code review
[Preprocessing]

Pre-Processing protocol

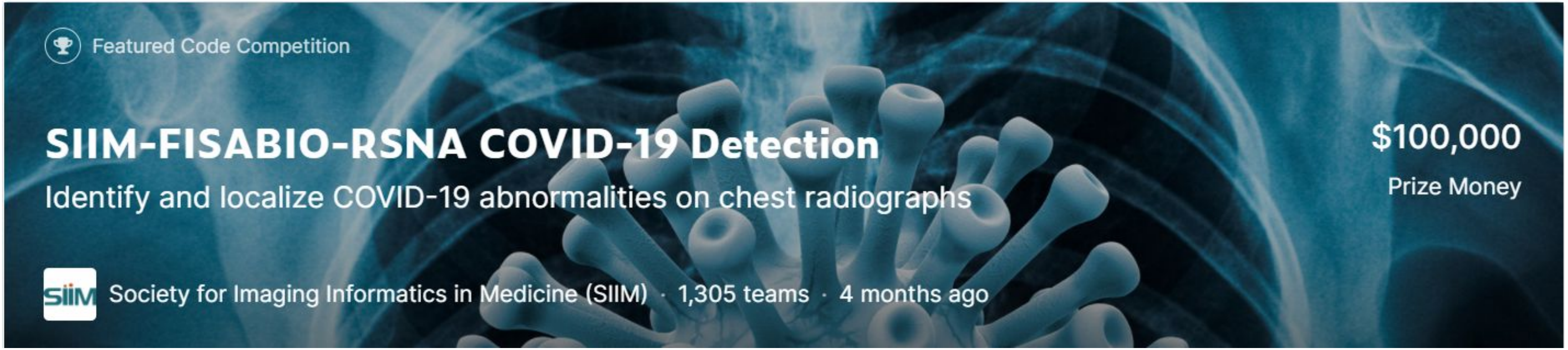
<2.DataPrepare> - Data Preprocessing

- **make_json_real.py** (dicom metadata와 data제공 csv를 각각 json으로 변환하여 하나의 json으로 병합하여 저장)

- **make_txt_c18.py** (json to txt convert for detection-YOLOv5)

- **crop_make_folder.py** (ROI crop & foldering image by class for classification-EfficientNetB4)

Dataset: SIIM COVID-19 Detection



Featured Code Competition

SIIM-FISABIO-RSNA COVID-19 Detection

Identify and localize COVID-19 abnormalities on chest radiographs

$100,000

Prize Money

Society for Imaging Informatics in Medicine (SIIM) · 1,305 teams · 4 months ago

- 약 7000(train-6000 / test-1000)장의 dicom 폐 이미지

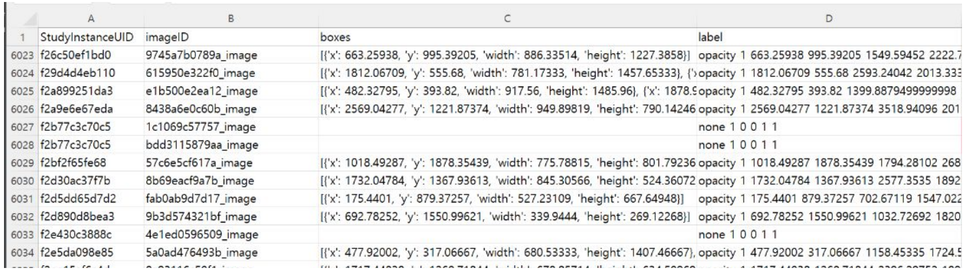
- class 및 bounding box 정보를 담은 csv

- 4개의 class
(Negative for Pneumonia / Typical Appearance / Indeterminate Appearance / Atypical Appearance)

