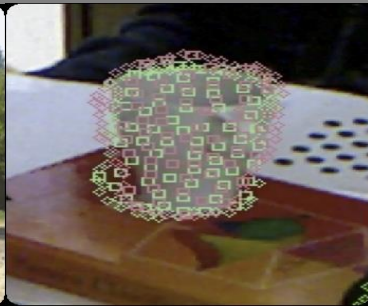
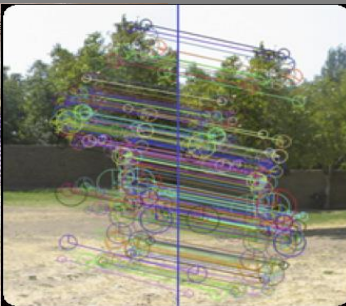


Computer Vision

Deep Learning Basics

(#22: Yolo v5 Object Detection Model Training in Colab)



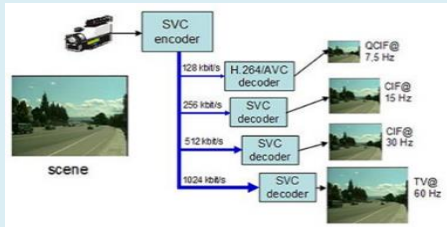
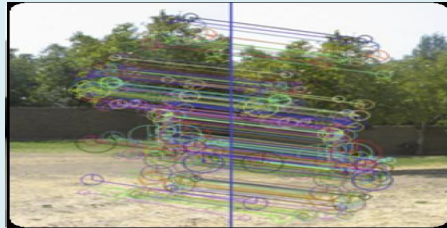
2023 Autumn

Prof. Byung-Gyu Kim
Intelligent Vision Processing Lab. (IVPL)
<http://ivpl.sookmyung.ac.kr>

Dept. of IT Engineering, Sookmyung Women's University
E-mail: bg.kim@sookmyung.ac.kr

Goal of this lecture

- ❖ How to train Yolo object detection API?
 - Preparation of My Dataset
 - Actual Training in Colab
 - How to apply the trained model to my local Tensorflow to detect objects?

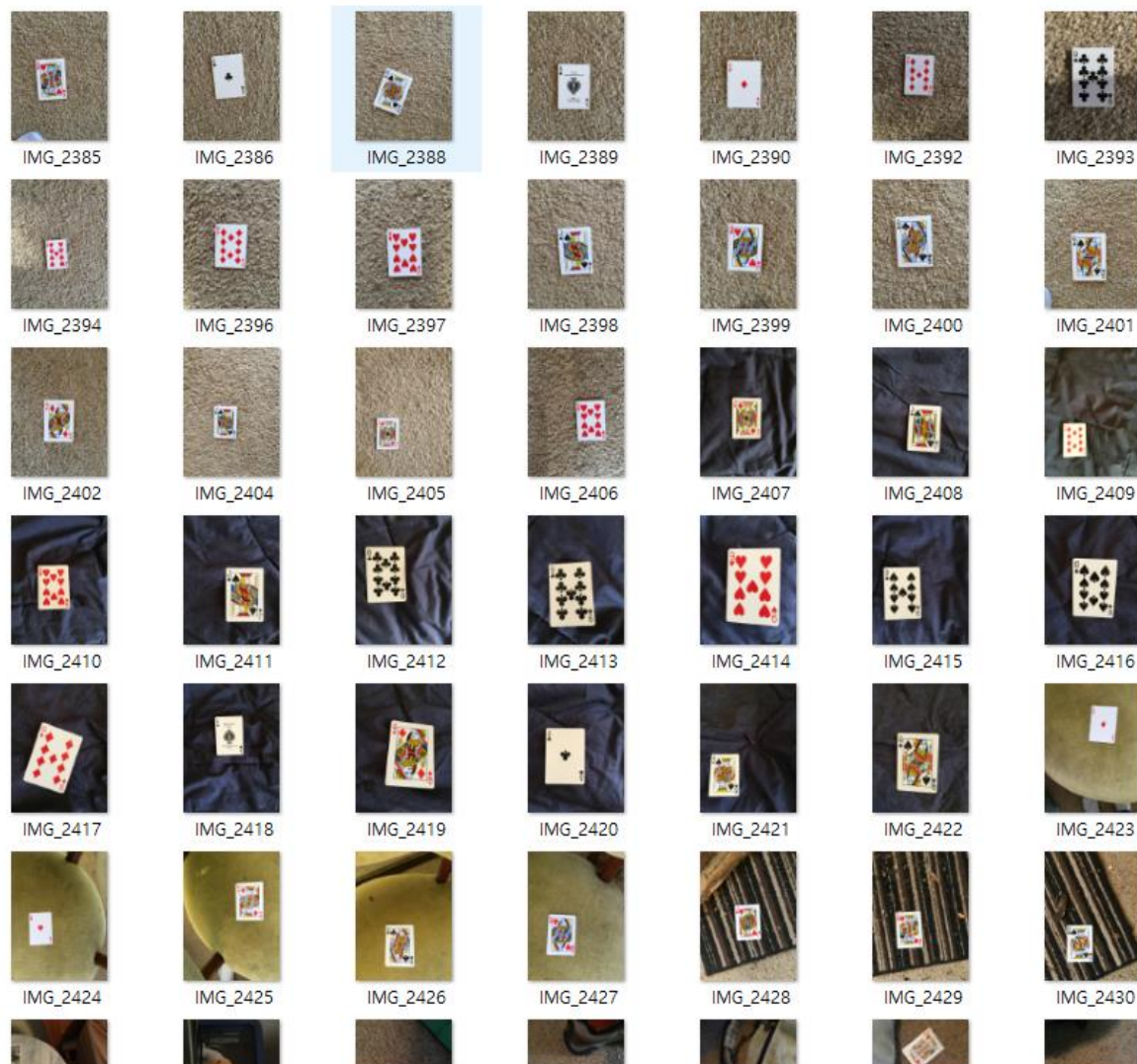


Contents

- My Github
- **Preparation of My Dataset**
- Actual Training Object Detection in Colab
- To my local Tensorflow to detect objects

Preparation of My Dataset (1)

❖ First you'd better to make very large number of photos in "raw" folder.



Preparation of My Dataset (2)

- Make all photos into the same size of images (in Keras):

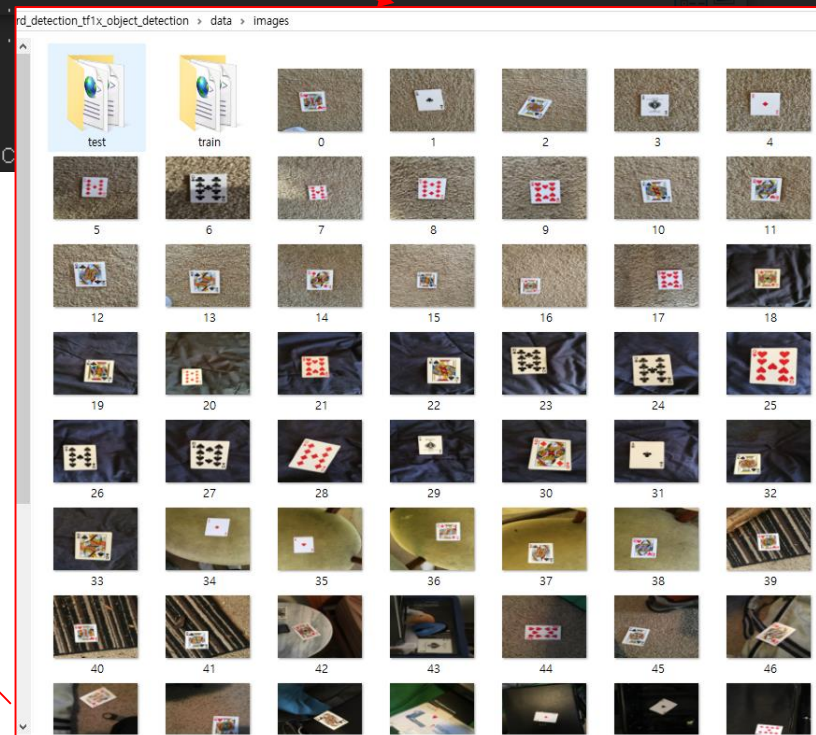
ard_detection_tf1x_object_detection > data

이름	수정된 날짜	유형
images	2019-12-05 오후...	파일 폴더
raw	2019-12-05 오후...	파일 폴더

```
>>python resize_images.py --raw-dir ./data/raw --save-dir ./data/images --ext jpg --target-size "(800, 600)"
```

```
(BGKim) C:\Users\vicl\practices\TensorflowAPI\card_detection_tf1x_object_detection>python resize_images.py --raw-dir ./data/raw --save-dir ./data/images --ext jpg --target-size "(800, 600)"  
251 files to resize from directory `./data/raw` to target size:(800, 600)
```

```
.....  
Done resizing 251 files.  
Saved to directory: `./data/images`  
(BGKim) C:\Users\vicl\practices\TensorflowAPI\card_detection_tf1x_object_detection
```



Preparation of My Dataset (3)

- Split dataset into "train" and "test" folders:
 - Usually, you should give more images into "train"(about 80%).
 - For "test", 20% is enough in usual.

rd_detection_tf1x_object_detection > data > images > train

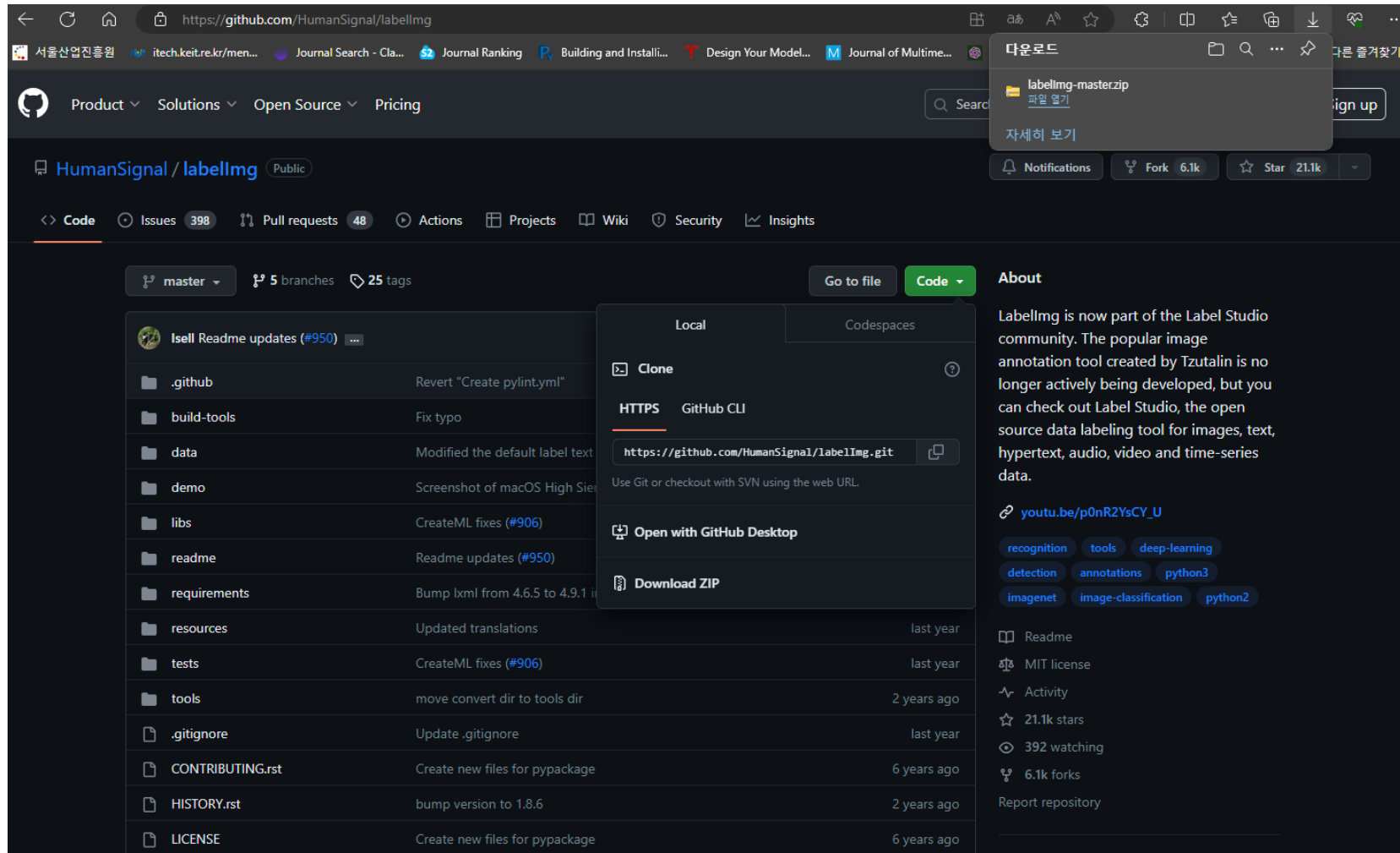
이름	수정된 날짜	유형	크기
0	2019-12-05 오후...	JPG 파일	229KB
0	2019-12-05 오후...	XML 문서	1KB
1	2019-12-05 오후...	JPG 파일	205KB
1	2019-12-05 오후...	XML 문서	1KB
2	2019-12-05 오후...	JPG 파일	225KB
2	2019-12-05 오후...	XML 문서	1KB
3	2019-12-05 오후...	JPG 파일	208KB
3	2019-12-05 오후...	XML 문서	1KB
5	2019-12-05 오후...	JPG 파일	185KB
5	2019-12-05 오후...	XML 문서	1KB
6	2019-12-05 오후...	JPG 파일	186KB
6	2019-12-05 오후...	XML 문서	1KB
10	2019-12-05 오후...	JPG 파일	210KB
10	2019-12-05 오후...	XML 문서	1KB
11	2019-12-05 오후...	JPG 파일	222KB
11	2019-12-05 오후...	XML 문서	1KB
12	2019-12-05 오후...	JPG 파일	230KB
12	2019-12-05 오후...	XML 문서	1KB
13	2019-12-05 오후...	JPG 파일	225KB
13	2019-12-05 오후...	XML 문서	1KB
14	2019-12-05 오후...	JPG 파일	195KB
14	2019-12-05 오후...	XML 문서	1KB
15	2019-12-05 오후...	JPG 파일	230KB
15	2019-12-05 오후...	XML 문서	1KB
16	2019-12-05 오후...	JPG 파일	224KB
16	2019-12-05 오후...	XML 문서	1KB
17	2019-12-05 오후...	JPG 파일	216KB
17	2019-12-05 오후...	XML 문서	1KB
18	2019-12-05 오후...	JPG 파일	124KB
18	2019-12-05 오후...	XML 문서	1KB
20	2019-12-05 오후...	JPG 파일	93KB

rd_detection_tf1x_object_detection > data > images > test

이름	수정된 날짜	유형	크기
210	2019-12-05 오후...	JPG 파일	172KB
210	2019-12-05 오후...	XML 문서	1KB
211	2019-12-05 오후...	JPG 파일	166KB
211	2019-12-05 오후...	XML 문서	1KB
212	2019-12-05 오후...	JPG 파일	182KB
212	2019-12-05 오후...	XML 문서	1KB
213	2019-12-05 오후...	JPG 파일	191KB
213	2019-12-05 오후...	XML 문서	1KB
214	2019-12-05 오후...	JPG 파일	171KB
214	2019-12-05 오후...	XML 문서	1KB
215	2019-12-05 오후...	JPG 파일	172KB
215	2019-12-05 오후...	XML 문서	1KB
216	2019-12-05 오후...	JPG 파일	149KB
216	2019-12-05 오후...	XML 문서	1KB
217	2019-12-05 오후...	JPG 파일	149KB
217	2019-12-05 오후...	XML 문서	1KB
238	2019-12-05 오후...	JPG 파일	196KB
238	2019-12-05 오후...	XML 문서	1KB
239	2019-12-05 오후...	JPG 파일	199KB
239	2019-12-05 오후...	XML 문서	1KB
242	2019-12-05 오후...	JPG 파일	155KB
242	2019-12-05 오후...	XML 문서	1KB
243	2019-12-05 오후...	JPG 파일	194KB
243	2019-12-05 오후...	XML 문서	1KB
244	2019-12-05 오후...	JPG 파일	161KB
244	2019-12-05 오후...	XML 문서	1KB
245	2019-12-05 오후...	JPG 파일	146KB
245	2019-12-05 오후...	XML 문서	1KB

Preparation of My Dataset (4): Data Annotation

- ❖ Data Annotation (object labelling) using “Labellmg” Tool (<https://github.com/HumanSignal/labellmg#usage/>)
 - Download zip file to install it.



The screenshot shows the GitHub repository page for HumanSignal/labellmg. The repository is public and has 21.1k stars and 6.1k forks. The 'Code' dropdown menu is open, showing options to clone the repository using HTTPS, GitHub CLI, or GitHub Desktop, and to download the ZIP file. The 'About' section states that Labellmg is now part of the Label Studio community and is no longer actively being developed. The repository structure includes folders like .github, build-tools, data, demo, libs, readme, requirements, resources, tests, tools, and files like .gitignore, CONTRIBUTING.rst, HISTORY.rst, and LICENSE.

