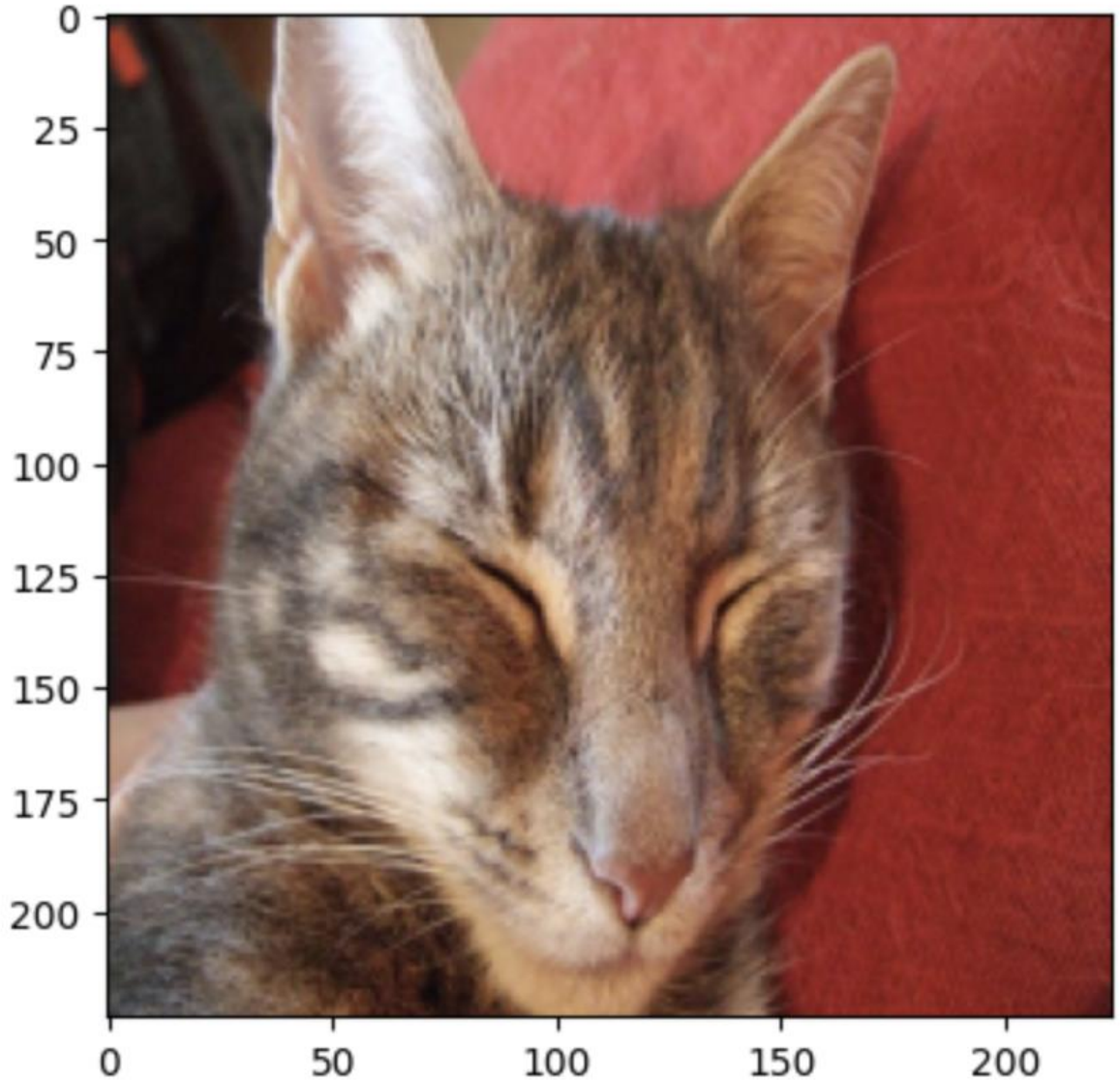
```
191/191 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 191s 998ms/step - acc: 0.8755 - loss: 0.3092
Train Loss : 0.3265800178050995
Train Accuracy : 0.8741396069526672
/usr/local/lib/python3.10/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121:
  self._warn_if_super_not_called()
50/50 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 41s 820ms/step - acc: 0.7713 - loss: 0.9673
Test Loss : 1.02086341381073
Test Accuracy : 0.7337499856948853
```