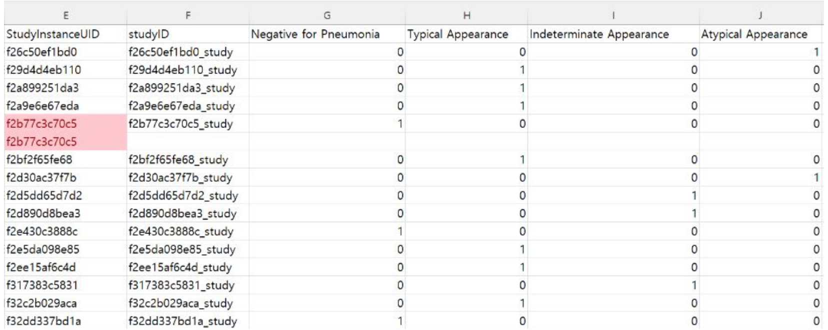
Pre-Processing; make_json_real.py

- dicom metadata와 data제공 csv를 각각 json으로 변환하여 하나의 json으로 병합하여 저장

- dataset 제공 csv

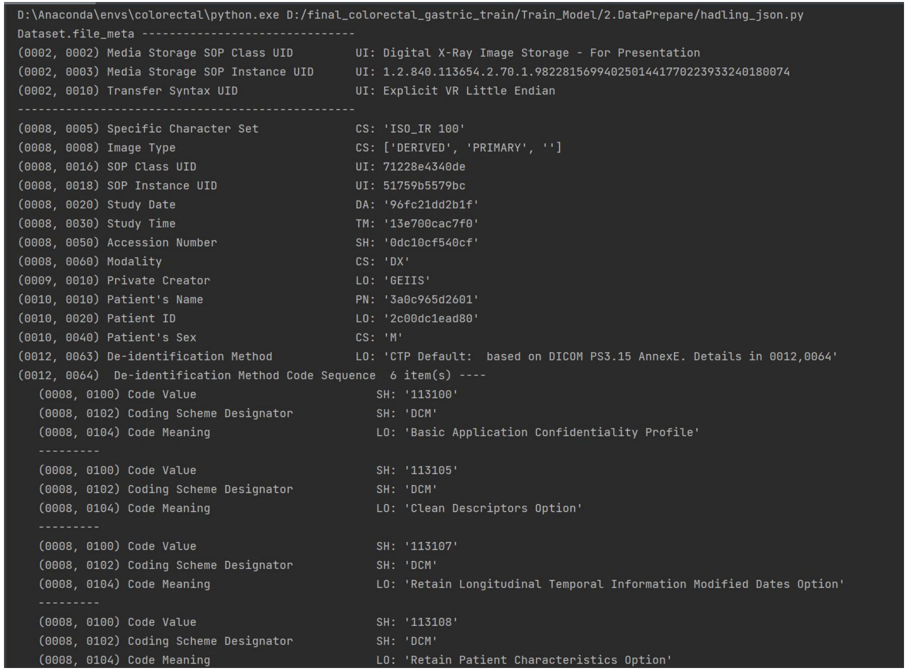


	A	B	C	D
1	StudyInstanceUID	imageID	boxes	label
6023	f26c50ef1bd0	9745a7b0789a_image	[{'x': 663.25938, 'y': 995.39205, 'width': 886.33514, 'height': 1227.3858}]	opacity 1 663.25938 995.39205 1549.59452 2222.7
6024	f29d4d4eb110	615950e322f0_image	[{'x': 1812.06709, 'y': 555.68, 'width': 781.17333, 'height': 1457.65333}, {'x	opacity 1 1812.06709 555.68 2593.24042 2013.333
6025	f2a899251da3	e1b500e2ea12_image	[{'x': 482.32795, 'y': 393.82, 'width': 917.56, 'height': 1485.96}, {'x': 1878.9	opacity 1 482.32795 393.82 1399.8879499999998
6026	f2a9e6e67eda	8438a6e0c60b_image	[{'x': 2569.04277, 'y': 1221.87374, 'width': 949.89819, 'height': 790.14246	opacity 1 2569.04277 1221.87374 3518.94096 201
6027	f2b77c3c70c5	1c1069c57757_image		none 1 0 0 1 1
6028	f2b77c3c70c5	bdd3115879aa_image		none 1 0 0 1 1
6029	f2bf2f65fe68	57c6e5cf617a_image	[{'x': 1018.49287, 'y': 1878.35439, 'width': 775.78815, 'height': 801.79236	opacity 1 1018.49287 1878.35439 1794.28102 268
6030	f2d30ac37f7b	8b69eacf9a7b_image	[{'x': 1732.04784, 'y': 1367.93613, 'width': 845.30566, 'height': 524.36072	opacity 1 1732.04784 1367.93613 2577.3535 1892
6031	f2d5dd65d7d2	fab0ab9d7d17_image	[{'x': 175.4401, 'y': 879.37257, 'width': 527.23109, 'height': 667.64948}]	opacity 1 175.4401 879.37257 702.67119 1547.022
6032	f2d890d8bea3	9b3d574321bf_image	[{'x': 692.78252, 'y': 1550.99621, 'width': 339.9444, 'height': 269.12268}]	opacity 1 692.78252 1550.99621 1032.72692 1820
6033	f2e430c3888c	4e1ed0596509_image		none 1 0 0 1 1
6034	f2e5da098e85	5a0ad476493b_image	[{'x': 477.92002, 'y': 317.06667, 'width': 680.53333, 'height': 1407.46667},	opacity 1 477.92002 317.06667 1158.45335 1724.5



E	F	G	H	I	J
StudyInstanceUID	studyID	Negative for Pneumonia	Typical Appearance	Indeterminate Appearance	Atypical Appearance
f26c50ef1bd0	f26c50ef1bd0_study	0	0	0	1
f29d4d4eb110	f29d4d4eb110_study	0	1	0	0
f2a899251da3	f2a899251da3_study	0	1	0	0
f2a9e6e67eda	f2a9e6e67eda_study	0	1	0	0
f2b77c3c70c5	f2b77c3c70c5_study	1	0	0	0
f2b77c3c70c5					
f2bf2f65fe68	f2bf2f65fe68_study	0	1	0	0
f2d30ac37f7b	f2d30ac37f7b_study	0	0	0	1
f2d5dd65d7d2	f2d5dd65d7d2_study	0	0	1	0
f2d890d8bea3	f2d890d8bea3_study	0	0	1	0
f2e430c3888c	f2e430c3888c_study	1	0	0	0
f2e5da098e85	f2e5da098e85_study	0	1	0	0
f2ee15af6c4d	f2ee15af6c4d_study	0	1	0	0
f317383c5831	f317383c5831_study	0	0	1	0
f32c2b029aca	f32c2b029aca_study	0	1	0	0
f32dd337bd1a	f32dd337bd1a_study	1	0	0	0