File/Folder	Commit Message	Time
.github	Revert "Create pylint.yml"	
build-tools	Fix typo	
data	Modified the default label text	
demo	Screenshot of macOS High Sierra	
libs	CreateML fixes (#906)	
readme	Readme updates (#950)	
requirements	Bump lxml from 4.6.5 to 4.9.1	
resources	Updated translations	last year
tests	CreateML fixes (#906)	last year
tools	move convert dir to tools dir	2 years ago
.gitignore	Update .gitignore	last year
CONTRIBUTING.rst	Create new files for pypackage	6 years ago
HISTORY.rst	bump version to 1.8.6	2 years ago
LICENSE	Create new files for pypackage	6 years ago

Preparation of My Dataset (4-1): Data Annotation

- Decompress it to your folder and activate your conda account!!!
- Install the required packages as:

```
>> pip install pyqt5  
>> pip install lxml  
>> pyrcc5 -o libs/resources.py resources.qrc
```

- Go to your Labellmg folder!!! And run Labellmg.py like:

```
>> python .\labellmg.py  
or  
>> python Labellmg.py
```

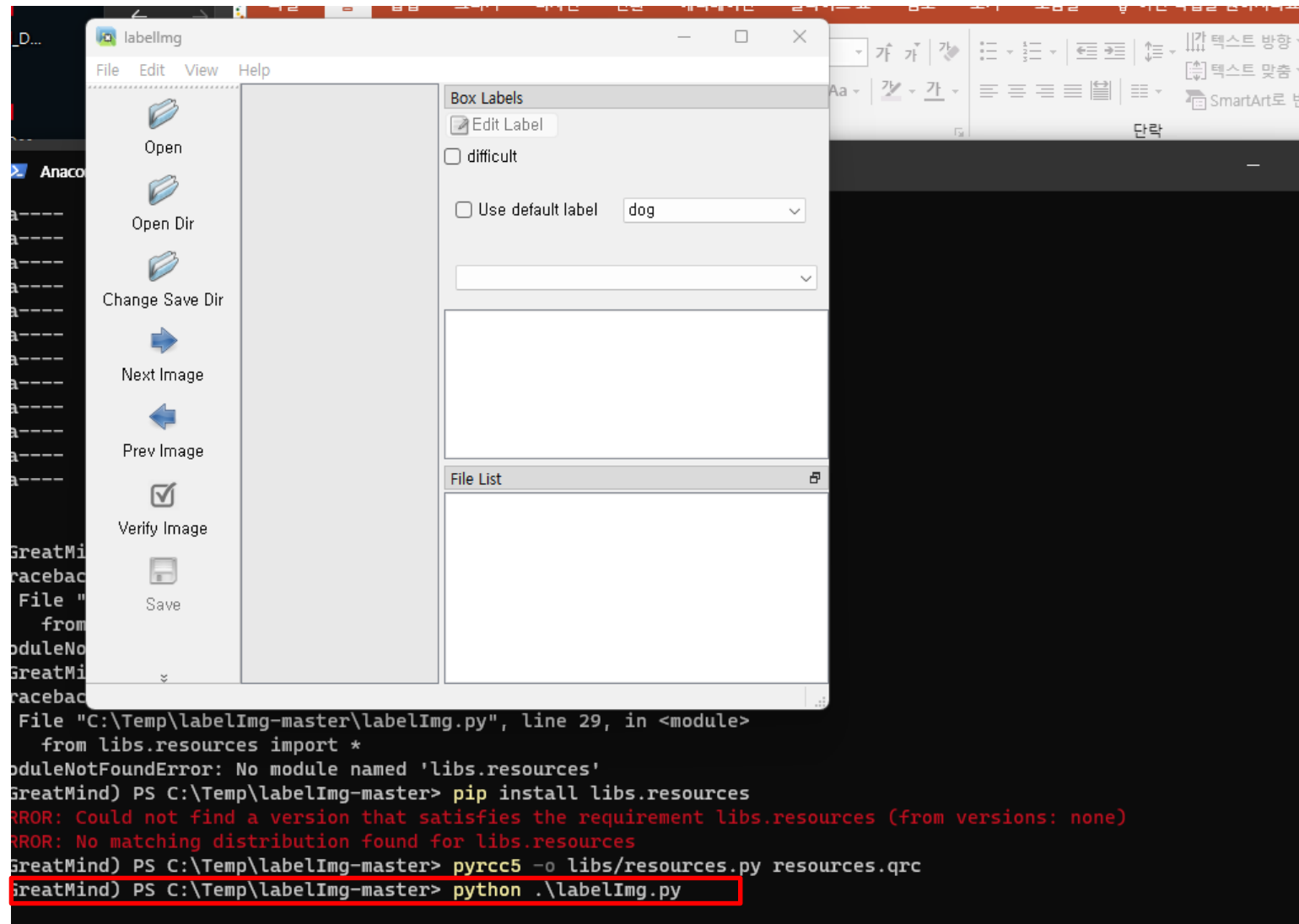
Windows + Anaconda

Download and install [Anaconda](#) (Python 3+)

Open the Anaconda Prompt and go to the [labellmg](#) directory

```
conda install pyqt=5  
conda install -c anaconda lxml  
pyrcc5 -o libs/resources.py resources.qrc  
python labelImg.py  
python labelImg.py [IMAGE_PATH] [PRE-DEFINED CLASS FILE]
```


Preparation of My Dataset (4-2): Data Annotation



The image shows a screenshot of the labelImg software interface and a terminal window. The labelImg window is titled "labelImg" and has a menu bar with "File", "Edit", "View", and "Help". The interface includes a toolbar with buttons for "Open", "Open Dir", "Change Save Dir", "Next Image", "Prev Image", "Verify Image", and "Save". On the right side, there is a "Box Labels" panel with the following options:

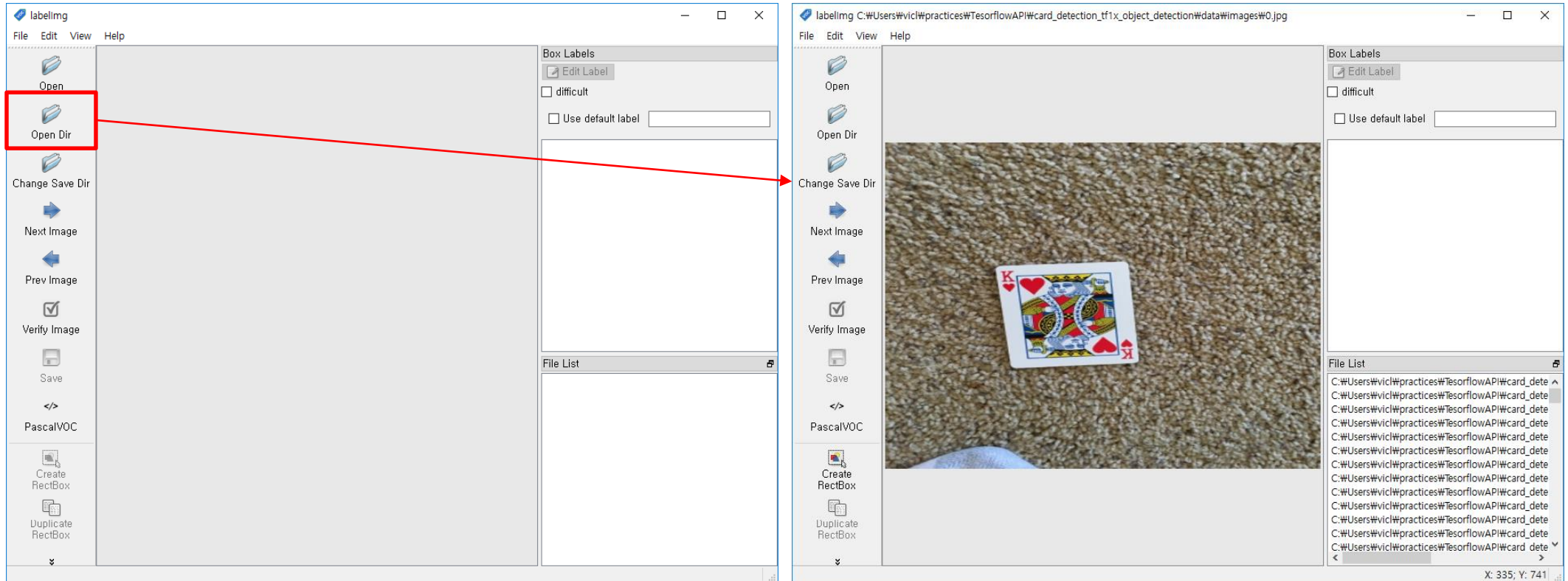
- Edit Label
- difficult
- Use default label: dog

Below the "Box Labels" panel is a "File List" panel. The terminal window shows the following commands and output:

```
File "C:\Temp\labelImg-master\labelImg.py", line 29, in <module>
  from libs.resources import *
ModuleNotFoundError: No module named 'libs.resources'
GreatMind) PS C:\Temp\labelImg-master> pip install libs.resources
ERROR: Could not find a version that satisfies the requirement libs.resources (from versions: none)
ERROR: No matching distribution found for libs.resources
GreatMind) PS C:\Temp\labelImg-master> pyrc5 -o libs/resources.py resources.qrc
GreatMind) PS C:\Temp\labelImg-master> python .\labelImg.py
```

Preparation of My Dataset (5): Data Annotation

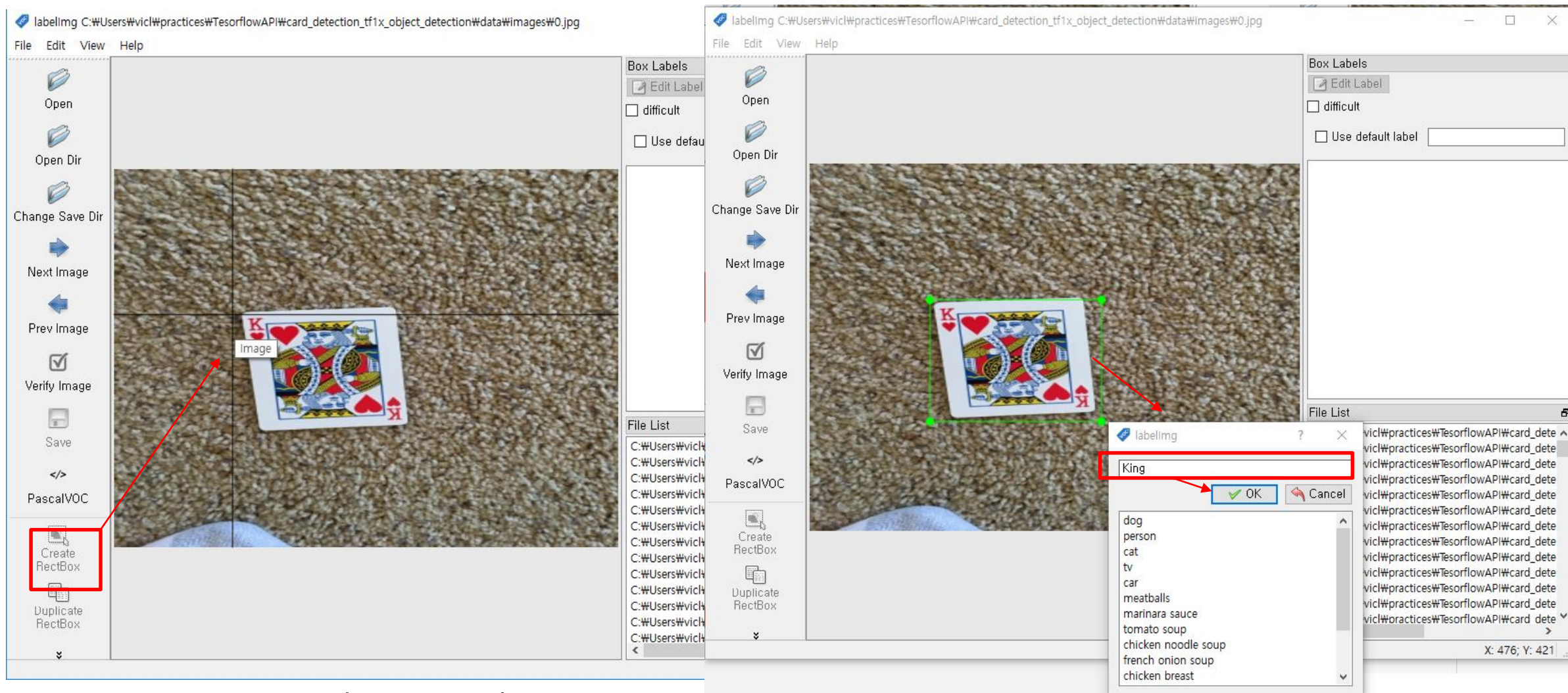
- Run Labellmg.exe



- Next Image: loading the next image
- Prev Image: go to the previous one

Preparation of My Dataset (6): Data Annotation

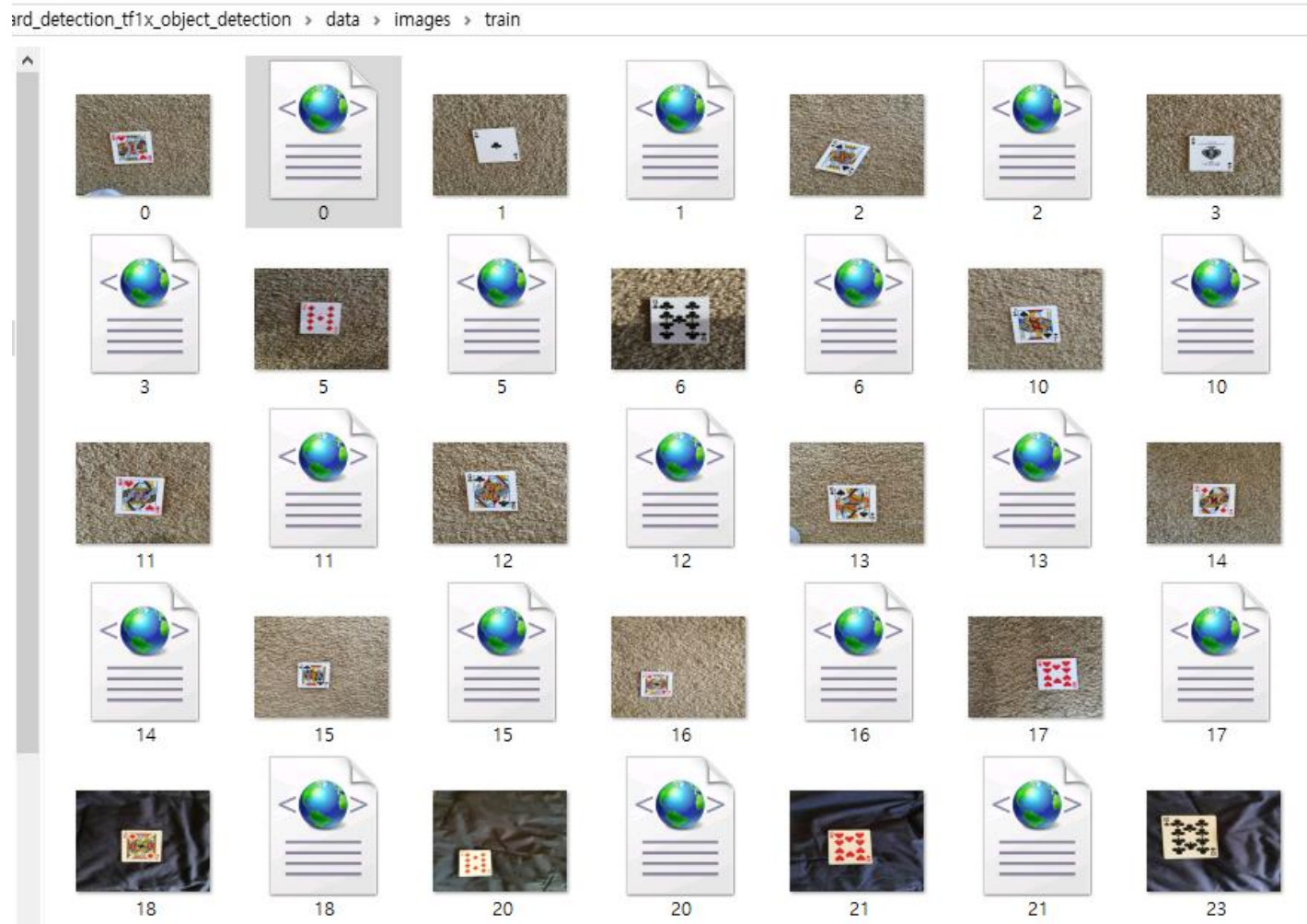
- Image labeling (annotation)



“Create RectBox” → select your object region with mouse click.

