

## Workshop II: Object Detection

---

### Assignment Workshop 02

1. ให้นักศึกษาทำโจทย์ของไฟล์ Workshop\_II-02\_ObjectDetection.ipynb ให้เรียบร้อยพร้อมทั้งมาตอบคำถามต่อไปนี้

1.1 อธิบายวิธีการเตรียมข้อมูล (Data Preparation) ของนักศึกษาที่ใช้ในการจัดการข้อมูลก่อนที่จะนำтренโนเมเดลครั้งนี้ สามารถแคปภาพหน้าจอแล้วมาอธิบายได้

1.2 ทำการสร้างและปรับแต่งโมเดลตามโครงสร้างที่กำหนด (สามารถปรับโครงสร้างได้ตามความคิดสร้างสรรค์ พร้อมอธิบายการออกแบบโมเดล) จากนั้นทำการ trenโนเมเดล โดยเก็บค่า Training Loss และ Validation Loss นำมาสร้างกราฟ พร้อมวิเคราะห์ว่าโมเดลมีการ Overfitting หรือ Underfitting หรือไม่ พร้อมอธิบายเหตุผล

1.3 ทำการวัดผลของโมเดลด้วยการใช้คะแนน IOU, Precision, Recall และ F1-Score

1.4 ทำการแสดงผลของการทำนายด้วยโมเดลเทียบระหว่างภาพ bounding box ของ labels และ predictions

## 1.1 อธิบายวิธีการเตรียมข้อมูล (Data Preparation) ของนักศึกษาที่ใช้ในการจัดการข้อมูล

ก่อนที่จะนำเท่านโนเมเดลครั้งนี้ สามารถแคปภาพหน้าจอแล้วมาอธิบายได้

### ① กำหนดรูปแบบ Custom Dataset ชนชั้น Dataset

```
class SyntheticDetectionDataset(Dataset):
    def __init__(self, images_path, annotations_file, transform=None):
        self.images_path = images_path
        self.coco = COCO(annotations_file)
        self.transform = transform
        self.image_ids = list(self.coco.imgs.keys())

    def __len__(self):
        return len(self.image_ids)

    def __getitem__(self, idx):
        image_id = self.image_ids[idx]
        image_info = self.coco imgs[image_id]
        image_path = os.path.join(self.images_path, image_info['file_name']).replace("\\", "/")
        image = Image.open(image_path).convert('L')
        image = np.array(image, dtype=np.float32) / 255.

        #load annotations
        anns_ids = self.coco.getAnnIds(imgIds=image_id)
        anns = self.coco.loadAnns(anns_ids)

        #prepare bbox and label
        bboxes = [ann['bbox'] for ann in anns]
        labels = [ann['category_id'] for ann in anns]

        #normalize bbox [x,y,w,h]
        bboxes = np.array(bboxes, dtype=np.float32)
        bboxes[:, 0] /= image_info['width']
        bboxes[:, 1] /= image_info['height']
        bboxes[:, 2] /= image_info['width']
        bboxes[:, 3] /= image_info['height']

        bboxes = torch.tensor(bboxes, dtype=torch.float32)
        labels = torch.tensor(labels, dtype=torch.int64)

        #create target dictionary
        target = {'bboxes': bboxes,
                  'labels': labels}

        if self.transform:
            image = self.transform(image)

        return image, target
```

คำอธิบาย:

- การอ่านไฟล์ภาพ (image)
- การแปลงภาพเป็น grayscale
- การบันทึกค่า pixel ให้เป็น 0-1
- การโหลด annotation ของแต่ละภาพ
- การจัดรูปแบบ annotation ให้เป็น tensor
- การบันทึกค่า bounding box และ label ให้เป็น tensor
- การบันทึกค่า bounding box ให้เป็น 0-1
- การสร้าง target dictionary สำหรับ model
- การแปลงภาพตาม transform ที่กำหนด

```
train_dir = "/content/train"
test_dir = "/content/test"
val_dir = "/content/val"

train_anno = "/content/train.json"
test_anno = "/content/test.json"
val_anno = "/content/val.json"

train_transform = transforms.Compose([
    transforms.ToTensor()
])

test_transform = transforms.Compose([
    transforms.ToTensor()
])

val_transform = transforms.Compose([
    transforms.ToTensor()
])
```

```
train_dataset = SyntheticDetectionDataset(train_dir, train_anno, transform=train_transform)
test_dataset = SyntheticDetectionDataset(test_dir, test_anno, transform=test_transform)
val_dataset = SyntheticDetectionDataset(val_dir, val_anno, transform=val_transform)

train_loader = DataLoader(train_dataset, batch_size=16, shuffle=True)
test_loader = DataLoader(test_dataset, batch_size=16, shuffle=False)
val_loader = DataLoader(val_dataset, batch_size=16, shuffle=False)
```