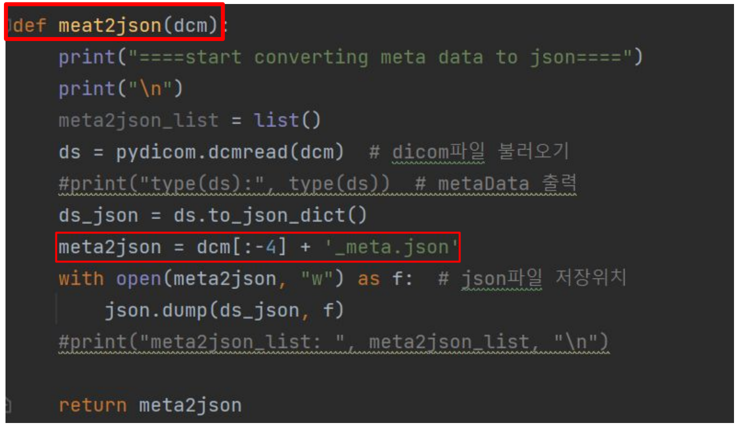
- dicom 메타데이터



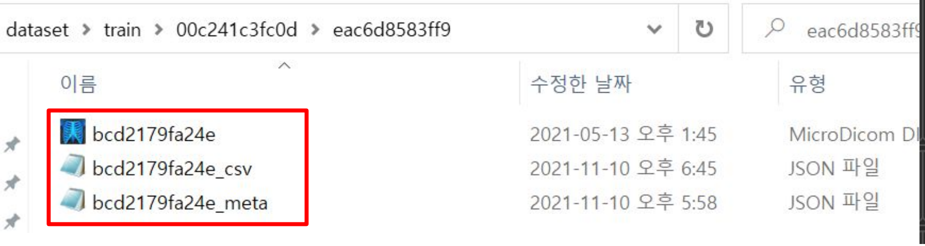
```
D:\Anaconda\envs\colorectal\python.exe D:/final_colorectal_gastric_train/Train_Model/2.DataPrepare/hadling_json.py
Dataset.file_meta -------------------------------
(0002, 0002) Media Storage SOP Class UID         UI: Digital X-Ray Image Storage - For Presentation
(0002, 0003) Media Storage SOP Instance UID      UI: 1.2.840.113654.2.70.1.98228156994025014417702239332401800074
(0002, 0010) Transfer Syntax UID                 UI: Explicit VR Little Endian
-------------------------------------------------
(0008, 0005) Specific Character Set              CS: 'ISO_IR 100'
(0008, 0008) Image Type                          CS: ['DERIVED', 'PRIMARY', '']
(0008, 0016) SOP Class UID                       UI: 71228e4340de
(0008, 0018) SOP Instance UID                    UI: 51759b5579bc
(0008, 0020) Study Date                          DA: '96fc21dd2b1f'
(0008, 0030) Study Time                          TM: '13e700cac7f0'
(0008, 0050) Accession Number                    SH: '0dc10cf540cf'
(0008, 0060) Modality                            CS: 'DX'
(0009, 0010) Private Creator                     LO: 'GEIIS'
(0010, 0010) Patient's Name                      PN: '3a0c965d2601'
(0010, 0020) Patient ID                          LO: '2c00dc1ead80'
(0010, 0040) Patient's Sex                       CS: 'M'
(0012, 0063) De-identification Method            LO: 'CTP Default:  based on DICOM PS3.15 AnnexE. Details in 0012,0064'
(0012, 0064)  De-identification Method Code Sequence  6 item(s) ----
   (0008, 0100) Code Value                          SH: '113100'
   (0008, 0102) Coding Scheme Designator            SH: 'DCM'
   (0008, 0104) Code Meaning                        LO: 'Basic Application Confidentiality Profile'
   ---------
   (0008, 0100) Code Value                          SH: '113105'
   (0008, 0102) Coding Scheme Designator            SH: 'DCM'
   (0008, 0104) Code Meaning                        LO: 'Clean Descriptors Option'
   ---------
   (0008, 0100) Code Value                          SH: '113107'
   (0008, 0102) Coding Scheme Designator            SH: 'DCM'
   (0008, 0104) Code Meaning                        LO: 'Retain Longitudinal Temporal Information Modified Dates Option'
   ---------
   (0008, 0100) Code Value                          SH: '113108'
   (0008, 0102) Coding Scheme Designator            SH: 'DCM'
   (0008, 0104) Code Meaning                        LO: 'Retain Patient Characteristics Option'
```

Pre-Processing; make_json_real.py

- dicom 메타데이터 / csv 정보 ⇒ 각각의 json으로 저장



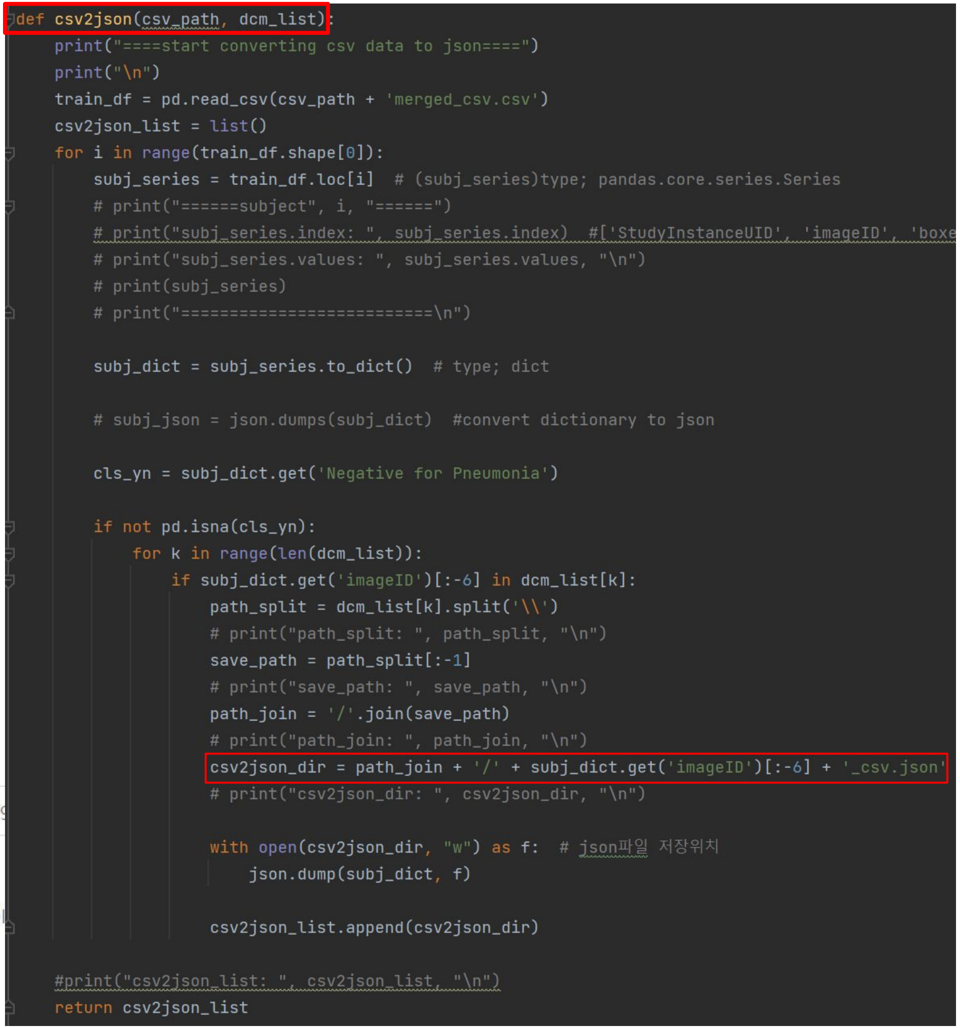
```
def meat2json(dcm):
    print("====start converting meta data to json====")
    print("\n")
    meta2json_list = list()
    ds = pydicom.dcmread(dcm)  # dicom파일 불러오기
    #print("type(ds):", type(ds))  # metaData 출력
    ds_json = ds.to_json_dict()
    meta2json = dcm[:-4] + '_meta.json'
    with open(meta2json, "w") as f:  # json파일 저장위치
        json.dump(ds_json, f)
    #print("meta2json_list: ", meta2json_list, "\n")

    return meta2json
```