Preparation of My Dataset (7): Data Annotation

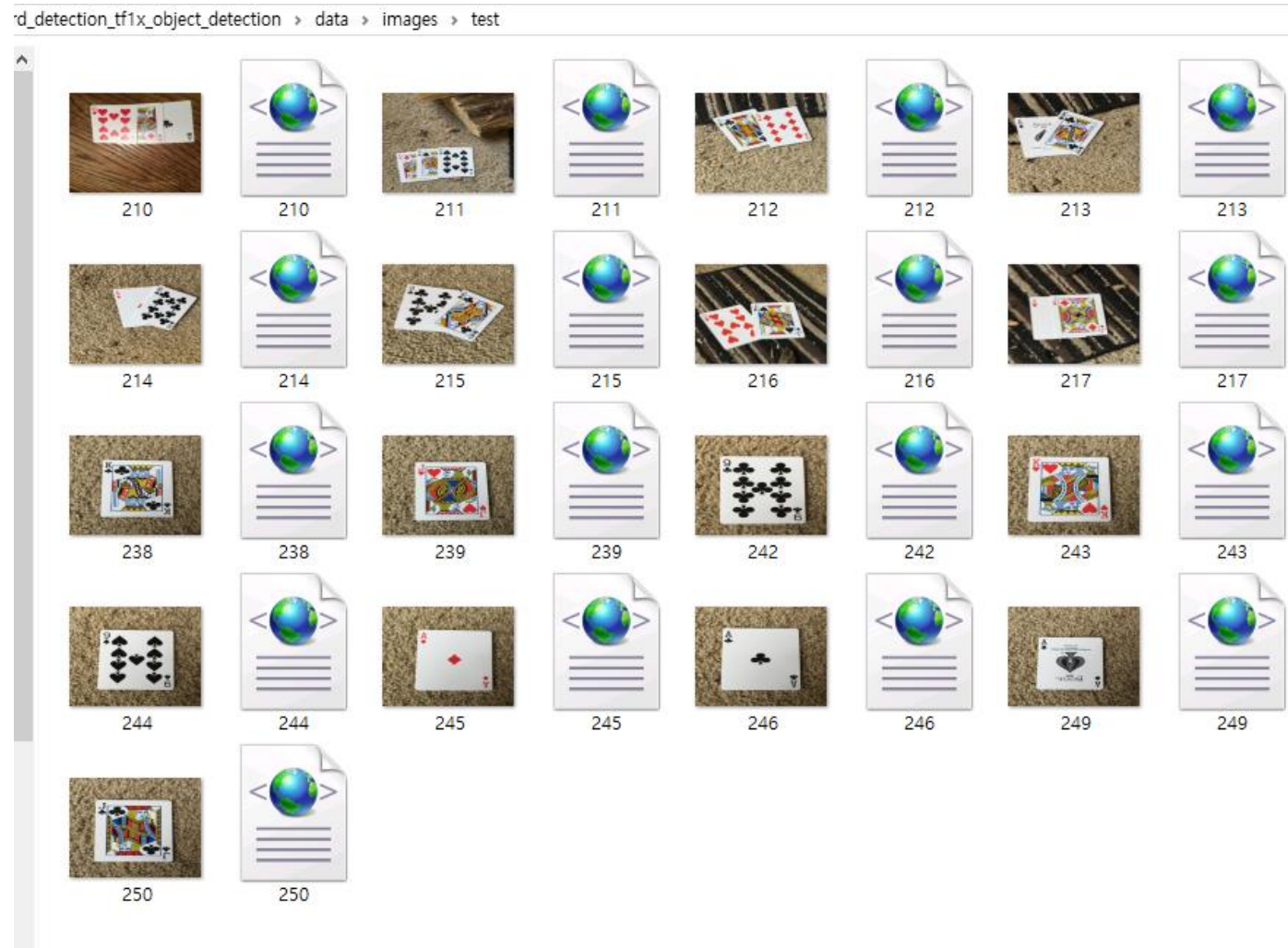
- After clicking "save" button, you can see what is change in "open dir".



For each image, you can save information in each xml file.

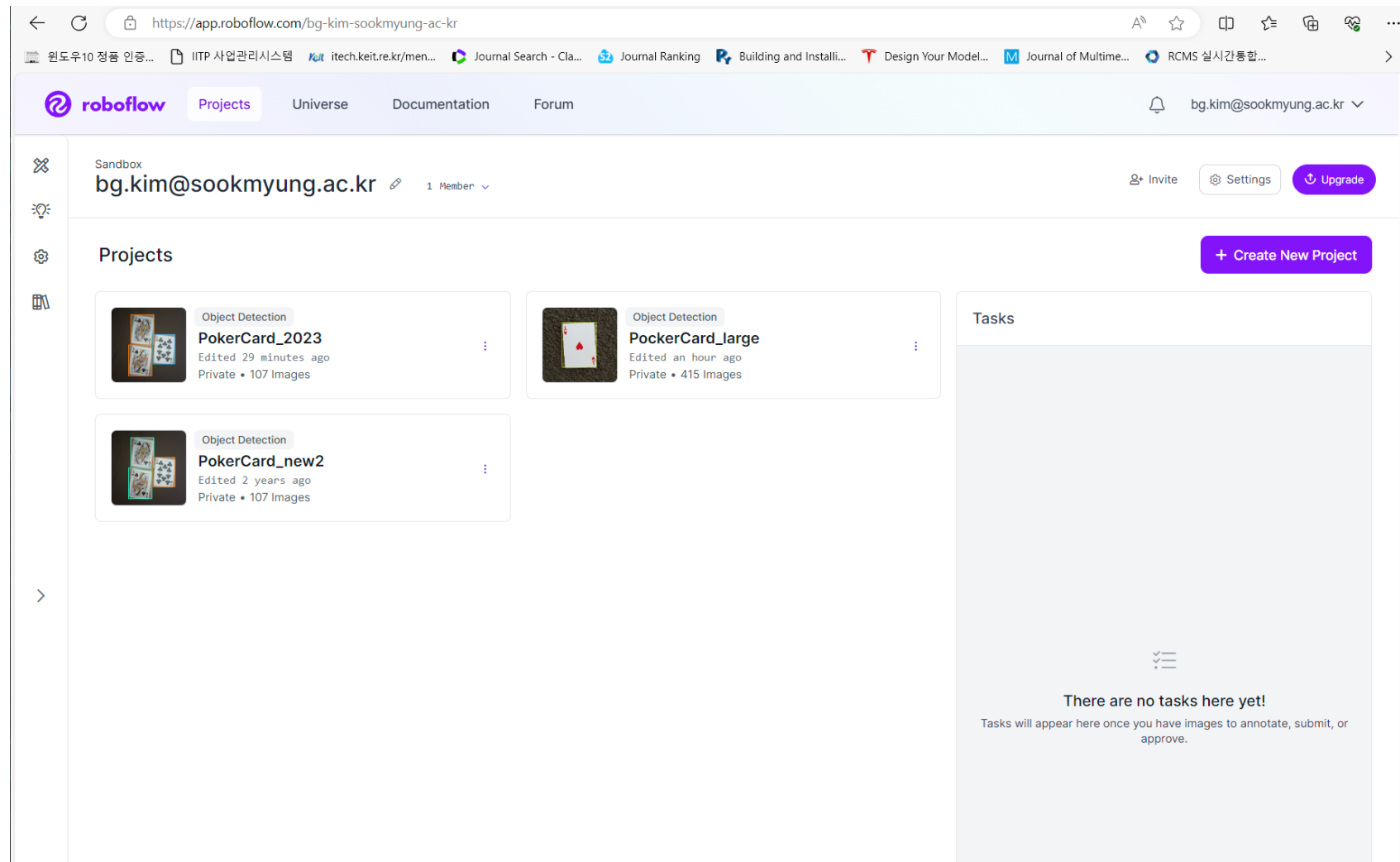
Preparation of My Dataset (8): Data Annotation

- For all your data, you have to give annotations (both of train and test datasets).



Preparation of My Dataset (8): Upload your dataset (annotated) to Roboflow

- ❖ Robflow AI (<https://app.roboflow.com/>)
 - Join (make your account) and login!



The screenshot displays the Roboflow AI web application interface. The browser address bar shows the URL <https://app.roboflow.com/bg-kim-sookmyung-ac-kr>. The page header includes the Roboflow logo, navigation links for 'Projects', 'Universe', 'Documentation', and 'Forum', and a user profile dropdown for 'bg.kim@sookmyung.ac.kr'. The main content area is titled 'Sandbox' and shows the user's profile 'bg.kim@sookmyung.ac.kr' with '1 Member'. A '+ Create New Project' button is visible. Under the 'Projects' section, three project cards are listed: 'PokerCard_2023' (Object Detection, Edited 29 minutes ago, Private, 107 Images), 'PokerCard_large' (Object Detection, Edited an hour ago, Private, 415 Images), and 'PokerCard_new2' (Object Detection, Edited 2 years ago, Private, 107 Images). A 'Tasks' section on the right is currently empty, displaying the message: 'There are no tasks here yet! Tasks will appear here once you have images to annotate, submit, or approve.'

Preparation of My Dataset (8): Upload your dataset (annotated) to Roboflow

- Click "Create New Project"

The screenshot displays the Roboflow web application interface. The main page shows a user profile for 'bg.kim@sookmyung.ac.kr' and a list of existing projects under the 'Projects' section, including 'PokerCard_2023', 'PockerCard_large', and 'PokerCard_new2'. A red box highlights the '+ Create New Project' button in the top right corner of the main interface. A modal window titled 'Create New Project' is open, showing the following details:

- Project Type:** Three options are shown: 'Object Detection' (Find multiple things and their specific location), 'Classification' (Assign labels to the entire image), and 'Instance Segmentation' (Detect multiple objects and their actual shape). The 'Object Detection' option is highlighted with a purple border.
- Project Name:** The text 'PokerCard_2023_2' is entered in the input field.
- What are you detecting?:** The text 'cards' is entered in the input field.
- Buttons:** 'Cancel' and 'Create Private Project' buttons are at the bottom. The 'Create Private Project' button is highlighted with a red box.

A red arrow points from the '+ Create New Project' button in the main interface to the 'Create Private Project' button in the modal.


Preparation of My Dataset (8): Upload your dataset (annotated) to Roboflow

- Upload your dataset: Use "Select Folder" and upload all data files where you choose.

Upload [Want to change the classes on your annotated images?](#)

Batch Name: [MODIFYING UPLOADED IMAGES](#)
If you need to crop, resize, rotate or remap classes, you can modify them with a pre-processing step called 'Modify Classes' when generating a dataset in the 'Generate' tab.
[More Info on Pre-Processing Steps >>](#)

Tags: [?](#)



Drag and drop images and annotations to upload them.

OR

[Select Files](#) [Select Folder](#)

Need images to get started? We've got you covered.

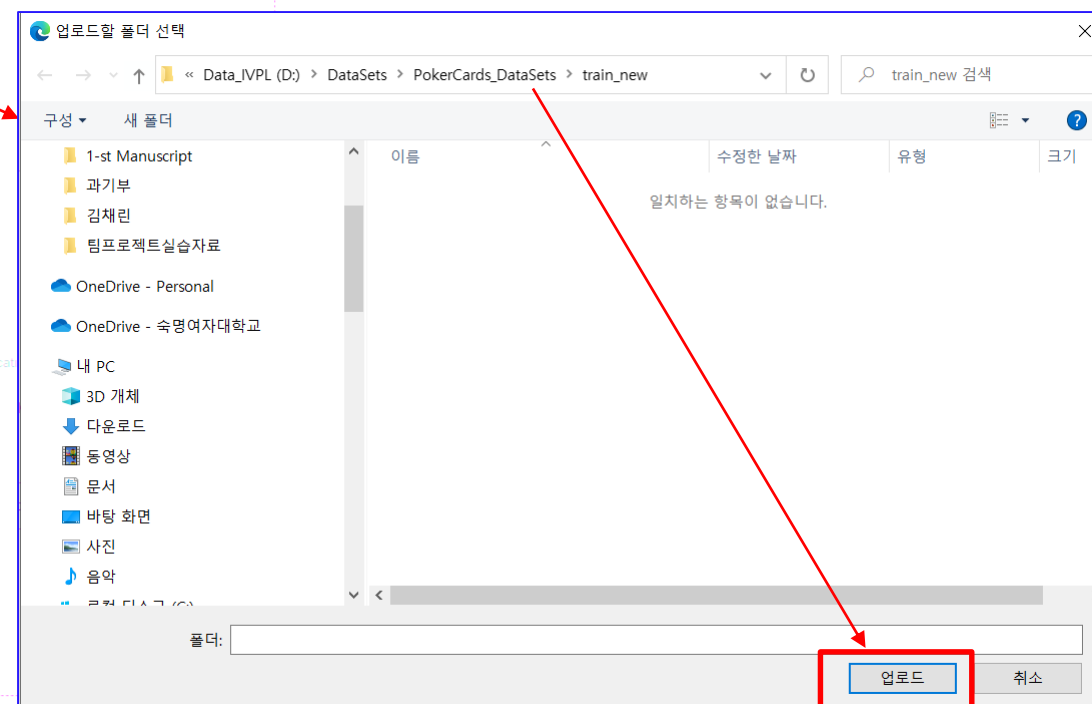
[Import YouTube Video:](#)

[Find a Universe Dataset →](#)
Browse over 100k free datasets for images and build a model in minutes.

[Integrate Our API →](#)
Collect real world images directly from your existing applications.

[Upload images directly from cloud storage →](#)
Upload images and annotations directly from AWS, Azure or GCP.

SUPPORTED FORMATS



Preparation of My Dataset (8): Upload your dataset (annotated) to Roboflow

- Check your uploaded data files

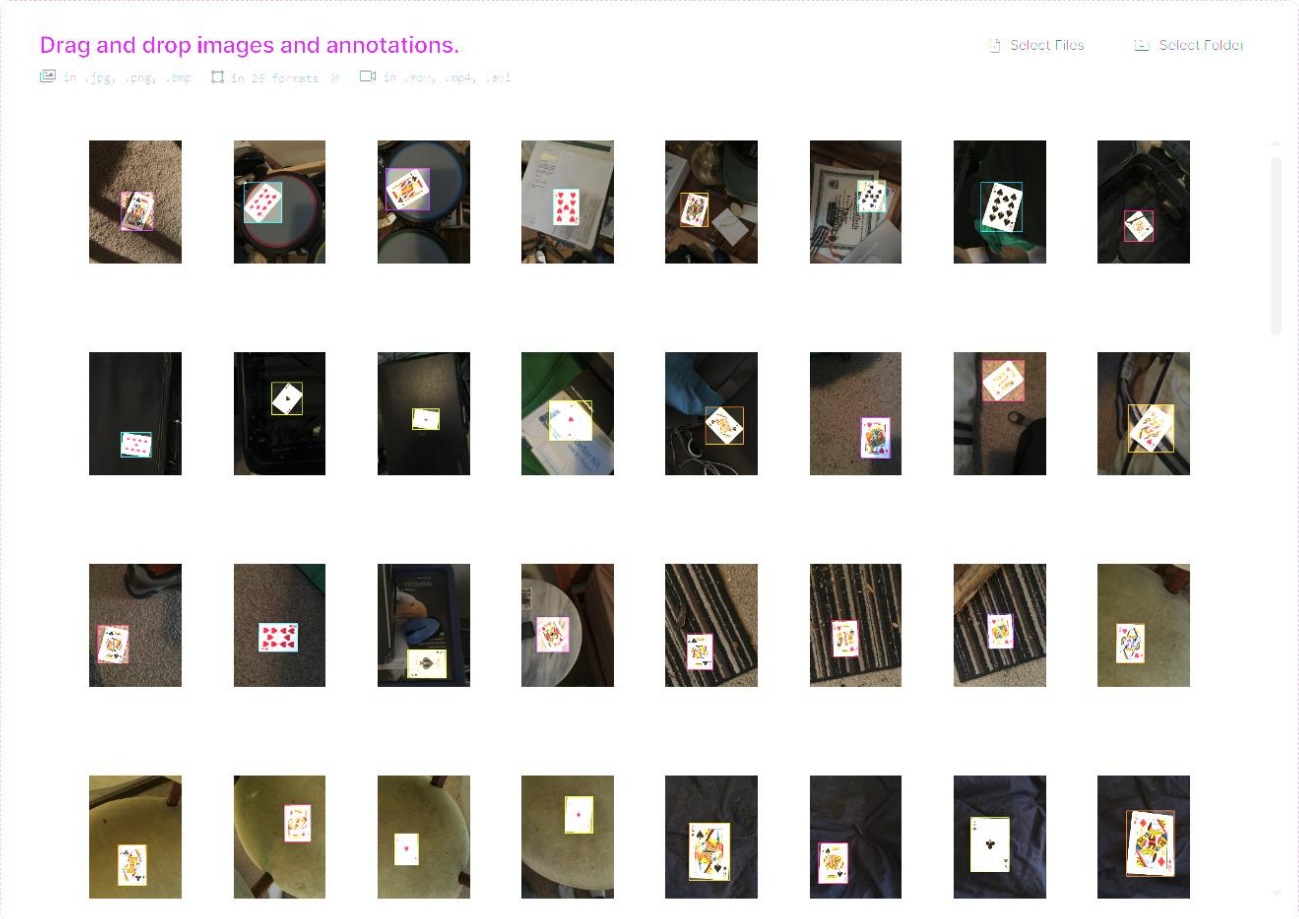
Upload [Want to change the classes on your annotated images?](#) [Save and Continue](#)

Batch Name: Uploaded on 11/29/23 at 7:45 pm Tags: [+](#)

All Images **107** Annotated 107 Not Annotated 0

Drag and drop images and annotations. [Select Files](#) [Select Folder](#)

[In .jpg, .png, .bmp](#) [In 25 formats >](#) [In .mov, .mp4, .avi](#)



The screenshot displays the Roboflow web interface for dataset management. At the top, there is an 'Upload' button with a link to change classes, and a 'Save and Continue' button. Below this, the batch name is 'Uploaded on 11/29/23 at 7:45 pm' and there are tags. A navigation bar shows 'All Images' with a count of 107, 'Annotated' with 107, and 'Not Annotated' with 0. The main area is titled 'Drag and drop images and annotations.' and contains a grid of 32 small image thumbnails, each with a bounding box around a playing card. Above the grid are 'Select Files' and 'Select Folder' buttons. Below the grid are file format options: 'In .jpg, .png, .bmp', 'In 25 formats >', and 'In .mov, .mp4, .avi'.