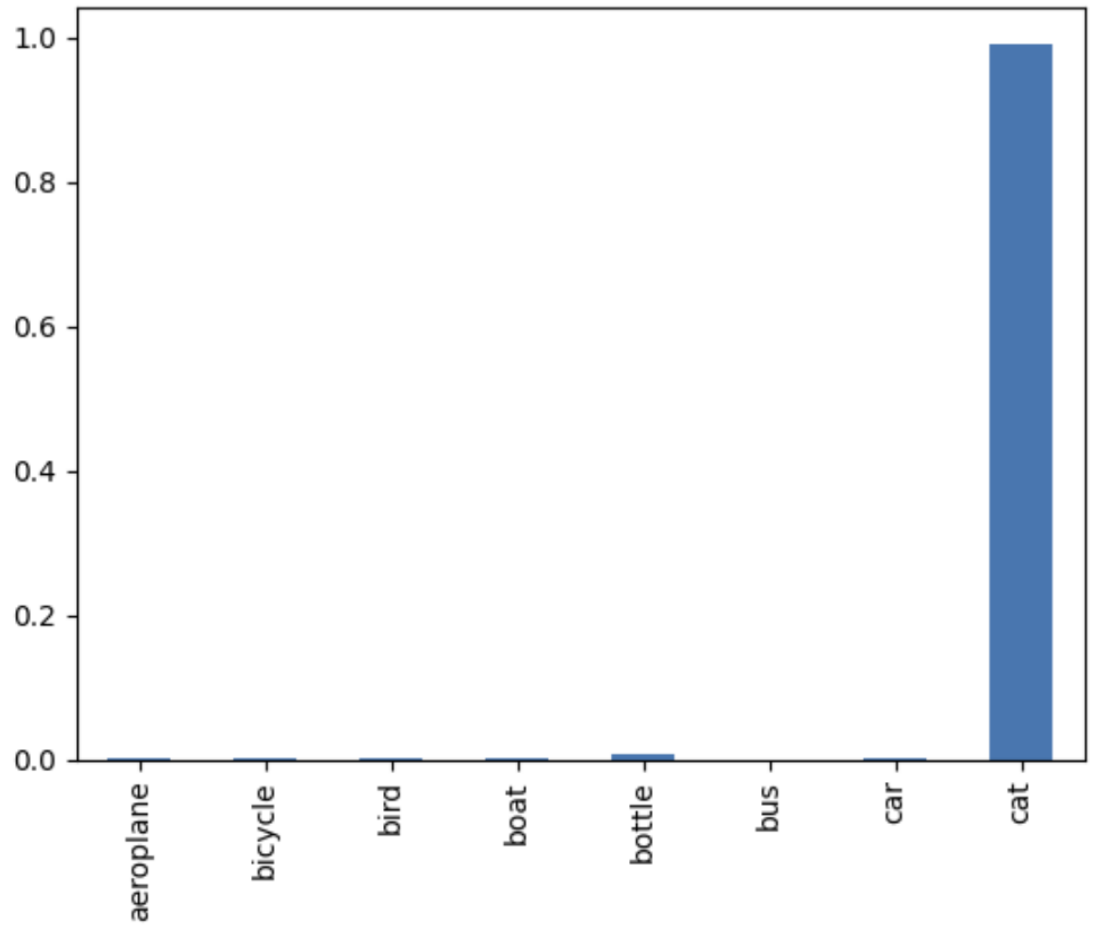
```python
# loss 측정값의 시각화 입니다.

loss = history.history['loss']
acc = history.history['acc']
epochs = range(1,len(loss)+1)

plt.plot(epochs,loss,label='Training Loss')
plt.plot(epochs,acc,label='Training Accuracy')
plt.title('Training Loss and Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Loss or Acc')
plt.legend()

plt.show()
```

모델 테스트해보기

```
import cv2
from tensorflow.keras.models import load_model
import numpy as np

# 추론하기 위한 작업입니다. 모델 설정, 입력 데이터 전처리.
def predict_test_img(path):
    img = cv2.imread(path)

    model = load_model(BASE_PATH+'bin/mobilenetv2_class8.h5')

    print('Original Shape : ',img.shape)

    img = cv2.resize(img, (224,224), fx=0.5, fy=0.5, interpolation=cv2.INTER_AREA)
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    img = img / 255
    print('Resized Shape : ',img.shape)
    plt.imshow(img)
    plt.show()
    class8 = []

    predicted_result = model.predict(np.expand_dims(img,axis=0))
    ans = np.round(predicted_result).astype(int)
    with open(BASE_PATH+'bin/class8.pickle','rb') as f:
        class8 = pickle.load(f)
    res = -1
    for x in range(8):
      if ans[0][x] == 1: res = x
    print('Predict : {}'.format(class8[res]))

    pd.DataFrame(predicted_result,columns=class8.values()).iloc[0].plot(kind='bar')
    plt.show()
```