dataset > train > 00c241c3fc0d > eac6d8583ff9

eac6d8583ff9

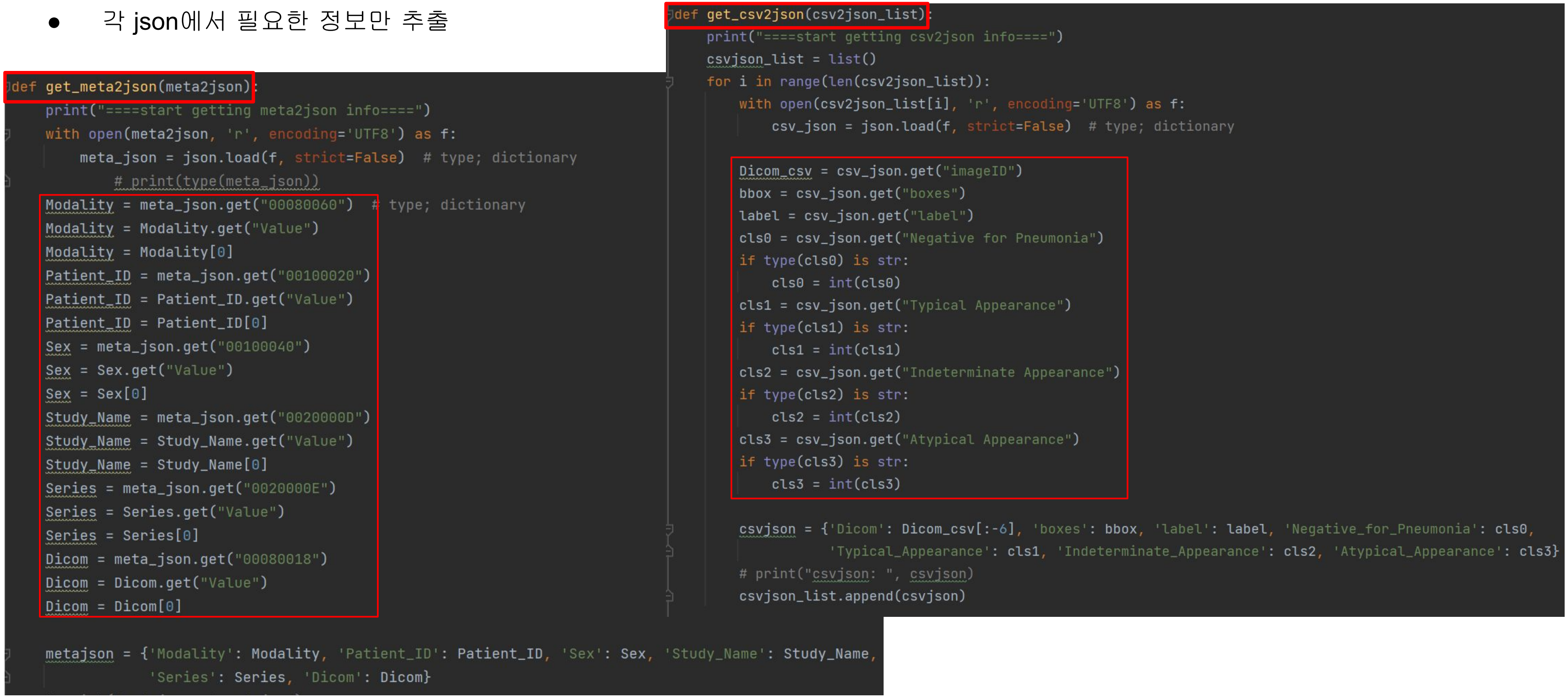
이름	수정한 날짜	유형
bcd2179fa24e	2021-05-13 오후 1:45	MicroDicom DI
bcd2179fa24e_csv	2021-11-10 오후 6:45	JSON 파일
bcd2179fa24e_meta	2021-11-10 오후 5:58	JSON 파일



```
def csv2json(csv_path, dcm_list):
    print("====start converting csv data to json====")
    print("\n")
    train_df = pd.read_csv(csv_path + 'merged_csv.csv')
    csv2json_list = list()
    for i in range(train_df.shape[0]):
        subj_series = train_df.loc[i]  # (subj_series)type; pandas.core.series.Series
        # print("======subject", i, "======")
        # print("subj_series.index: ", subj_series.index)  #['StudyInstanceUID', 'imageID', 'boxe
        # print("subj_series.values: ", subj_series.values, "\n")
        # print(subj_series)
        # print("========================\n")

        subj_dict = subj_series.to_dict()  # type; dict

        # subj_json = json.dumps(subj_dict)  #convert dictionary to json

        cls_yn = subj_dict.get('Negative for Pneumonia')

        if not pd.isna(cls_yn):
            for k in range(len(dcm_list)):
                if subj_dict.get('imageID')[:-6] in dcm_list[k]:
                    path_split = dcm_list[k].split('\\')
                    # print("path_split: ", path_split, "\n")
                    save_path = path_split[:-1]
                    # print("save_path: ", save_path, "\n")
                    path_join = '/'.join(save_path)
                    # print("path_join: ", path_join, "\n")
                    csv2json_dir = path_join + '/' + subj_dict.get('imageID')[:-6] + '_csv.json'
                    # print("csv2json_dir: ", csv2json_dir, "\n")

                    with open(csv2json_dir, "w") as f:  # json파일 저장위치
                        json.dump(subj_dict, f)

                    csv2json_list.append(csv2json_dir)

    #print("csv2json_list: ", csv2json_list, "\n")
    return csv2json_list
```