Preparation of My Dataset (8): Upload your dataset (annotated) to Roboflow

- Check your uploaded data files

Upload [Want to change the classes on your annotated images?](#)

Batch Name: Uploaded on 11/29/23 at 7:45 pm Tags: [?](#)

All Images **107** Annotated 187 Not Annotated 8

Drag and drop images and annotations. [Select Files](#) [Select Folder](#)

in .jpg, .png, .bmp in 25 formats in .mov, .mp4, .avi

How should we split these images?

Choose one [? What's Train, Valid, Test?](#)

Split Images Between Train/Valid/Test

Train 70% Valid 20% Test 10%

[Learn more on our blog.](#)

Cancel Continue

Preparation of My Dataset (8): Upload your dataset (annotated) to Roboflow

- Finish your dataset generation: Go to "5-Generate"

PokerCard_2023_2 Dataset

Generating New Version

Prepare your images and data for training by compiling them into a version. Experiment with different configurations to achieve better training results.

- ✓ **Source Images** Images: 107
Classes: 6
Unannotated: 0
- ✓ **Train/Test Split** Training Set: 75 images
Validation Set: 21 images
Testing Set: 11 images
- 3 **Preprocessing**
[What can preprocessing do?](#)
Decrease training time and increase performance by applying image transformations to all images in this dataset.
 - Auto-Orient [Edit](#) x
 - Resize
Stretch to 640×640 [Edit](#) x
 - [+ Add Preprocessing Step](#)
- 4 **Augmentation**
- 5 **Generate**

PokerCard_2023_2 Dataset

Generating New Version

Prepare your images and data for training by compiling them into a version. Experiment with different configurations to achieve better training results.

- ✓ **Source Images** Images: 107
Classes: 6
Unannotated: 0
- ✓ **Train/Test Split** Training Set: 75 images
Validation Set: 21 images
Testing Set: 11 images
- ✓ **Preprocessing** Auto-Orient: Applied
Resize: Stretch to 640×640
- ✓ **Augmentation** Turned Off
- 5 **Generate**
Review your selections then click "Generate" to create a moment-in-time snapshot of your dataset with the applied preprocessing steps.
Maximum Version Size: 107
[See how this is calculated >>](#)

Preparation of My Dataset (8): Upload your dataset (annotated) to Roboflow

- You can see the final processing for your dataset.

PokerCard_2023_2 Image Dataset

Generate New Version

v1 2023-11-29 7:58pm
Generated on Nov 29, 2023

VERSIONS

2023-11-29 7:58pm
v1 Nov 29, 2023

The images for your new dataset version are now being created and processed. All of the images are being processed.

Initializing

PokerCard_2023_2 Image Dataset

Generate New Version

v1 2023-11-29 7:58pm
Generated on Nov 29, 2023

Export Dataset

VERSIONS

2023-11-29 7:58pm
v1 Nov 29, 2023

This version doesn't have a model.

Train an optimized, state of the art model with Roboflow or upload a custom trained model to use features like Label Assist and Model Evaluation and deployment options like our auto-scaling API and edge device support.

Train with Roboflow Custom Train and Upload

Available Credits: 3

107 Total Images [View All Images →](#)

Dataset Split

TRAIN SET	70%	VALID SET	20%	TEST SET	10%
75 Images		21 Images		11 Images	

Preprocessing Auto-Orient: Applied
Resize: Stretch to 640x640

Augmentations No augmentations were applied.

Preparation of My Dataset (8): Upload your dataset (annotated) to Roboflow

❖ Important Variables here....!!!!!!

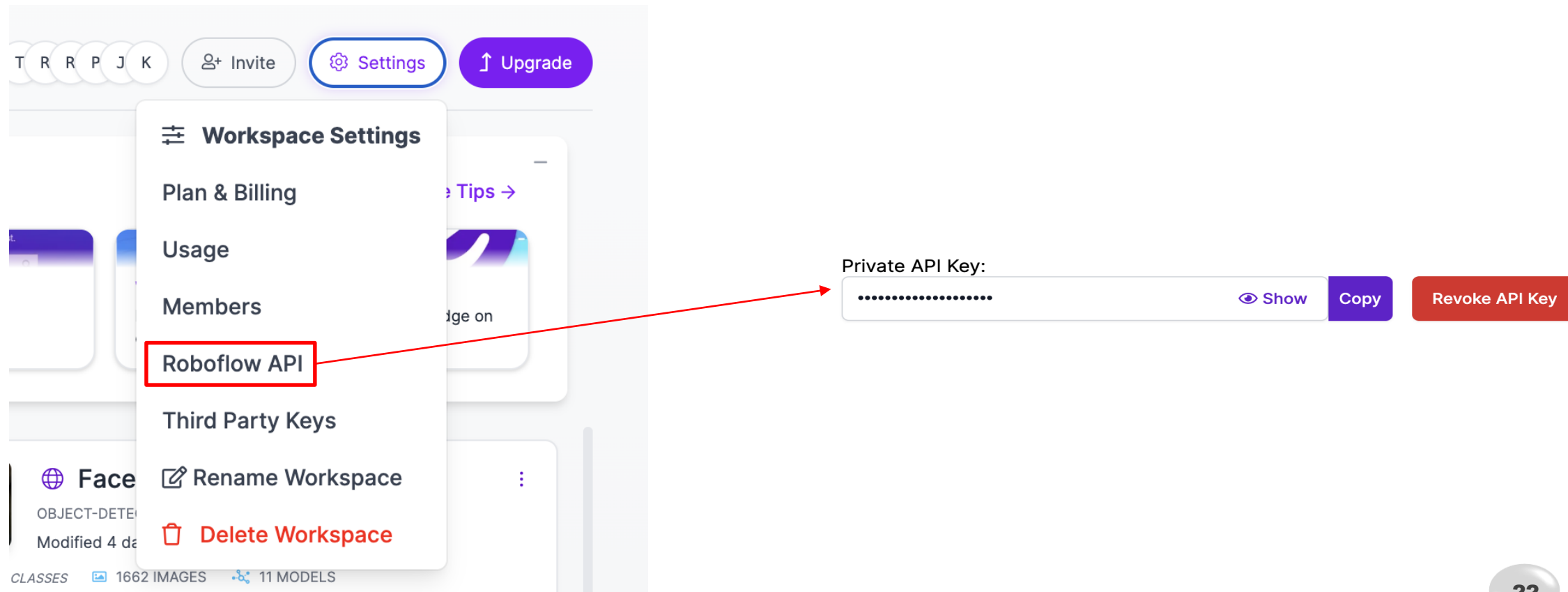
- API_KEY
- **Workspace ID** - - - - -
- **Project ID** - - - - -

The screenshot shows the Roboflow web interface for a dataset named 'PokerCard_2023_2'. The browser address bar shows the URL: `https://app.roboflow.com/bg-kim-sookmyung-ac-kr/pokercard_2023_2`. The workspace ID 'bg-kim-sookmyung-ac-kr' is highlighted with a red dashed line, and the project ID 'pokercard_2023_2' is highlighted with a blue dashed line. The page displays the dataset name, a 'Generate New Version' button, and a version history table with one entry: '2023-11-29 7:58pm v1 Nov 29, 2023'. A 'Train with Roboflow' button is visible, along with a note that the version doesn't have training support. At the bottom, it shows '107 Total Images' and a preview of four images with bounding boxes.

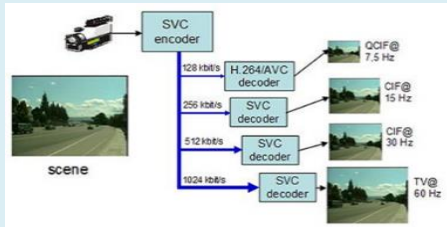
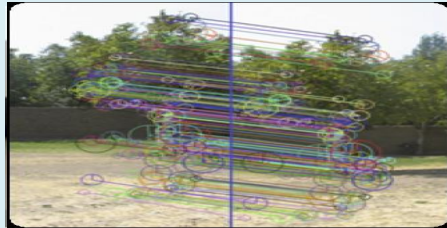
Preparation of My Dataset (9): Upload your dataset (annotated) to Roboflow

❖ User API Key

- First, go to the [Roboflow dashboard](#). Then, select the Roboflow API tab from the settings page's sidebar navigation:
- Finally, **copy your Private API Key**. Be sure to keep this secret. Treat it like a password; it grants the bearer access to all of your workspace's data and models



The image shows a screenshot of the Roboflow dashboard. At the top, there are navigation buttons: 'T R R P J K', 'Invite', 'Settings', and 'Upgrade'. A 'Workspace Settings' dropdown menu is open, listing options: 'Plan & Billing', 'Usage', 'Members', 'Roboflow API' (highlighted with a red box), 'Third Party Keys', 'Rename Workspace', and 'Delete Workspace'. A red arrow points from the 'Roboflow API' option to a 'Private API Key' field. The field contains a masked key (dots) and has 'Show', 'Copy', and 'Revoke API Key' buttons next to it. At the bottom left, there is a status bar with 'CLASSES', '1662 IMAGES', and '11 MODELS'.



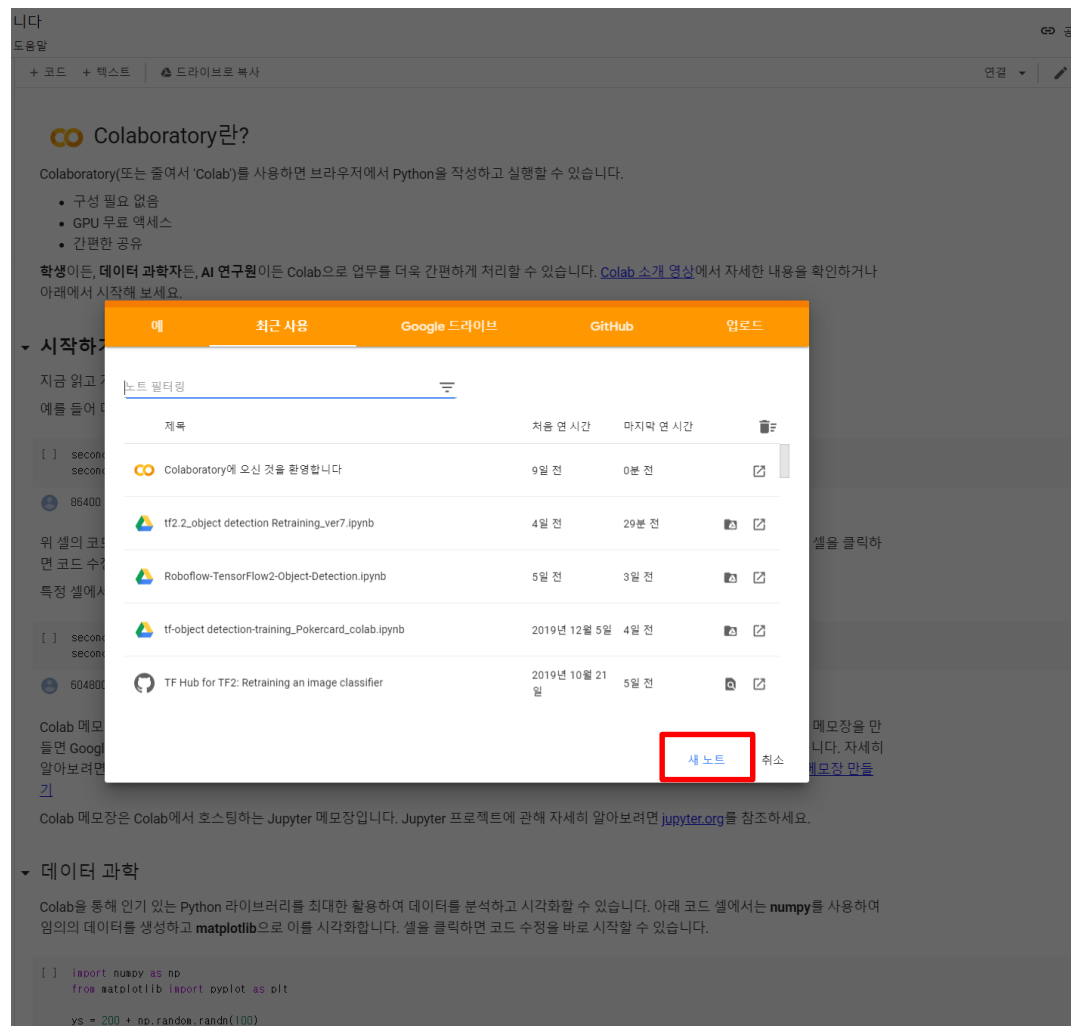
Contents

- My Github
- Preparation of My Dataset
- **Actual Training Object Detection in Colab**
- To my local Tensorflow to detect objects

Actual Training in Colab (1): Yolo v5

❖ Connect your colab (jupyter Notebook) with your account in Google.

- First you have to log-in Google.
- Type "<https://colab.research.google.com/>" in your broser. → "새노트" or 기존 노트 선택하면 됨



The screenshot shows the Google Colaboratory web interface. At the top, there's a header with the Colab logo and the text 'Colaboratory란?'. Below this, there's a list of features: '구성 필요 없음', 'GPU 무료 액세스', and '간편한 공유'. A section titled '시작하기' (Getting Started) is visible, with a sub-section '지금 읽고 싶은 것들' (What you might want to read now) containing a table of recent notebooks. The table has columns for '제목' (Title), '최근 연 시간' (Last modified), and '마지막 연 시간' (Last accessed). The '새 노트' (New Notebook) button is highlighted with a red box at the bottom right of the dialog.

제목	최근 연 시간	마지막 연 시간
Colaboratory에 오신 것을 환영합니다	9일 전	0분 전
tf2.2_object detection Retraining_ver7.ipynb	4일 전	29분 전
Roboflow-TensorFlow2-Object-Detection.ipynb	5일 전	3일 전
tf-object detection-training_Pokercard_colab.ipynb	2019년 12월 5일	4일 전
TF Hub for TF2: Retraining an image classifier	2019년 10월 21일	5일 전

Actual Training in Colab (2): Yolo v5

❖ Download YOLOv5 source code in PyTorch platform.

- Check on the download status.

```
!git clone https://github.com/ultralytics/yolov5 # clone
%cd yolov5
!pip install -r requirements.txt # install
```

```
ls -al
```

```
total 376
drwxr-xr-x 9 root root 4096 Nov 30 07:40 ./
drwxr-xr-x 1 root root 4096 Nov 30 07:40 ../
-rw-r--r-- 1 root root 8009 Nov 30 07:40 benchmarks.py
-rw-r--r-- 1 root root 393 Nov 30 07:40 CITATION.cff
drwxr-xr-x 2 root root 4096 Nov 30 07:40 classify/
-rw-r--r-- 1 root root 5009 Nov 30 07:40 CONTRIBUTING.md
drwxr-xr-x 5 root root 4096 Nov 30 07:40 data/
-rw-r--r-- 1 root root 15364 Nov 30 07:40 detect.py
-rw-r--r-- 1 root root 3701 Nov 30 07:40 .dockerignore
-rw-r--r-- 1 root root 41275 Nov 30 07:40 export.py
drwxr-xr-x 8 root root 4096 Nov 30 07:40 .git/
-rw-r--r-- 1 root root 75 Nov 30 07:40 .gitattributes
drwxr-xr-x 4 root root 4096 Nov 30 07:40 .github/
-rwxr-xr-x 1 root root 3998 Nov 30 07:40 .gitignore*
-rw-r--r-- 1 root root 7769 Nov 30 07:40 hubconf.py
-rw-r--r-- 1 root root 34523 Nov 30 07:40 LICENSE
drwxr-xr-x 4 root root 4096 Nov 30 07:40 models/
-rw-r--r-- 1 root root 1815 Nov 30 07:40 .pre-commit-config.yaml
-rw-r--r-- 1 root root 42023 Nov 30 07:40 README.md
-rw-r--r-- 1 root root 41050 Nov 30 07:40 README.zh-CN.md
-rw-r--r-- 1 root root 1563 Nov 30 07:40 requirements.txt
drwxr-xr-x 2 root root 4096 Nov 30 07:40 segment/
-rw-r--r-- 1 root root 1723 Nov 30 07:40 setup.cfg
-rw-r--r-- 1 root root 33959 Nov 30 07:40 train.py
-rw-r--r-- 1 root root 41420 Nov 30 07:40 tutorial.ipynb
drwxr-xr-x 8 root root 4096 Nov 30 07:40 utils/
-rw-r--r-- 1 root root 20586 Nov 30 07:40 val.py
```

Actual Training in Colab (3): Yolo v5

❖ Download your dataset in Roboflow server.