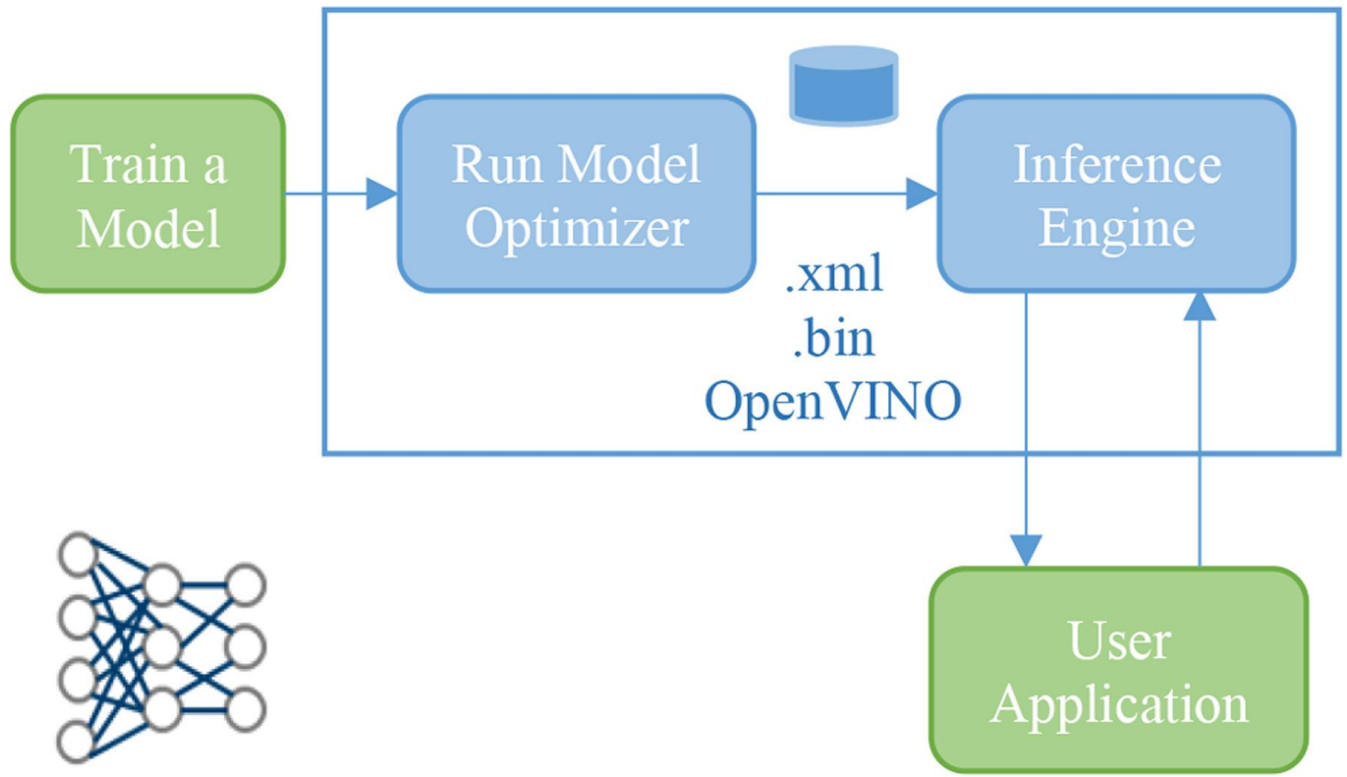
```
  saveable.load_own_variables(weights_store.get(inner_path))
Original Shape :  (375, 500, 3)
Resized Shape :  (224, 224, 3)
```






```
1/1 ━━━━━━━━━━━━━━━━━━━━ 2s 2s/step
Predict : cat
```

AI Hackaton 2


Train a Model
Run Model Optimizer
.xml
.bin
OpenVINO
Inference Engine
User Application