Pre-Processing; make_json_real.py

- 각 json에서 필요한 정보만 추출

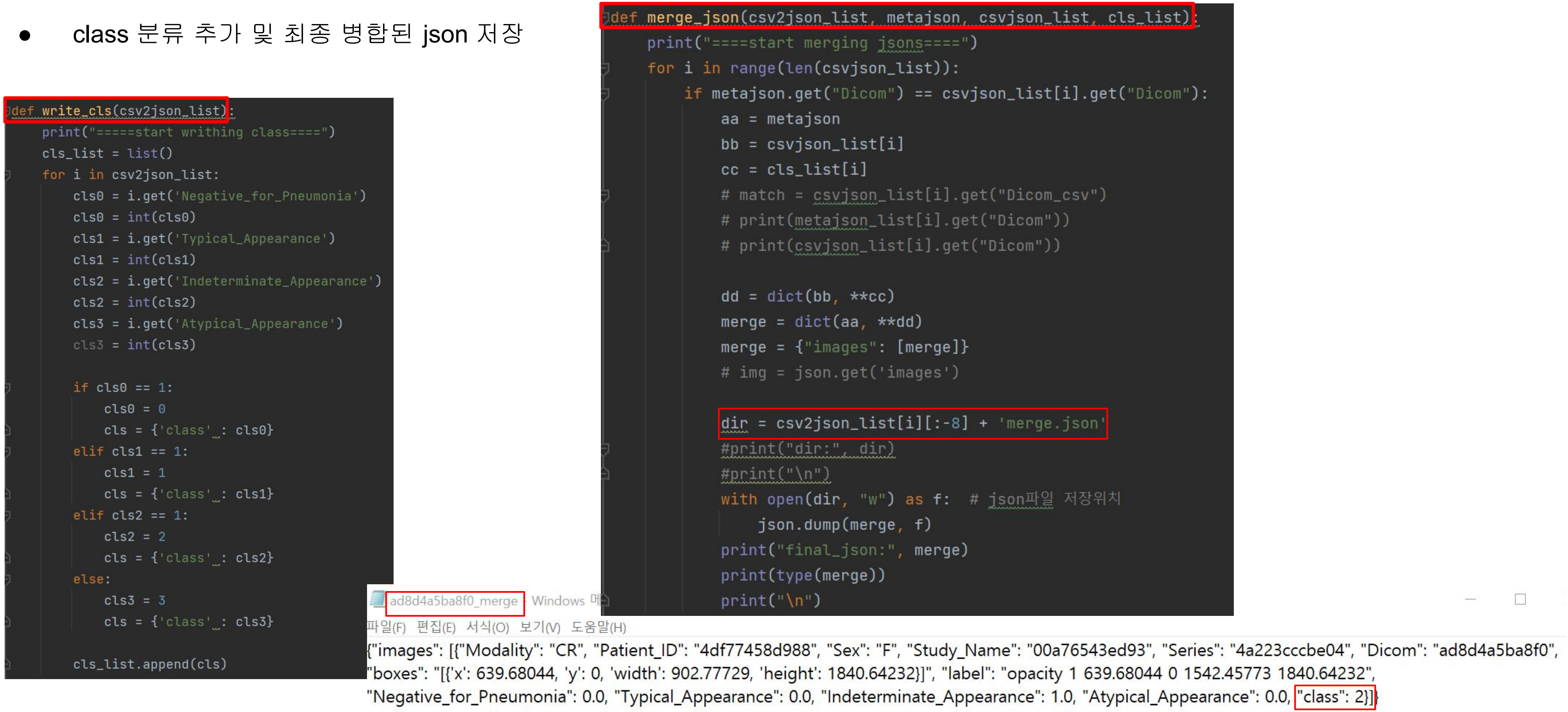


```python
def get_meta2json(meta2json):
    print("====start getting meta2json info====")
    with open(meta2json, 'r', encoding='UTF8') as f:
        meta_json = json.load(f, strict=False)  # type; dictionary
            # print(type(meta_json))
    Modality = meta_json.get("00080060")  # type; dictionary
    Modality = Modality.get("Value")
    Modality = Modality[0]
    Patient_ID = meta_json.get("00100020")
    Patient_ID = Patient_ID.get("Value")
    Patient_ID = Patient_ID[0]
    Sex = meta_json.get("00100040")
    Sex = Sex.get("Value")
    Sex = Sex[0]
    Study_Name = meta_json.get("0020000D")
    Study_Name = Study_Name.get("Value")
    Study_Name = Study_Name[0]
    Series = meta_json.get("0020000E")
    Series = Series.get("Value")
    Series = Series[0]
    Dicom = meta_json.get("00080018")
    Dicom = Dicom.get("Value")
    Dicom = Dicom[0]

    metajson = {'Modality': Modality, 'Patient_ID': Patient_ID, 'Sex': Sex, 'Study_Name': Study_Name,
                'Series': Series, 'Dicom': Dicom}
```

```python
def get_csv2json(csv2json_list):
    print("====start getting csv2json info====")
    csvjson_list = list()
    for i in range(len(csv2json_list)):
        with open(csv2json_list[i], 'r', encoding='UTF8') as f:
            csv_json = json.load(f, strict=False)  # type; dictionary

        Dicom_csv = csv_json.get("imageID")
        bbox = csv_json.get("boxes")
        label = csv_json.get("label")
        cls0 = csv_json.get("Negative for Pneumonia")
        if type(cls0) is str:
            cls0 = int(cls0)
        cls1 = csv_json.get("Typical Appearance")
        if type(cls1) is str:
            cls1 = int(cls1)
        cls2 = csv_json.get("Indeterminate Appearance")
        if type(cls2) is str:
            cls2 = int(cls2)
        cls3 = csv_json.get("Atypical Appearance")
        if type(cls3) is str:
            cls3 = int(cls3)

        csvjson = {'Dicom': Dicom_csv[:-6], 'boxes': bbox, 'label': label, 'Negative_for_Pneumonia': cls0,
                   'Typical_Appearance': cls1, 'Indeterminate_Appearance': cls2, 'Atypical_Appearance': cls3}
        # print("csvjson: ", csvjson)
        csvjson_list.append(csvjson)
```