- You need Your API_KEY, Workspace ID, Project ID.

```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="MMQccvY9HKjgRLQzIwLO")
project = rf.workspace("bg-kim-sookmyung-ac-kr").project("pokercard_2023")
dataset = project.version(1).download("yolov5")

!ls
```

```
Found existing installation: chardet 5.2.0
Uninstalling chardet-5.2.0:
  Successfully uninstalled chardet-5.2.0
Successfully installed chardet-4.0.0 cyclier-0.10.0 idna-2.10 opencv-python-headless-4.8.0.74 pyparsing-2.4.7 python-dotenv-1.0.0 python-magic-0.4.27
WARNING: The following packages were previously imported in this runtime:
[cyclier,pyparsing]
You must restart the runtime in order to use newly installed versions.

RESTART RUNTIME

loading Roboflow workspace...
loading Roboflow project...
Downloading Dataset Version Zip in PokerCard_2023-1 to yolov5pytorch:: 100%|██████████| 14295/14295 [00:00<00:00, 57002.08it/s]

Extracting Dataset Version Zip to PokerCard_2023-1 in yolov5pytorch:: 100%|██████████| 525/525 [00:00<00:00, 5541.03it/s]
benchmarks.py  data      LICENSE    README.zh-CN.md  train.py
CITATION.cff  detect.py  models     requirements.txt  tutorial.ipynb
classify      export.py  PokerCard_2023-1  segment          utils
CONTRIBUTING.md  hubconf.py  README.md   setup.cfg        val.py
```

Actual Training in Colab (4): Yolo v5

❖ Download the pre-trained weight from Guihub.

```
!wget https://github.com/ultralytics/yolov5/releases/download/v7.0/yolov5s.pt  
!ls
```

```
--2023-11-30 07:48:56-- https://github.com/ultralytics/yolov5/releases/download/v7.0/yolov5s.pt  
Resolving github.com (github.com)... 140.82.114.4  
Connecting to github.com (github.com)|140.82.114.4|:443... connected.  
HTTP request sent, awaiting response... 302 Found  
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/264818686/381bd8a8-8910-4e9e-b0dd-2752951ef78  
--2023-11-30 07:48:57-- https://objects.githubusercontent.com/github-production-release-asset-2e65be/264818686/381bd8a8-8910-4e9e-b  
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...  
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108.133|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 14808437 (14M) [application/octet-stream]  
Saving to: 'yolov5s.pt'  
  
yolov5s.pt      100%[=====] 14.12M  --.-KB/s   in 0.09s  
  
2023-11-30 07:48:57 (162 MB/s) - 'yolov5s.pt' saved [14808437/14808437]  
  
benchmarks.py  data      LICENSE    README.zh-CN.md  train.py      yolov5s.pt  
CITATION.cff  detect.py  models     requirements.txt  tutorial.ipynb  
classify      export.py  PokerCard_2023-1  segment          utils  
CONTRIBUTING.md  hubconf.py  README.md    setup.cfg        val.py
```

Actual Training in Colab (5): Yolo v5

❖ Download the pre-trained weight from Guihub.

```
!wget https://github.com/ultralytics/yolov5/releases/download/v7.0/yolov5s.pt  
!ls
```

```
--2023-11-30 07:48:56-- https://github.com/ultralytics/yolov5/releases/download/v7.0/yolov5s.pt  
Resolving github.com (github.com)... 140.82.114.4  
Connecting to github.com (github.com)|140.82.114.4|:443... connected.  
HTTP request sent, awaiting response... 302 Found  
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/264818686/381bd8a8-8910-4e9e-b0dd-2752951ef78  
--2023-11-30 07:48:57-- https://objects.githubusercontent.com/github-production-release-asset-2e65be/264818686/381bd8a8-8910-4e9e-b  
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...  
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108.133|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 14808437 (14M) [application/octet-stream]  
Saving to: 'yolov5s.pt'  
  
yolov5s.pt      100%[=====>]  14.12M  --.-KB/s   in 0.09s  
  
2023-11-30 07:48:57 (162 MB/s) - 'yolov5s.pt' saved [14808437/14808437]  
  
benchmarks.py  data      LICENSE  README.zh-CN.md  train.py      yolov5s.pt  
CITATION.cff  detect.py  models   requirements.txt  tutorial.ipynb  
classify      export.py  PokerCard_2023-1  segment         utils  
CONTRIBUTING.md  hubconf.py  README.md  setup.cfg       val.py
```

```
!ls -al
```

Actual Training in Colab (6): Yolo v5

❖ Train Yolo v5 model (training stage).

```
# 학습 단계
# --img      : 이미지 크기
# --batch   : 배치 크기
# --epochs  : 전체 학습 반복 횟수
# --data    : 데이터셋 경로 (data.yaml 파일은 로보플로에서 자동 생성됨)
# --weights : 이전에 학습된 weight 파일의 경로

#!python train.py --img 640 --batch 64 --epochs 300 --data PokerCard-1/data.yaml --
weights yolov5s.pt
!python train.py --img 640 --batch 64 --epochs 300 --data PokerCard_2023-1/data.yaml --
weights yolov5s.pt
```

```
...
22      [-1, 10] 1      0 models.common.Concat [200, 200, 3, 1]
23      -1 1 1182720 models.common.C3 [1]
24      [17, 20, 23] 1 29667 models.yolo.Detect [512, 512, 1, False]
Model summary: 214 layers, 7035811 parameters, 7035811 gradients, 16.0 GFLOPs

Transferred 343/349 items from yolov5s.pt
AMP: checks passed ✓
optimizer: SGD(lr=0.01) with parameter groups 57 weight(decay=0.0), 60 weight(decay=0.0005), 60 bias
augmentations: Blur(p=0.01, blur_limit=(3, 7)), MedianBlur(p=0.01, blur_limit=(3, 7)), ToGray(p=0.01), CLAHE(p=0.01, clip_limit=(1, 4.0), tile_grid_size=(8, 8))
train: Scanning /content/yolov5/PokerCard_2023-1/train/labels... 225 Images, 0 backgrounds, 0 corrupt: 100% 225/225 [00:00<00:00, 1670.48it/s]
train: New cache created: /content/yolov5/PokerCard_2023-1/train/labels.cache
val: Scanning /content/yolov5/PokerCard_2023-1/valid/labels... 21 Images, 0 backgrounds, 0 corrupt: 100% 21/21 [00:00<00:00, 366.59it/s]
val: New cache created: /content/yolov5/PokerCard_2023-1/valid/labels.cache

AutoAnchor: 3.70 anchors/target, 1.000 Best Possible Recall (BPR). Current anchors are a good fit to dataset ✓
Plotting labels to runs/train/exp/labels.jpg...
Image sizes 640 train, 640 val
Using 2 dataloader workers
Logging results to runs/train/exp
Starting training for 300 epochs...

Epoch  GPU_mem  box_loss  obj_loss  cls_loss  Instances  Size
0/299   13.16    0.1225   0.03187  0.05694   93         640: 100% 4/4 [00:09<00:00, 2.28s/it]
      Class  Images  Instances  P      R      mAP50   mAP50-95: 100% 1/1 [00:03<00:00, 3.71s/it]
      all    21      33        0.00146 0.241  0.00119 0.000206

Epoch  GPU_mem  box_loss  obj_loss  cls_loss  Instances  Size
1/299   13.16    0.119    0.03231  0.05746   95         640: 100% 4/4 [00:02<00:00, 1.54it/s]
      Class  Images  Instances  P      R      mAP50   mAP50-95: 100% 1/1 [00:00<00:00, 1.38it/s]
      all    21      33        0.00166 0.307  0.00145 0.000384

Epoch  GPU_mem  box_loss  obj_loss  cls_loss  Instances  Size
2/299   13.16    0.1036   0.03278  0.05541   104        640: 100% 4/4 [00:07<00:00, 1.84s/it]
      Class  Images  Instances  P      R      mAP50   mAP50-95: 100% 1/1 [00:00<00:00, 1.48it/s]
      all    21      33        0.00284 0.421  0.00565 0.001

Epoch  GPU_mem  box_loss  obj_loss  cls_loss  Instances  Size
0% 0/4 [00:00<?, ?it/s]
```

Your data folder

Actual Training in Colab (7): Yolo v5

❖ Checking the training status.

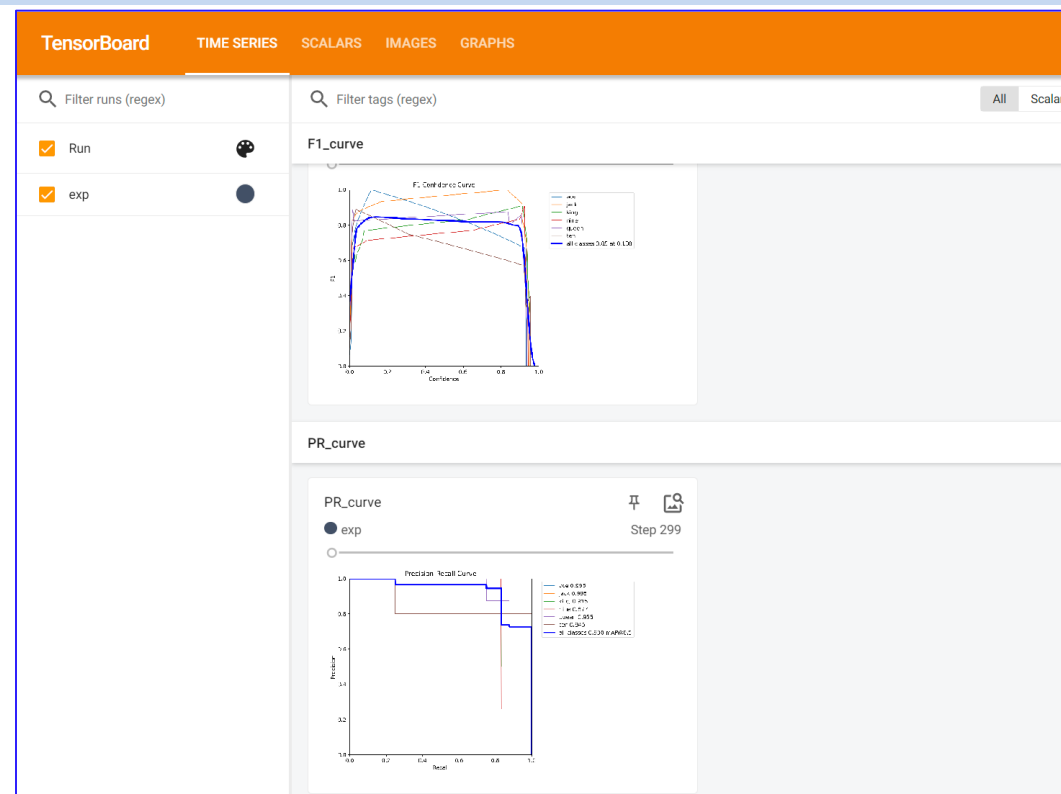
학습 추세 확인

추세 및 가중치 파일은 runs/train/ 경로 아래에 exp 라는 이름의 디렉토리에 생성됨

학습을 중단 후 다시 시작하면 exp2, exp3, exp4, ... 와 같이 숫자를 붙여 생성됨



```
%load_ext tensorboard
```

```
%tensorboard --logdir runs/train
```



❖ Evaluating the trained network.

```
# 평가 단계
#!python detect.py --weight runs/train/exp/weights/best.pt --conf
0.1 --source PokerCard-1/valid/images
!python detect.py --weight runs/train/exp/weights/best.pt --conf 0.1
--source PokerCard_2023-1/valid/images
```

```
 detect: weights=['runs/train/exp/weights/best.pt'], source=PokerCard-1/valid/images, data=data/coco128.yaml, imgsz=[640, 640], conf_thres=0.1, iou_thres=0.45
YOLOv5  v7.0-247-g3f02fde Python-3.10.12 torch-2.1.0+cu118 CUDA:0 (Tesla T4, 15102MiB)

Fusing layers...
Model summary: 157 layers, 7026307 parameters, 0 gradients, 15.8 GFLOPs
image 1/15 /content/yolov5/PokerCard-1/valid/images/14_jpg.rf.cc24109f8fc6b9ed09c61b3526a04d78.jpg: 640x640 1 nine, 11.6ms
image 2/15 /content/yolov5/PokerCard-1/valid/images/15_jpg.rf.77f89f8dab903e9468a74a0ae5768072.jpg: 640x640 1 nine, 11.6ms
image 3/15 /content/yolov5/PokerCard-1/valid/images/18_jpg.rf.df4563f18597a4515731367bfbdb08e41.jpg: 640x640 1 ace, 11.6ms
image 4/15 /content/yolov5/PokerCard-1/valid/images/246_jpg.rf.2978c534b557456fc23c8658d583d0e0.jpg: 640x640 1 ace, 11.6ms
image 5/15 /content/yolov5/PokerCard-1/valid/images/249_jpg.rf.2b887dbda28ac7e5cc4fd0b2a507fd63.jpg: 640x640 1 ace, 11.5ms
image 6/15 /content/yolov5/PokerCard-1/valid/images/2_jpg.rf.e1504b3030dec38ca389d2e6d74da527.jpg: 640x640 1 queen, 11.5ms
image 7/15 /content/yolov5/PokerCard-1/valid/images/35_jpg.rf.0cb327ebfd08127a93b24cb5b6956888.jpg: 640x640 1 ace, 1 ten, 11.5ms
image 8/15 /content/yolov5/PokerCard-1/valid/images/38_jpg.rf.3af00ca6fe19faf2d8e041f72759a5ab.jpg: 640x640 1 ace, 2 jacks, 1 nine, 2 queens, 1 ten, 11.5ms
image 9/15 /content/yolov5/PokerCard-1/valid/images/40_jpg.rf.e80820292b668aaa7f65d5d5c3de25ad.jpg: 640x640 2 nines, 2 queens, 11.5ms
image 10/15 /content/yolov5/PokerCard-1/valid/images/45_jpg.rf.4167b7229ce818cb437f779696c58f67.jpg: 640x640 2 queens, 1 ten, 11.5ms
image 11/15 /content/yolov5/PokerCard-1/valid/images/58_jpg.rf.b7ce2b54996e2344da7f22aef91ad2a0.jpg: 640x640 2 aces, 1 nine, 11.5ms
image 12/15 /content/yolov5/PokerCard-1/valid/images/5_jpg.rf.69f056c6aa72c0ca249d788757af9d58.jpg: 640x640 1 nine, 1 ten, 11.5ms
image 13/15 /content/yolov5/PokerCard-1/valid/images/61_jpg.rf.91d4e2d1921faf8dd72e559e58d484c1.jpg: 640x640 1 queen, 10.8ms
image 14/15 /content/yolov5/PokerCard-1/valid/images/6_jpg.rf.0b24a3a53c2e4d5efdde266045de8f51.jpg: 640x640 1 ten, 10.8ms
image 15/15 /content/yolov5/PokerCard-1/valid/images/7_jpg.rf.1a2eac32811901e5e067b3d9d7b0bd01.jpg: 640x640 1 ten, 10.9ms
Speed: 0.5ms pre-process, 11.4ms inference, 1.1ms NMS per image at shape (1, 3, 640, 640)
Results saved to runs/detect/exp
```

❖ Visualize the inference results.

```
# 평가 결과를 시각화
import glob
from IPython.display import Image, display

for i, imageName in enumerate(glob.glob('runs/detect/exp/*.jpg')): #assuming
    JPG
    if i > 10:
        break
    display(Image(filename=imageName))
    print("\n")
```



Actual Training in Colab (10): Yolo v5

❖ Convert the format into C++ libtorch.

```
# c++ libtorch 에서 사용하기 위한 포맷으로 변환
!python export.py --weights runs/train/exp/weights/best.pt --include torchscript
!ls runs/train/exp/weights/
```

```
export: data=data/coco128.yaml, weights=['runs/train/exp/weights/best.pt'], imgsz=[640, 640], batch_size=1, device=cpu, half=False, inplace=False, YOLOv5 v7.0-247-g3f02fde Python=3.10.12 torch=2.1.0+cu118 CPU

Fusing layers...
Model summary: 157 layers, 7026307 parameters, 0 gradients, 15.8 GFLOPs

PyTorch: starting from runs/train/exp/weights/best.pt with output shape (1, 25200, 11) (13.8 MB)


TorchScript: starting export with torch 2.1.0+cu118...
TorchScript: export success 3.7s, saved as runs/train/exp/weights/best.torchscript (27.3 MB)

Export complete (5.4s)
Results saved to /content/yolov5/runs/train/exp/weights
Detect:      python detect.py --weights runs/train/exp/weights/best.torchscript
Validate:    python val.py --weights runs/train/exp/weights/best.torchscript
PyTorch Hub: model = torch.hub.load('ultralytics/yolov5', 'custom', 'runs/train/exp/weights/best.torchscript')
Visualize:   https://netron.app
best.pt     best.torchscript  last.pt
```

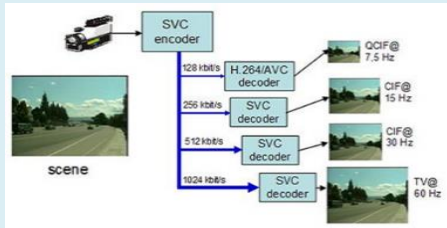
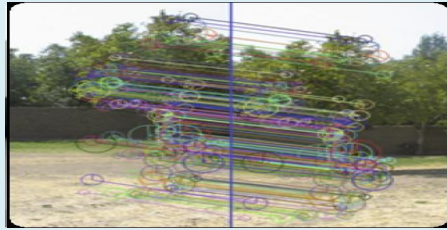
Actual Training in Colab (11): Yolo v5

- ❖ Upload the transformed format (weight) to your Google drive.

```
# 학습된 weight 파일을 구글드라이브에 저장
from google.colab import drive
drive.mount('/content/gdrive')
%cp /content/yolov5/runs/train/exp/weights/best.torchscript
/content/gdrive/My\ Drive/yolov5_best.torchscript
```

 Mounted at /content/gdrive

- *Then check your google drive. You can see the trained weight for Yolo v5.*



Contents

- My Github
- Preparation of My Dataset
- Actual Training Object Detection in Colab
- **To my local Yolo v5 model to detect objects**

Change your inference model into your local webcam...!!!! (1)

❖ OpenCV Environment Variables (We already set this...!!)