OpenVINO 준비

```
[1]  from google.colab import drive
     drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
[2]  %pip install -q "openvino>=2023.1.0"
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 40.5/40.5 MB 30.4 MB/s eta 0:00:00
```

```
!pip install openvino-dev
```

```
Collecting openvino-dev
  Downloading openvino_dev-2024.3.0-16041-py3-none-any.whl.metadata (16 kB)
Requirement already satisfied: defusedxml>=0.7.1 in /usr/local/lib/python3.10/dist-packages (from openvino-dev) (0.7.1)
Collecting networkx<=3.1.0 (from openvino-dev)
  Downloading networkx-3.1-py3-none-any.whl.metadata (5.3 kB)
Requirement already satisfied: numpy<2.0.0,>=1.16.6 in /usr/local/lib/python3.10/dist-packages (from openvino-dev) (1.26.4)
Requirement already satisfied: openvino-telemetry>=2023.2.1 in /usr/local/lib/python3.10/dist-packages (from openvino-dev) (2024.1.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from openvino-dev) (24.1)
Requirement already satisfied: pyyaml>=5.4.1 in /usr/local/lib/python3.10/dist-packages (from openvino-dev) (6.0.2)
Requirement already satisfied: requests>=2.25.1 in /usr/local/lib/python3.10/dist-packages (from openvino-dev) (2.32.3)
Requirement already satisfied: openvino==2024.3.0 in /usr/local/lib/python3.10/dist-packages (from openvino-dev) (2024.3.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests>=2.25.1->openvino-dev) (3.3.2
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests>=2.25.1->openvino-dev) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests>=2.25.1->openvino-dev) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests>=2.25.1->openvino-dev) (2024.7.4)
Downloading openvino_dev-2024.3.0-16041-py3-none-any.whl (4.7 MB)
```

keras 모델을 Intel OpenVINO IR 파일(xml, bin)로 한다.

```
import openvino
import openvino as ov
from openvino.tools.mo import convert_model

BASE_PATH = './drive/MyDrive/data/'
ov_model = convert_model(BASE_PATH+'bin/mobilenetv2_class')
ov.save_model(ov_model, BASE_PATH+'bin/mobilenetv2_class8.xml')
```

테스트 파일 목록으로 읽어와서 답안 출력하기.

[15]
```
# ---------------------------- Step 1. Initialize OpenVINO Runtime Core ------------
core = ov.Core()
```

[16]
```
# ---------------------------- Step 2. Read a model --------------------------------
# (.xml and .bin files) or (.onnx file)
model = core.read_model(BASE_PATH+'bin/mobilenetv2_class8.xml')

if len(model.inputs) != 1: print('Sample supports only single input topologies')
if len(model.outputs) != 1: print('Sample supports only single output topologies')
```

```python
import os

directory_path = "./drive/MyDrive/data/val"

# Get a list of all files in the directory
folder_list = [f for f in os.listdir(directory_path) if os.path.isdir(os.path.join(directory_path, f))]

# Print the list of files
print("List of files in the directory:")
file_list = []
for folder in folder_list:
    flist = [f for f in os.listdir(directory_path+'/'+folder) if os.path.isfile(os.path.join(directory_path+'/'+folder, f))]
    for file in flist:
        file_list.append(directory_path+'/'+folder+"/"+file)
        print(file)
```

```
2008_000536.jpg
2008_001111.jpg
2008_000116.jpg
2008_002004.jpg
2008_000839.jpg
2008_001640.jpg
2008_005252.jpg
2008_001210.jpg
2008_005003.jpg
2008_004538.jpg
2008_005460.jpg
2008_003045.jpg
2008_003343.jpg
2008_004688.jpg
```

[19]
```python
import pickle

with open(BASE_PATH+'bin/class8.pickle', 'rb') as f:
    labels = pickle.load(f)
```