Pre-Processing; make_json_real.py

- class 분류 추가 및 최종 병합된 json 저장



```python
def write_cls(csv2json_list):
    print("=====start writhing class====")
    cls_list = list()
    for i in csv2json_list:
        cls0 = i.get('Negative_for_Pneumonia')
        cls0 = int(cls0)
        cls1 = i.get('Typical_Appearance')
        cls1 = int(cls1)
        cls2 = i.get('Indeterminate_Appearance')
        cls2 = int(cls2)
        cls3 = i.get('Atypical_Appearance')
        cls3 = int(cls3)

        if cls0 == 1:
            cls0 = 0
            cls = {'class' : cls0}
        elif cls1 == 1:
            cls1 = 1
            cls = {'class' : cls1}
        elif cls2 == 1:
            cls2 = 2
            cls = {'class' : cls2}
        else:
            cls3 = 3
            cls = {'class' : cls3}

        cls_list.append(cls)
```

```python
def merge_json(csv2json_list, metajson, csvjson_list, cls_list):
    print("====start merging jsons====")
    for i in range(len(csvjson_list)):
        if metajson.get("Dicom") == csvjson_list[i].get("Dicom"):
            aa = metajson
            bb = csvjson_list[i]
            cc = cls_list[i]
            # match = csvjson_list[i].get("Dicom_csv")
            # print(metajson_list[i].get("Dicom"))
            # print(csvjson_list[i].get("Dicom"))

            dd = dict(bb, **cc)
            merge = dict(aa, **dd)
            merge = {"images": [merge]}
            # img = json.get('images')

            dir = csv2json_list[i][:-8] + 'merge.json'
            #print("dir:", dir)
            #print("\n")
            with open(dir, "w") as f:  # json파일 저장위치
                json.dump(merge, f)
            print("final_json:", merge)
            print(type(merge))
            print("\n")
```

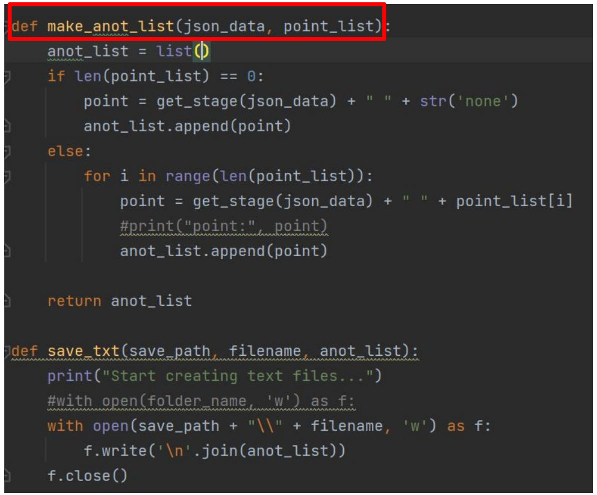
ad8d4a5ba8f0_merge - Windows 메
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
{"images": [{"Modality": "CR", "Patient_ID": "4df77458d988", "Sex": "F", "Study_Name": "00a76543ed93", "Series": "4a223cccbe04", "Dicom": "ad8d4a5ba8f0",
"boxes": "[{'x': 639.68044, 'y': 0, 'width': 902.77729, 'height': 1840.64232}]", "label": "opacity 1 639.68044 0 1542.45773 1840.64232",
"Negative_for_Pneumonia": 0.0, "Typical_Appearance": 0.0, "Indeterminate_Appearance": 1.0, "Atypical_Appearance": 0.0, "class": 2}]}
```

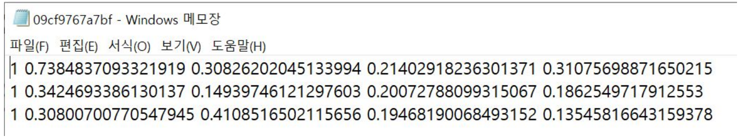
Pre-Processing; make_txt_c18.py

- json to txt convert for detection-YOLOv5

- json에서 class와 bbox 정보를 추출하여 txt로 저장



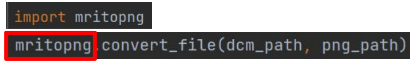
```python
def make_anot_list(json_data, point_list):
    anot_list = list()
    if len(point_list) == 0:
        point = get_stage(json_data) + " " + str('none')
        anot_list.append(point)
    else:
        for i in range(len(point_list)):
            point = get_stage(json_data) + " " + point_list[i]
            #print("point:", point)
            anot_list.append(point)