- Add (OPENCV_DIR) in Path environment
- Add %OPENCV_DIR%\#bin in Path variable.

The image illustrates the process of configuring environment variables for OpenCV. It consists of several overlapping windows:

- File Explorer:** Shows the directory path `C:\Users\yvwlee\Downloads\opencv\opencv\build\install\x64\vc17` containing `bin` and `lib` folders.
- Environment Variables (User variables for yvwlee):** Shows existing variables like `ChocolateyLastPathUpdate`, `GOPATH`, `NVM_HOME`, `NVM_SYMLINK`, `OneDrive`, and `OneDriveCommercial`.
- New System Variable:** A dialog box where a new variable is being created. The **Variable name** is `OPENCV_DIR` and the **Variable value** is `C:\Users\yvwlee\Downloads\opencv\build\install\x64\vc17`. A red arrow points to the `Browse File...` button.
- Environment Variables (System variables):** Shows the `Path` variable highlighted in red, indicating it is being edited.
- Edit environment variable:** A dialog box for editing the `Path` variable. It shows a list of paths, with `%OPENCV_DIR%\bin` added at the bottom. A red arrow points to the `New...` button.

Change your inference model into your local webcam...!!!! (2)

❖ Install LibTorch.

- LibTorch: Pytorch C++ version library
- Download it from <https://pytorch.org/get-started/locally/> with following options.
- In this lecture, I explain **Debug version** as reference.

You need to download ver 1.13.0 which has been checked..!

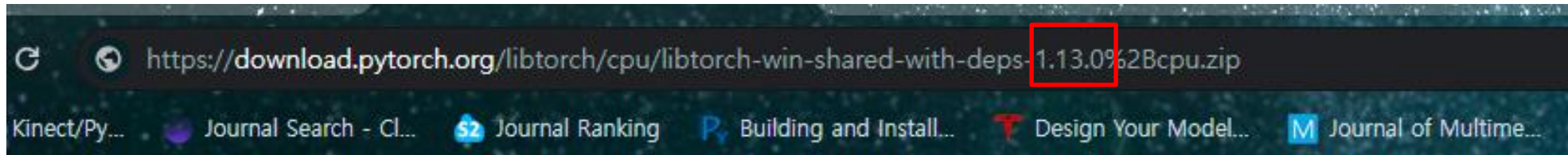
PyTorch Build	Stable (1.13.0)		Preview (Nightly)	
Your OS	Linux	Mac	Windows	
Package	Conda	Pip	LibTorch	Source
Language	Python		C++ / Java	
Compute Platform	CUDA 11.6	CUDA 11.7	ROCm 5.2	CPU
Run this Command:	<u>Download here (Release version):</u> https://download.pytorch.org/libtorch/cpu/libtorch-win-shared-with-deps-1.13.0%2Bcpu.zip <u>Download here (Debug version):</u> https://download.pytorch.org/libtorch/cpu/libtorch-win-shared-with-deps-debug-1.13.0%2Bcpu.zip			

Change your inference model into your local webcam...!!!! (2-1)

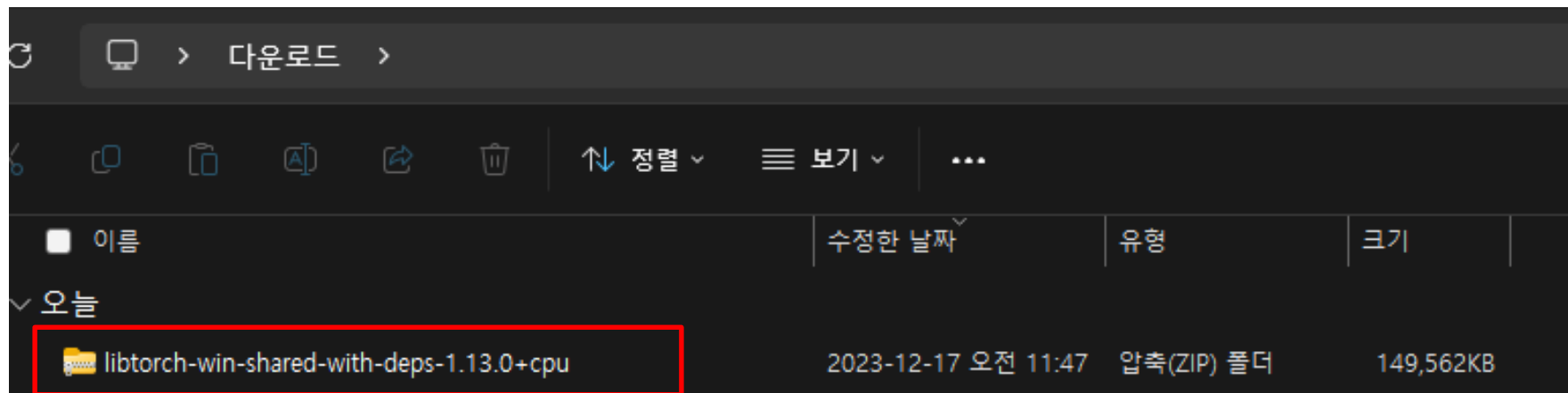
- Copy downlink link to URL of your browser as:

<https://download.pytorch.org/libtorch/cpu/libtorch-win-shared-with-deps-2.1.2%2Bcpu.zip>

- Check your version and mode (release or debug mode as your VS project)
 - In url, **you just need to change the version 2.1.2 into 1.13.0**. That is,



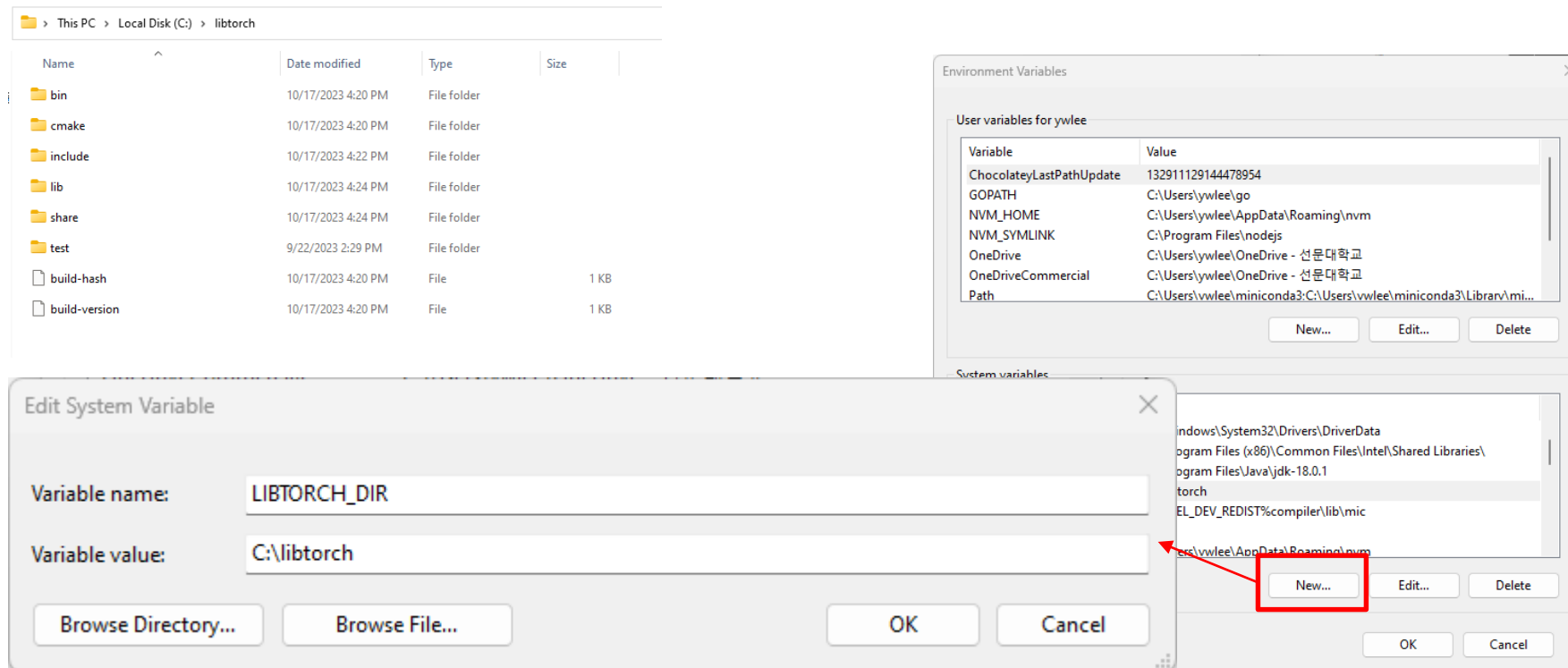
- To download, you can just "enter".



**본인의 프로젝트가 debug 모드인 경우 debug 모드 다운로드 링크를 복사해서 사용해야 함!*

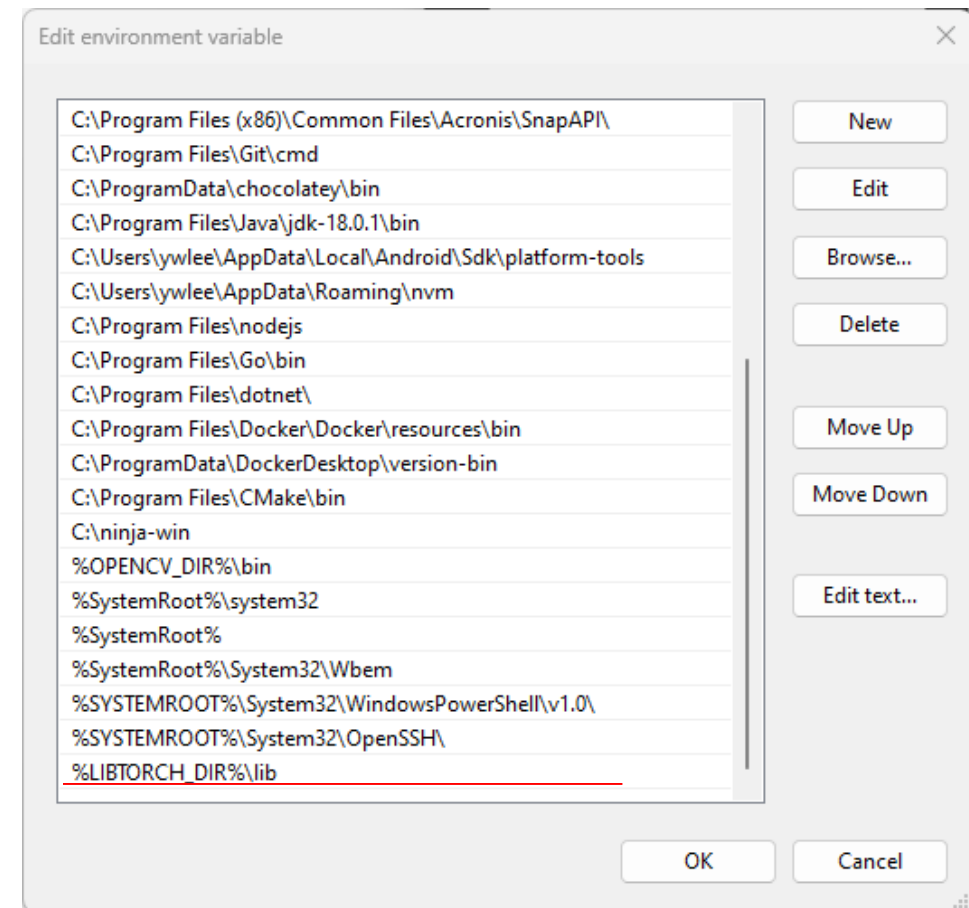
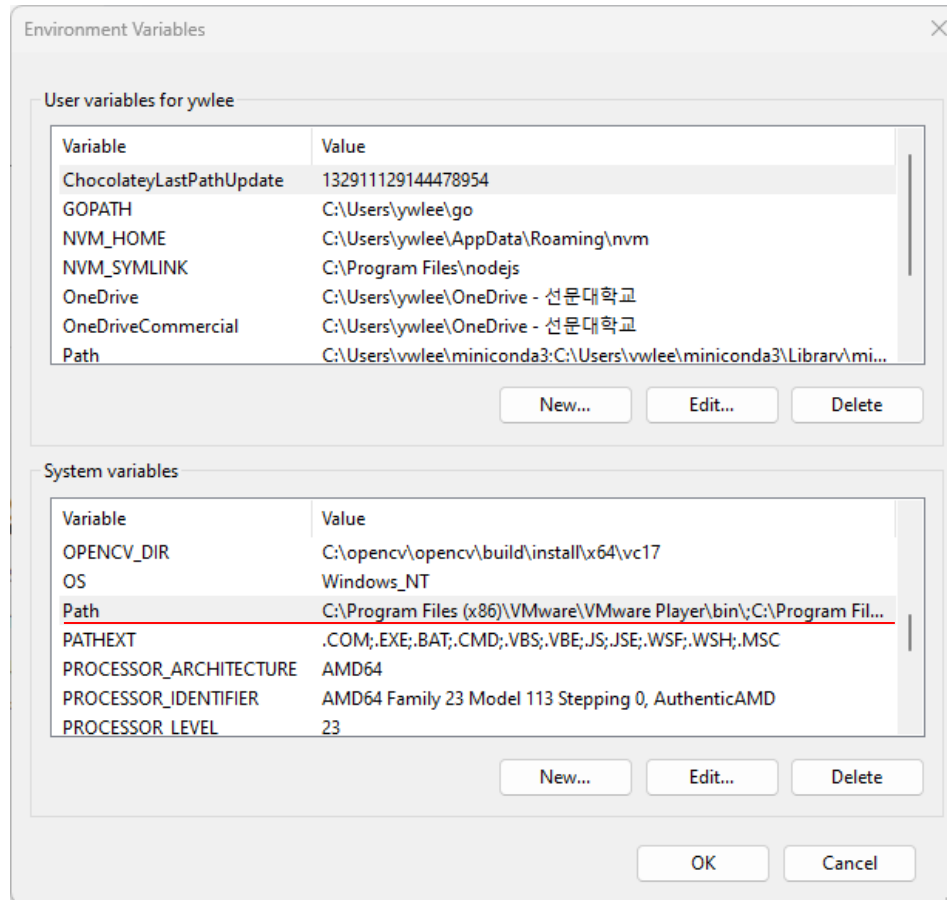
Change your inference model into your local webcam...!!!! (3)

- Decompress the file and locate the folder at root C:/ (C:/libtorch).
 - You can see the following figure.
- Then add "LIBTORCH_DIR" in the environment (system) variable.



Change your inference model into your local webcam...!!!! (4)

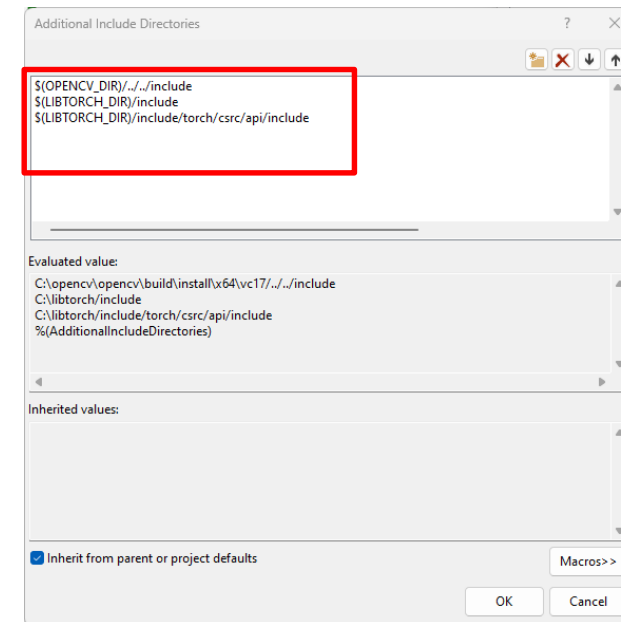
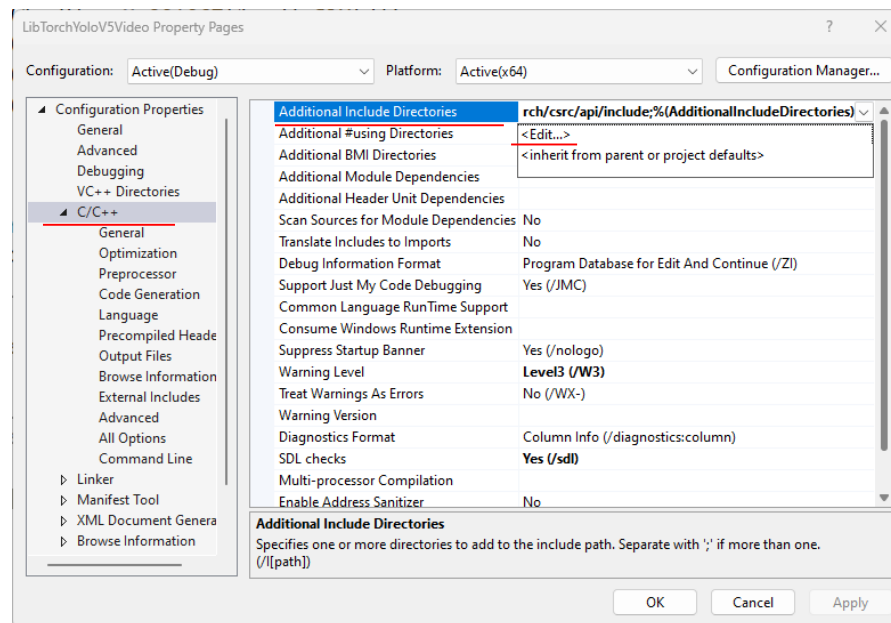
- Add “%LIBTORCH_DIR%\lib” in system variables: path.