5분

```
import pandas as pd
import cv2
import numpy as np

# --------------------------- Step 4. Loading model to the device ---------------------------------------------
print('Loading the model to the plugin')
compiled_model = core.compile_model(model, 'CPU')
img_height = 224

result = []

res = open(BASE_PATH+"result.txt", "w")

print('Starting inference in synchronous mode')
cnt = 0;

for file in file_list:
    # --------------------------- Step 5. Set up input --------------------------------------------------------
    # Read input image
    image_path = file
    org_image = cv2.imread(image_path)
    image = cv2.resize(org_image, (img_height,img_height), fx=0.5, fy=0.5, interpolation=cv2.INTER_LINEAR)
    #img = cv2.resize(img, (224,224), fx=0.5, fy=0.5, interpolation=cv2.INTER_AREA)
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    image = image / 255.

    # Add N dimension
    nchw_tensor = np.expand_dims(image, 0)

    # NHWC -> NCHW
    input_tensor = np.transpose(nchw_tensor, (0,1,2,3))#(0,3,1,2))

    # --------------------------- Step 6. Create infer request and do inference synchronously -----------------
    results = compiled_model.infer_new_request({0:input_tensor})
```

```
    # ---------------------------- Step 7. Process output ----------------------------------------------------------
    predictions = next(iter(results.values()))
    output_node_name = next(iter(results.keys())) #

    # Change a shape of a numpy.ndarray with results to get another one with one dimension
    probs = predictions.reshape(-1)

    # Get an array of 8 class IDs in descending order of probability
    top_8 = np.argsort(probs)[-8:][::-1]
    print(top_8)

    header = 'class_id            probability'
    print(f'Image path: {image_path}')
    print('Top 8 results: ')
    print(header)
    print('-' * len(header))

    for class_id in top_8:
        probability_indent = ' ' * (len('class_id          ') - len(labels[clas
        print(f'{labels[class_id]}{probability_indent}{probs[class_id]*100:.7f}'

    res.write(file+':top1 = ('+labels[top_8[0]]+')'+str(probs[top_8[0]]*100)+',

res.close()
```

```
스트리밍 출력 내용이 길어서 마지막 5000줄이 삭제되었습니다.
bottle              99.9633908
boat                0.0126524
bicycle             0.0089207
cat                 0.0062351
car                 0.0042802
bird                0.0041528
aeroplane           0.0002919
bus                 0.0000783
[4 2 1 0 6 7 3 5]
Image path: ./drive/MyDrive/Colab Notebooks/data/val/bottle/2008_001060.jpg
Top 8 results:
class_id            probability
------------------------------
bottle              87.8898144
bird                11.5332916
bicycle             0.4489214
aeroplane           0.0723466
car                 0.0387620
cat                 0.0143387
boat                0.0020355
bus                 0.0004962
[4 1 6 3 2 7 0 5]
Image path: ./drive/MyDrive/Colab Notebooks/data/val/bottle/2008_002775.jpg
```

공지

1. 구글 계정을 갖고 있을 것 : 구글 코랩 활용
2. 경진대회때는 교육과 다른 데이터셋을 사용
3. 1등~3등은 코드 검사 필수 : 치팅 코드 적발 시 -100점
4. 조별 노트북 개수 제한 없음
5. 동일한 조건의 환경을 적용하기 위해서, 코랩 내에서 학습과 추론 적용할 것
6. 더 좋은 모델 사용이나, pretrained model을 변경하는 것은 허용함.

- 예선 프로젝트 – 본선 참석 팀 선발

1) 3가지 사물을 선정하여 이름 부여
   예) tv, cup, chair

2) 휴대폰으로 사물을 찍어서 이미지세트 만들기

3) 학습하여 최종 학습 이미지 캡처

4) 캡처 파일과 데이터셋(압축파일), 변환된 모델(xml, bin) 제출

◆ 본선 팀 선발 규칙

a. 3가지 이상 데이터셋 존재 유무

b. 추론 Accuracy 높은 순서

- 본선 대회

1) 팀별 드론 제공

2) 6개의 이미지 파일 본선 전 제공

3) 사전에 모니터 화면에 이미지를 띄우고 촬영하거나, 칼라 인쇄하여 촬영하여 데이터셋 구성

4) 사전에 학습하여 Accuracy 높여둔 데이터셋 구성

5) 본선 대회에서 2개의 추가 이미지 공개

6) 대회장에서 기존 6개의 이미지와 2개의 추가 이미지를 드론이나 폰으로 촬영하여 학습

7) 3시 ~ 5시 드론 인식 대회

8) 3시 이후 추가 학습 발각시 실격, 3시전에 학습 중이던 작업은 인정함. 단, 학습중인 모델 완료를 이유로 시간 지연은 불가함.

9) 각 팀별 운행 시간은 5분으로 제한함.