① in Dataset ก็ยังสามารถเก็บใน DataLoader  
เพื่อไปรับข้อมูลใน model

→ ② กำหนด path ที่ต้องใช้ใน function transform

1.2 ทำการสร้างและปรับแต่งโมเดลตามโครงสร้างที่กำหนด (สามารถปรับโครงสร้างได้ตามความคิดสร้างสรรค์ พร้อมอธิบายการออกแบบโมเดล) จากนั้นทำการเทรนโมเดล โดยเก็บค่า Training Loss และ Validation Loss นำมาสร้างกราฟ พร้อมวิเคราะห์ว่าโมเดลมีการ Overfitting หรือ Underfitting หรือไม่ พร้อมอธิบายเหตุผล

```
model = SimpleObjectDetector().to(device)
summary(model, (1, 64, 64))

-----  

Layer (type)           Output Shape        Param #  

=====  

    Conv2d-1            [-1, 32, 64, 64]      320  

    MaxPool2d-2          [-1, 32, 32, 32]      0  

    Conv2d-3            [-1, 64, 32, 32]     18,496  

    MaxPool2d-4          [-1, 64, 16, 16]      0  

    Conv2d-5            [-1, 128, 16, 16]    73,856  

    MaxPool2d-6          [-1, 128, 8, 8]       0  

    Linear-7             [-1, 128]         1,048,704  

    Linear-8             [-1, 70]          9,030  

    Linear-9             [-1, 128]         9,088  

    Linear-10            [-1, 5]           645  

-----  

Total params: 1,160,139  

Trainable params: 1,160,139  

Non-trainable params: 0  

-----  

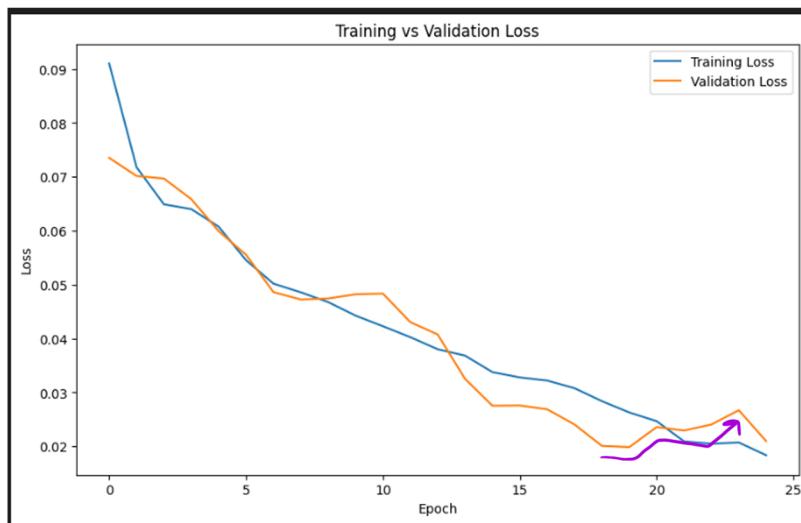
Input size (MB): 0.02  

Forward/backward pass size (MB): 2.19  

Params size (MB): 4.43  

Estimated Total Size (MB): 6.63  

-----
```



Model อาจกำลัง overfit เนื่องจาก Loss ช่วงท้ายมีการเพิ่มขึ้น

### 1.3 ทำการวัดผลของโมเดลด้วยการใช้คะแนน IOU, Precision, Recall และ F1-Score

```
predictions, ground_truths = test_model(model, test_dataset) ↗ 98% acc test
precision, recall, f1_score, mean_iou = calculate_performance(predictions, ground_truths)

print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1-Score: {f1_score:.4f}")
print(f"Mean IoU: {mean_iou:.4f}")

[0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0] ground truths
[0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0] predictions
Precision: 0.7333 } average by weight
Recall: 0.7333
F1-Score: 0.7333
Mean IoU: 0.7582
```

### 1.4 ทำการแสดงผลของการทำนายด้วยโมเดลเทียบระหว่างภาพ bounding box ของ labels และ predictions