Change your inference model into your local webcam...!!!! (5)

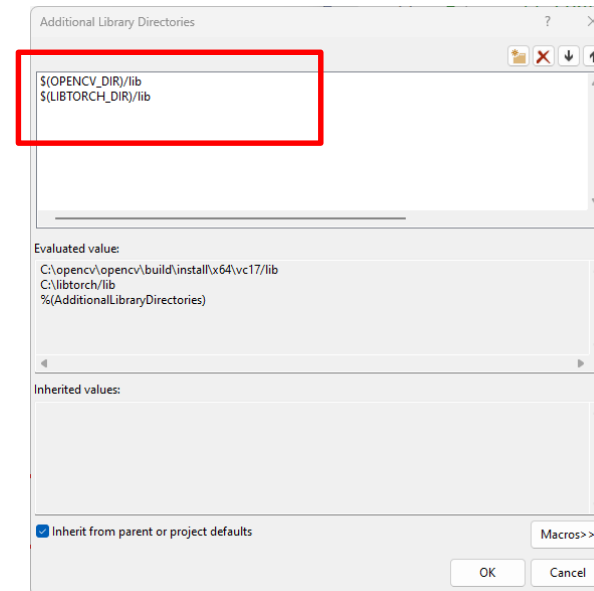
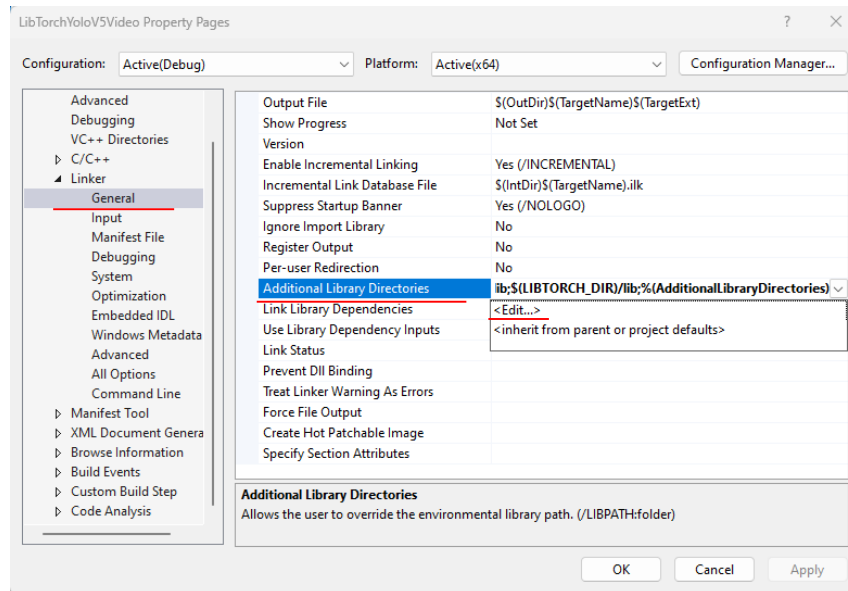
❖ LibTorch Configuration in Visual Studio 2022.

- Visual Studio → New project → add (.cpp) file.
- Change “Properties” of my project (solution)
 - Set the OpenCV & LibTorch path.
- Item: **C/C++** → **Additional Include Directories**
 - `$(OPENCV_DIR)/.././include` ← **Already set...!!!**
 - `$(LIBTORCH_DIR)/include`
 - `$(LIBTORCH_DIR)/include/torch/csrc/api/include`



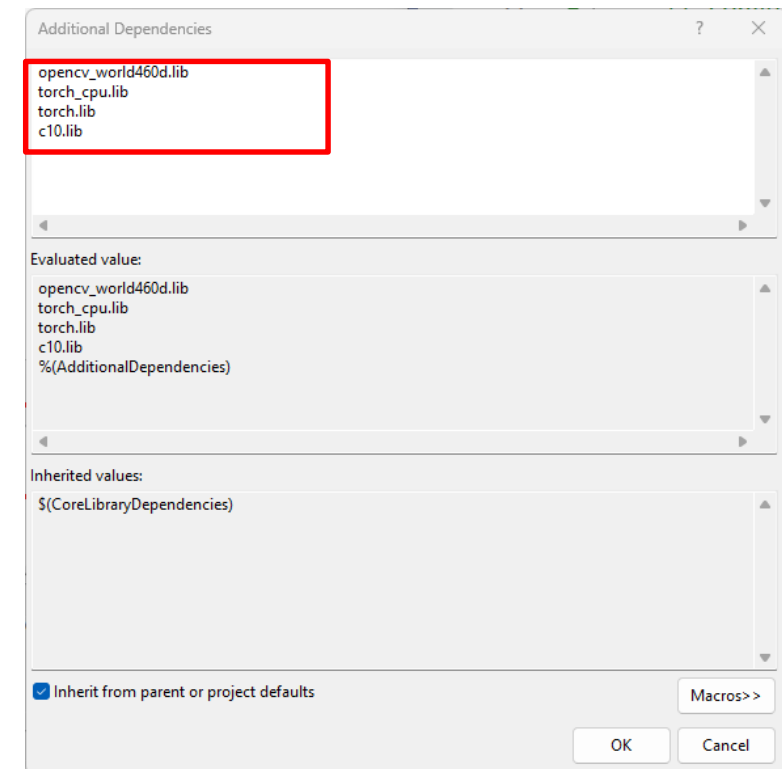
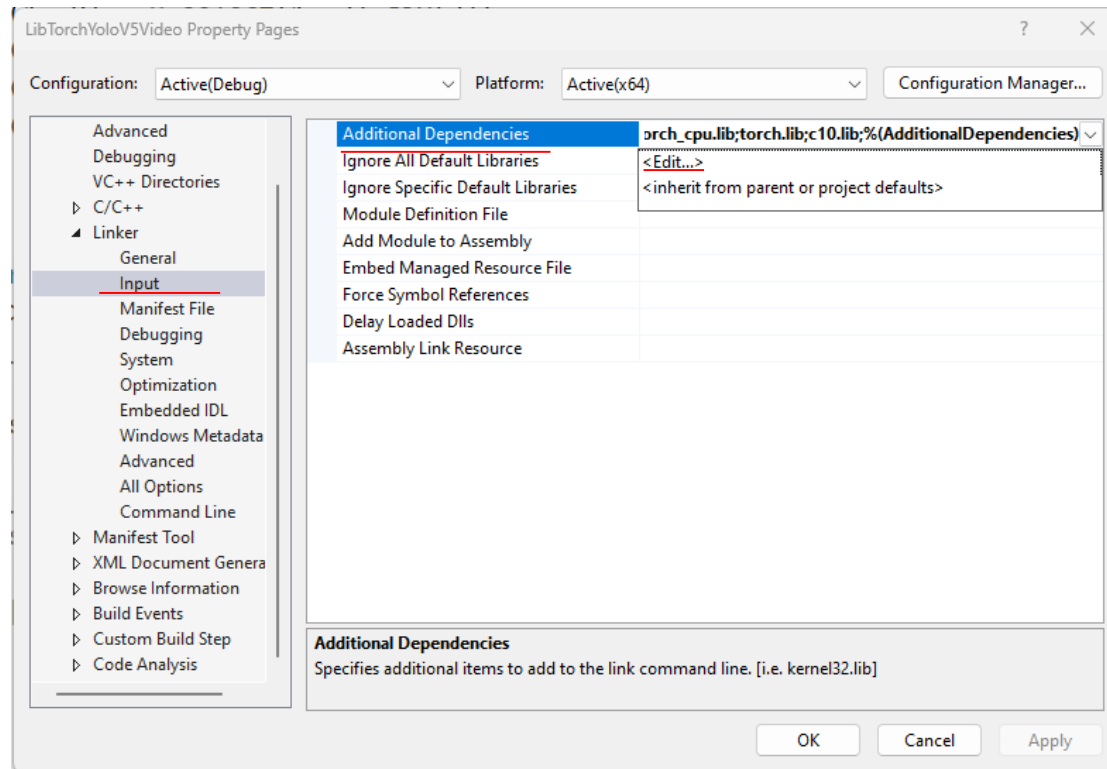
Change your inference model into your local webcam...!!!! (6)

- Linker → General → Additional Library Directories
 - \$(OPENCV_DIR)/lib ← **Already set...!!!**
 - \$(LIBTORCH_DIR)/lib



Change your inference model into your local webcam...!!!! (7)

- Linker → Input → Additional Dependencies
 - opencv_worldXXXd.lib (XXX is your version number) ← **Already set...!!!**
 - torch_cpu.lib
 - torch.lib
 - c10.lib



Change your inference model into your local webcam...!!!! (8)

❖ LibTorch Test

▪ Test Code sample

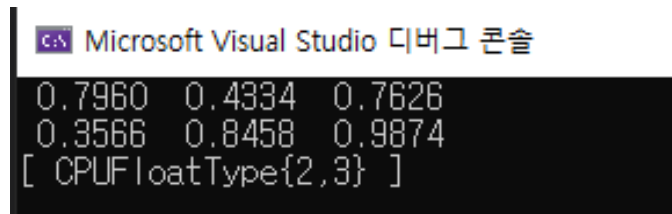
An example for displaying some random tensors in the screen...!!!!

```
#include <torch/script.h>
#include <iostream>

using namespace std;

int main()
{
    at::Tensor tensor = torch::rand({ 2,3 });
    cout << tensor << endl;

    return 0;
}
```



```
Microsoft Visual Studio 디버그 콘솔
0.7960 0.4934 0.7626
0.3566 0.8458 0.9874
[ CPUFloatType{2,3} ]
```

*If you got the result, your torch lib.
works well..!!*

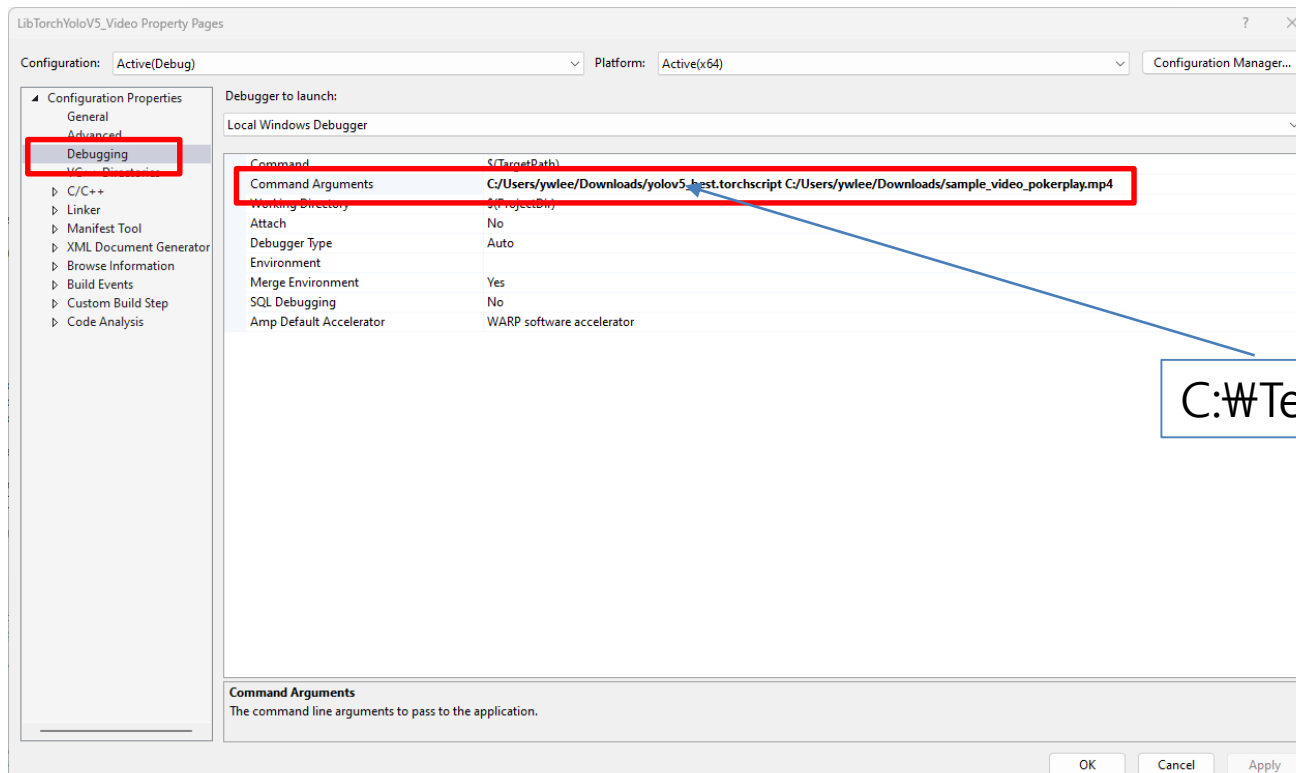
Yolo v5 Application Implementation (1)

❖ 실습에 필요한 파일

- Colab 을 통해 학습된 weight 파일 (yolov5_best.torchscript)
- 테스트용 포커이미지

❖ Visual Studio 프로젝트 속성에서 매개변수 전달 방법

- Configuration Properties → Debugging → Command Arguments
- 매개변수로 전달할 경로를 입력, 2개 이상일 경우 공백(스페이스바) 로 구분



C:\Temp\yolov5_best_2023.torchscript

Yolo v5 Application Implementation (1)

❖ Make your C++ Code as:

- Define header files.

```
#include <opencv2/opencv.hpp>  
#include <torch/script.h>  
#include <iostream>  
#include <vector>
```

```
using namespace std;  
using namespace cv;
```

Yolo v5 Application Implementation (2)

- Structure for object recognition results (객체인식 결과 정보를 담는 구조체 정의)

```
enum Det {  
    tl_x = 0,  
    tl_y = 1,  
    br_x = 2,  
    br_y = 3,  
    score = 4,  
    class_idx = 5  
};
```

```
struct Detection {  
    cv::Rect bbox;  
    float score;  
    int class_idx;  
};
```

Yolo v5 Application Implementation (3)

- Function for changing the coordinate to OpenCV format.
(YOLO 형식의 좌표를 OpenCV 형식의 좌표로 변환하는 함수)

```
torch::Tensor xywh2xyxy(const torch::Tensor& x) {  
    auto y = torch::zeros_like(x);  
    y.select(1, Det::tl_x) = x.select(1, 0) - x.select(1, 2).div(2);  
    y.select(1, Det::tl_y) = x.select(1, 1) - x.select(1, 3).div(2);  
    y.select(1, Det::br_x) = x.select(1, 0) + x.select(1, 2).div(2);  
    y.select(1, Det::br_y) = x.select(1, 1) + x.select(1, 3).div(2);  
    return y;  
}
```


Yolo v5 Application Implementation (4)

- Function for changing the detection results to Detection structure.
(텐서 자료형의 객체인식 결과를 Detection 구조체로 변환하는 함수)

```
void Tensor2Detection(const at::TensorAccessor<float, 2>& offset_boxes,
                    const at::TensorAccessor<float, 2>& det,
                    std::vector<cv::Rect>& offset_box_vec,
                    std::vector<float>& score_vec)
{
    for (int i = 0; i < offset_boxes.size(0); i++) {
        offset_box_vec.emplace_back(
            cv::Rect(cv::Point(offset_boxes[i][Det::tl_x],
                               offset_boxes[i][Det::tl_y]),
                    cv::Point(offset_boxes[i][Det::br_x],
                               offset_boxes[i][Det::br_y])));
        score_vec.emplace_back(det[i][Det::score]);
    }
}
```

Yolo v5 Application Implementation (5)

- Function for changing the object detection results to the original image coordinate.
(객체인식 결과의 좌표를 원본 이미지의 좌표로 환산하는 함수)

```
void ScaleCoordinates(std::vector<Detection>& data, float pad_w, float pad_h,
                    float scale, const cv::Size& img_shape)
{
    auto clip = [](float n, float lower, float upper)
    {
        return std::max(lower, std::min(n, upper));
    };
    std::vector<Detection> detections;
    for (auto& i : data) {
        float x1 = (i.bbox.tl().x - pad_w) / scale; // x padding
        float y1 = (i.bbox.tl().y - pad_h) / scale; // y padding
        float x2 = (i.bbox.br().x - pad_w) / scale; // x padding
        float y2 = (i.bbox.br().y - pad_h) / scale; // y padding

        x1 = clip(x1, 0, img_shape.width);
        y1 = clip(y1, 0, img_shape.height);
        x2 = clip(x2, 0, img_shape.width);
        y2 = clip(y2, 0, img_shape.height);

        i.bbox = cv::Rect(cv::Point(x1, y1), cv::Point(x2, y2));
    }
}
```

Yolo v5 Application Implementation (6)

- Function that converts input images to the fixed input size of the YOLO model and calculates the difference.