    return anot_list

def save_txt(save_path, filename, anot_list):
    print("Start creating text files...")
    #with open(folder_name, 'w') as f:
    with open(save_path + "\\" + filename, 'w') as f:
        f.write('\n'.join(anot_list))
    f.close()
```



09cf9767a7bf - Windows 메모장

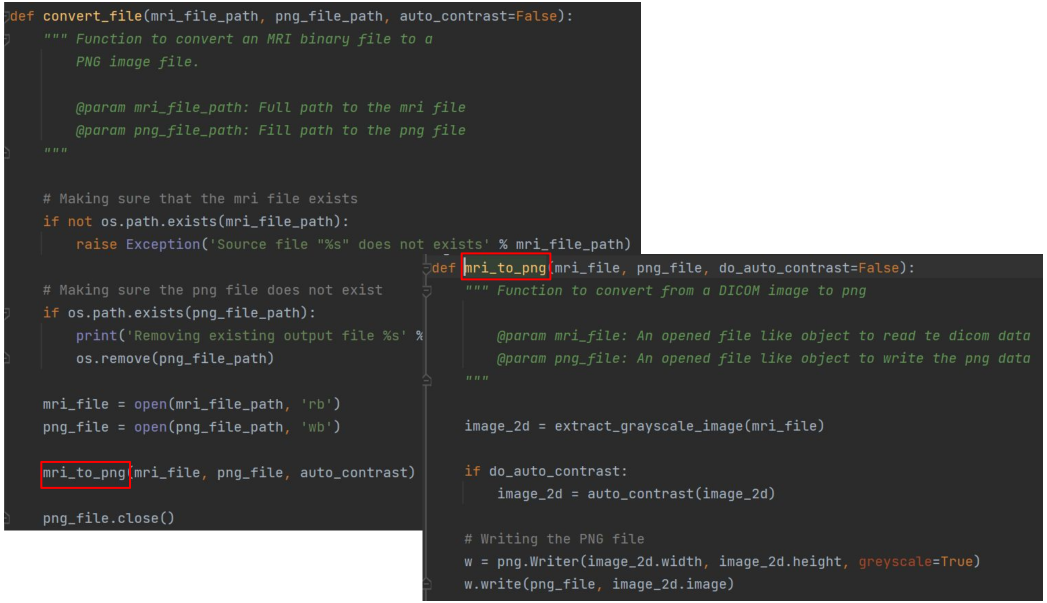
```
1 0.7384837093321919 0.30826202045133994 0.21402918236301371 0.31075698871650215
1 0.3424693386130137 0.14939746121297603 0.20072788099315067 0.1862549717912553
1 0.30800700770547945 0.4108516502115656 0.19468190068493152 0.13545816643159378
```

- dicom to png 이미지 형식 변환



```python
import mritopng
mritopng.convert_file(dcm_path, png_path)
```

mritopng 라이브러리 __init__.py



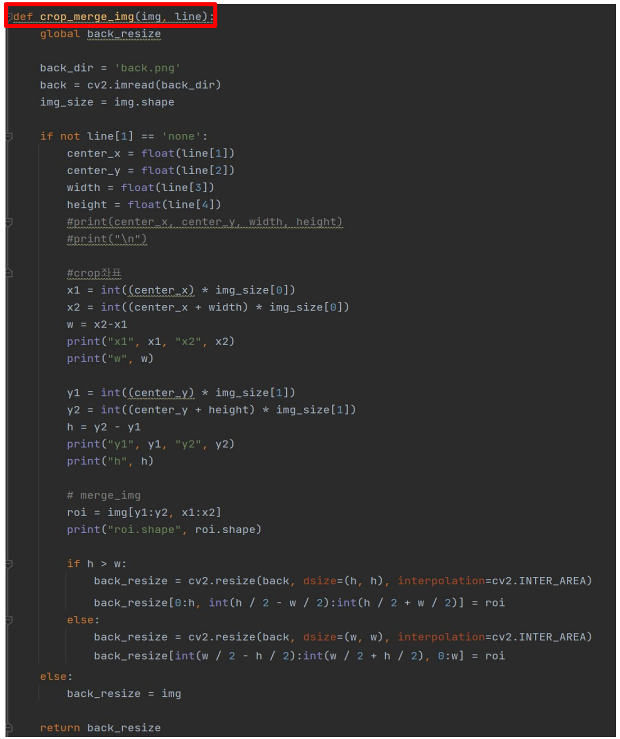
```python
def convert_file(mri_file_path, png_file_path, auto_contrast=False):
    """ Function to convert an MRI binary file to a
        PNG image file.

        @param mri_file_path: Full path to the mri file
        @param png_file_path: Fill path to the png file
    """

    # Making sure that the mri file exists
    if not os.path.exists(mri_file_path):
        raise Exception('Source file "%s" does not exists' % mri_file_path)

    # Making sure the png file does not exist
    if os.path.exists(png_file_path):
        print('Removing existing output file %s' %
        os.remove(png_file_path)

    mri_file = open(mri_file_path, 'rb')
    png_file = open(png_file_path, 'wb')

    mri_to_png(mri_file, png_file, auto_contrast)

    png_file.close()
```

```python
def mri_to_png(mri_file, png_file, do_auto_contrast=False):
    """ Function to convert from a DICOM image to png

        @param mri_file: An opened file like object to read te dicom data
        @param png_file: An opened file like object to write the png data
    """

    image_2d = extract_grayscale_image(mri_file)

    if do_auto_contrast:
        image_2d = auto_contrast(image_2d)