(입력 이미지를 YOLO 모델의 고정된 입력 크기로 변환 후 차이 값을 계산하는 함수)

```
std::vector<float> LetterboxImage(const cv::Mat& src,
                                cv::Mat& dst, const cv::Size& out_size)
{
    auto in_h = static_cast<float>(src.rows);
    auto in_w = static_cast<float>(src.cols);
    float out_h = out_size.height;
    float out_w = out_size.width;

    float scale = std::min(out_w / in_w, out_h / in_h);

    int mid_h = static_cast<int>(in_h * scale);
    int mid_w = static_cast<int>(in_w * scale);

    int top = (static_cast<int>(out_h) - mid_h) / 2;
    int down = (static_cast<int>(out_h) - mid_h + 1) / 2;
    int left = (static_cast<int>(out_w) - mid_w) / 2;
    int right = (static_cast<int>(out_w) - mid_w + 1) / 2;

    cv::copyMakeBorder(dst, dst, top, down, left, right,
                       cv::BORDER_CONSTANT, cv::Scalar(114, 114, 114));

    std::vector<float> pad_info{ static_cast<float>(left),
                                static_cast<float>(top), scale };
    return pad_info;
}
```

Yolo v5 Application Implementation (7)

- Write main () part...!!! (메인함수 작성)

```
int main(int argc, char** argv) {  
    if (argc < 3) {  
        cout << "Wrong path of weight and image" << endl;  
        return -1;  
    }
```

```
float conf_thres = 0.25; // 검출 정확도 기준  
float iou_thres = 0.45; // BBox 정확도 기준
```

```
// 클래스 이름
```

```
std::vector<std::string> class_names = {  
    "ace",  
    "jack",  
    "king",  
    "nine",  
    "queen",  
    "ten"  
};
```

Yolo v5 Application Implementation (8)

- (continue)

```
// 파이토치 모델 객체
torch::jit::script::Module model;
try {
    // 첫번째 인자 (argv[1]) 는 weight 파일 경로
    model = torch::jit::load(argv[1]);
    cout << "YOLO model loaded" << endl;
}
catch (const c10::Error& e) {
    cerr << "cannot load the weight file" << endl;
    cerr << e.backtrace() << endl;
    return -1;
}
// 두번째 인자 (argv[2]) 는 이미지 파일 경로
cv::Mat img = imread(argv[2], IMREAD_COLOR);
cv::Mat img_input = img.clone(); // 복사

// 이미지 크기를 640 x 640 크기로 일괄 변경
std::vector<float> pad_info = LetterboxImage(
    img_input, img_input, cv::Size(640, 640));
const float pad_w = pad_info[0];
const float pad_h = pad_info[1];
const float scale = pad_info[2];
```

Yolo v5 Application Implementation (9)

- (continue)

```
cv::resize(img_input, img_input,
           cv::Size(640, 640)); // 크기 변경
cv::cvtColor(img_input, img_input,
            cv::COLOR_BGR2RGB); // BGR -> RGB
img_input.convertTo(img_input, CV_32FC3,
                   1.0f / 255.0f); // normalization 1/255

// 텐서 객체 생성
auto tensor_img = torch::from_blob(img_input.data,
                                   { 1, img_input.rows, img_input.cols,
                                   img_input.channels() }, torch::kFloat);
// BHWC -> BCHW (Batch, Channel, Height, Width)
tensor_img = tensor_img.permute({ 0, 3, 1, 2 }).contiguous();
std::vector<torch::jit::IValue> inputs;
inputs.emplace_back(tensor_img);
```

Yolo v5 Application Implementation (10)

- (continue)

```
// 추론
torch::jit::IValue output = model.forward(inputs);
auto detections = output.toTuple()->elements()[0].toTensor();
constexpr int item_attr_size = 5;
int batch_size = detections.size(0); // 배치 크기
auto num_classes = detections.size(2)
    - item_attr_size; // 클래스 수

// confidence > threshold 조건에 해당하는 후보만 남김
auto conf_mask = detections.select(2, 4).ge(conf_thres).unsqueeze(2);
auto det = torch::masked_select(detections[0], conf_mask[0]).view(
    { -1, num_classes + item_attr_size });

// det 크기가 0 이면 검출된 결과가 존재하는 않는 것
if (0 == det.size(0)) {
    return -1;
}
```

Yolo v5 Application Implementation (11)

- (continue)

```
std::vector<std::vector<Detection>> result;
result.reserve(batch_size);

// 스코어 계산 = obj_conf * cls_conf, similar to x[:, 5:] *= x[:, 4:5]
det.slice(1, item_attr_size, item_attr_size + num_classes)
    *= det.select(1, 4).unsqueeze(1);

// 좌표 계산 (center x, center y, width, height) to (x1, y1, x2, y2)
torch::Tensor box = xywh2xyxy(det.slice(1, 0, 4));

// 클래스 스코어가 가장 높은 것만 남김
std::tuple<torch::Tensor, torch::Tensor> max_classes =
    torch::max(det.slice(1, item_attr_size,
        item_attr_size + num_classes), 1);
```


Yolo v5 Application Implementation (12)

- (continue)

```
// class score
auto max_conf_score = std::get<0>(max_classes);
// index
auto max_conf_index = std::get<1>(max_classes);

max_conf_score = max_conf_score.to(torch::kFloat).unsqueeze(1);
max_conf_index = max_conf_index.to(torch::kFloat).unsqueeze(1);

// shape: n * 6, top-left x/y (0,1),
// bottom-right x/y (2,3), score(4),
// class index(5)
det = torch::cat({ box.slice(1, 0, 4),
                  max_conf_score, max_conf_index }, 1);

// NMS 계산을 하기 위해 검출된 박스 결과를 배치 단위화
constexpr int max_wh = 4096;
auto c = det.slice(1, item_attr_size,
                  item_attr_size + 1) * max_wh;
auto offset_box = det.slice(1, 0, 4) + c;

std::vector<cv::Rect> offset_box_vec;
std::vector<float> score_vec;
```

- (continue)

```
// copy data back to cpu
auto offset_boxes_cpu = offset_box.cpu();
auto det_cpu = det.cpu();
const auto& det_cpu_array = det_cpu.accessor<float, 2>();

// 텐서 객체를 Detection 객체로 변경
Tensor2Detection(offset_boxes_cpu.accessor<float, 2>(),
                 det_cpu_array, offset_box_vec, score_vec);

// NMS: Non-Maximum Suppression 실행
std::vector<int> nms_indices;
cv::dnn::NMSSBoxes(offset_box_vec, score_vec,
                  conf_thres, iou_thres, nms_indices);
```

- (continue)

```
// NMS 계산 후의 박스들의 위치 좌표 저장
std::vector<Detection> det_vec;
for (int index : nms_indices) {
    Detection t;
    const auto& b = det_cpu_array[index];
    t.bbox = cv::Rect(cv::Point(b[Det::tl_x], b[Det::tl_y]),
                    cv::Point(b[Det::br_x], b[Det::br_y]));
    t.score = det_cpu_array[index][Det::score];
    t.class_idx = det_cpu_array[index][Det::class_idx];
    det_vec.emplace_back(t);
}

// 원본 이미지 크기에 맞추어 좌표를 재계산
ScaleCoordinates(det_vec, pad_w, pad_h, scale, img.size());
result.emplace_back(det_vec);
```

Yolo v5 Application Implementation (15)

- (continue)

```
// 이미지에 검출 결과를 표시
for (const auto& detection : result[0]) {
    const auto& box = detection.bbox;
    float score = detection.score;
    int class_idx = detection.class_idx;

    cv::rectangle(img, box, cv::Scalar(0, 0, 255), 2);
    std::stringstream ss;
    ss << std::fixed << std::setprecision(2) << score;
    std::string s = class_names[class_idx] + " " + ss.str();

    auto font_face = cv::FONT_HERSHEY_DUPLEX;
    auto font_scale = 1.0;
    int thickness = 1;
    int baseline = 0;
    auto s_size = cv::getTextSize(s, font_face,
        font_scale, thickness, &baseline);
    cv::rectangle(img,
        cv::Point(box.tl().x, box.tl().y - s_size.height - 5),
        cv::Point(box.tl().x + s_size.width, box.tl().y),
        cv::Scalar(0, 0, 255), -1);
    cv::putText(img, s, cv::Point(box.tl().x, box.tl().y - 5),
        font_face, font_scale,
        cv::Scalar(255, 255, 255), thickness);
}
```

- (continue)

```
// 화면 출력
cv::namedWindow("Result", WINDOW_AUTOSIZE);
cv::imshow("Result", img);
cv::waitKey(0);

return 0;
} //← 메인 함수의 끝
```

**실행 방법: CVApp.exe C:\Temp\yolov5_best_2023.torchscript IMG_2384.JPG
(CV.exe Yolov5 weight 경로 테스트이미지)**

Yolo v5 Application Implementation (17)

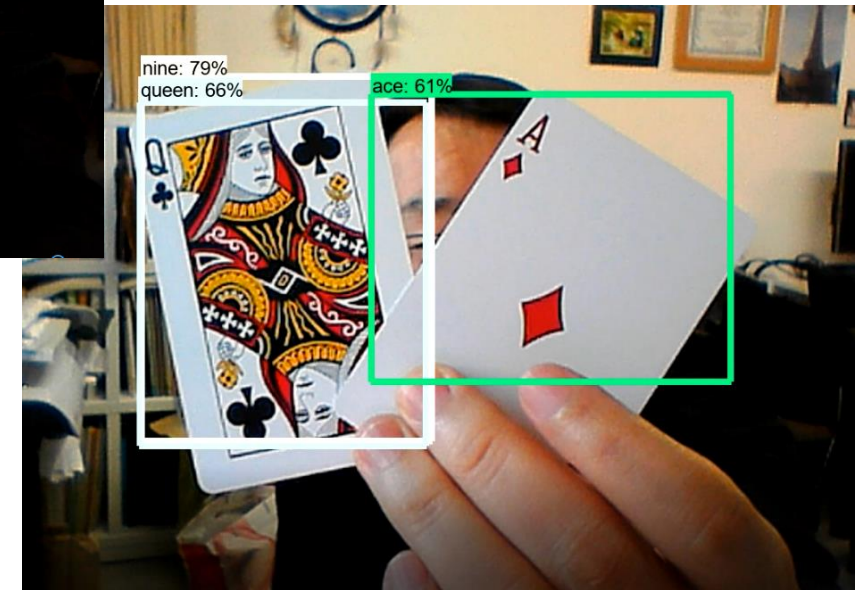
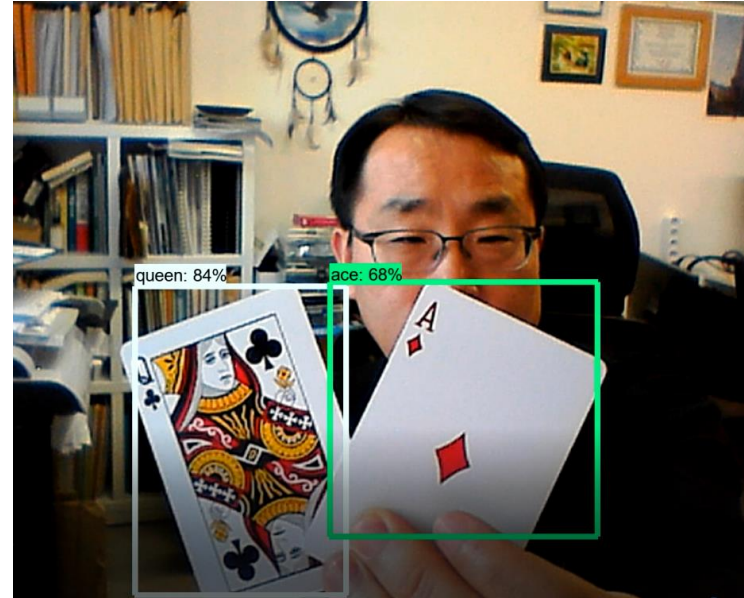
- ❖ Run your Yolo v5 model to infer your input image as:

```
C:\Users\SM-PC\Lectures\2023\ComputerVision\CVApp_YoLov5ObjectDetection\x64\Release>CVApp.exe C:\Temp\yoLov5_best_2023.torchscript IMG_2391.JPG  
YOLO model loaded
```

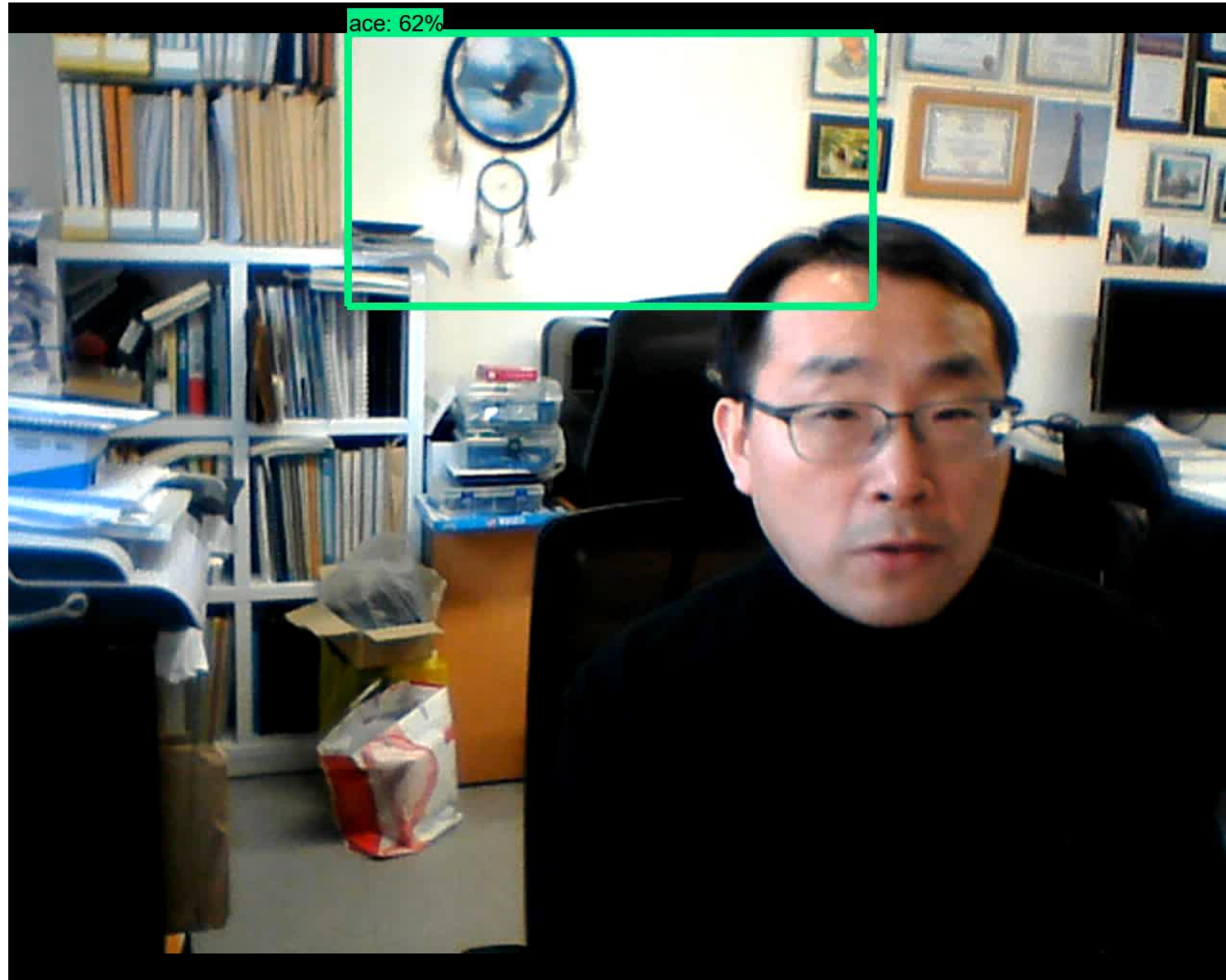


Yolo v5 Application Implementation (18): With your local webcam...!!!!

❖ Change the input as your webcam. Then run your inference as:



Yolo v5 Application Implementation (19): With your local webcam...!!!!



Thank you for your attention!!!
QnA

<http://ivpl.sookmyung.ac.kr>