    # Writing the PNG file
    w = png.Writer(image_2d.width, image_2d.height, greyscale=True)
    w.write(png_file, image_2d.image)
```

Pre-Processing; crop_make_folder.py

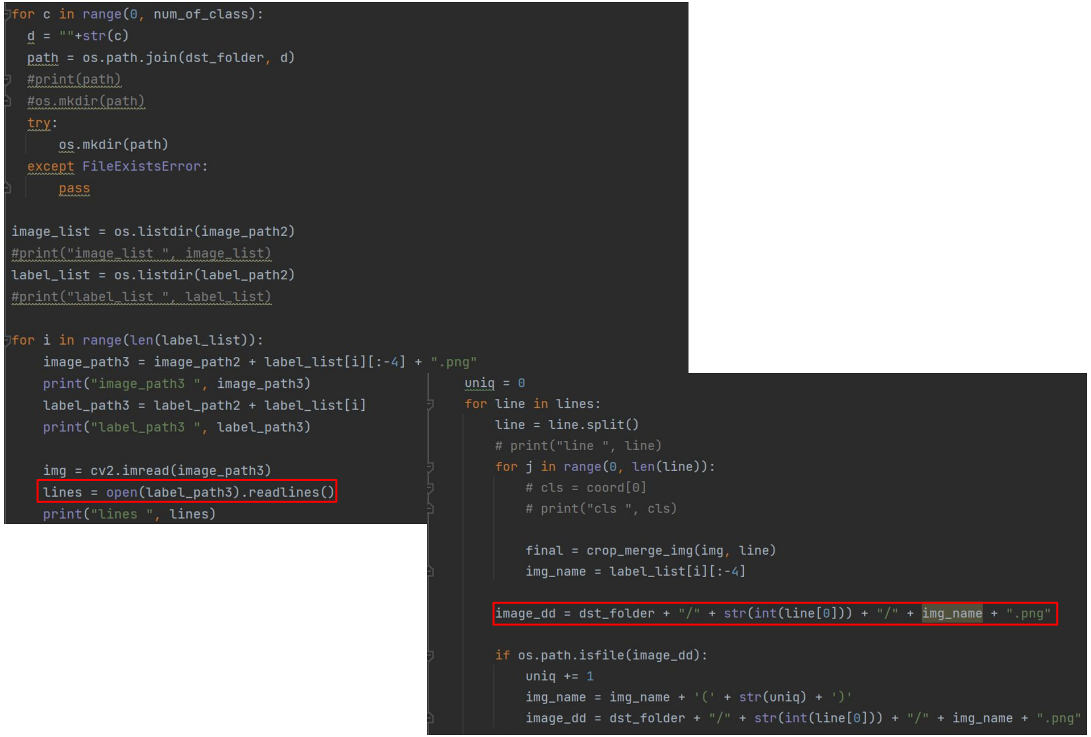
- ROI crop & foldering image by class for classification-EfficientNetB4

- ROI crop & 검은 배경에 센터 맞춰 붙이기



```python
def crop_merge_img(img, line):
    global back_resize

    back_dir = 'back.png'
    back = cv2.imread(back_dir)
    img_size = img.shape

    if not line[1] == 'none':
        center_x = float(line[1])
        center_y = float(line[2])
        width = float(line[3])
        height = float(line[4])
        #print(center_x, center_y, width, height)
        #print("\n")

        #crop좌표
        x1 = int((center_x) * img_size[0])
        x2 = int((center_x + width) * img_size[0])
        w = x2-x1
        print("x1", x1, "x2", x2)
        print("w", w)

        y1 = int((center_y) * img_size[1])
        y2 = int((center_y + height) * img_size[1])
        h = y2 - y1
        print("y1", y1, "y2", y2)
        print("h", h)

        # merge_img
        roi = img[y1:y2, x1:x2]
        print("roi.shape", roi.shape)

        if h > w:
            back_resize = cv2.resize(back, dsize=(h, h), interpolation=cv2.INTER_AREA)
            back_resize[0:h, int(h / 2 - w / 2):int(h / 2 + w / 2)] = roi
        else:
            back_resize = cv2.resize(back, dsize=(w, w), interpolation=cv2.INTER_AREA)
            back_resize[int(w / 2 - h / 2):int(w / 2 + h / 2), 0:w] = roi
    else:
        back_resize = img

    return back_resize
```

- class에 맞춰 foldering



```python
for c in range(0, num_of_class):
    d = ""+str(c)
    path = os.path.join(dst_folder, d)
    #print(path)
    #os.mkdir(path)
    try:
        os.mkdir(path)
    except FileExistsError:
        pass

image_list = os.listdir(image_path2)
#print("image_list ", image_list)
label_list = os.listdir(label_path2)
#print("label_list ", label_list)

for i in range(len(label_list)):
    image_path3 = image_path2 + label_list[i][:-4] + ".png"
    print("image_path3 ", image_path3)
    label_path3 = label_path2 + label_list[i]
    print("label_path3 ", label_path3)

    img = cv2.imread(image_path3)
    lines = open(label_path3).readlines()
    print("lines ", lines)
```

```python
    uniq = 0
    for line in lines:
        line = line.split()
        # print("line ", line)
        for j in range(0, len(line)):
            # cls = coord[0]
            # print("cls ", cls)

            final = crop_merge_img(img, line)
            img_name = label_list[i][:-4]

        image_dd = dst_folder + "/" + str(int(line[0])) + "/" + img_name + ".png"

        if os.path.isfile(image_dd):
            uniq += 1
            img_name = img_name + '(' + str(uniq) + ')'
            image_dd = dst_folder + "/" + str(int(line[0])) + "/" + img_name + ".png"
```

Pre-Processing; crop_make_folder.py

- 이미지 처리 결과