

# VST HW2

## 1. Experiment Setup (Data pre-process, Hyperparameters,... )

### (1) Data preprocess

使用[1]yolox 官方 github 預設的 transform augmentation，如下圖。

```
# ----- transform config ----- #
# prob of applying mosaic aug
self.mosaic_prob = 1.0
# prob of applying mixup aug
self.mixup_prob = 1.0
# prob of applying hsv aug
self.hsv_prob = 1.0
# prob of applying flip aug
self.flip_prob = 0.5
# rotation angle range, for example, if set to 2, the true range is (-2, 2)
self.degrees = 10.0
# translate range, for example, if set to 0.1, the true range is (-0.1, 0.1)
self.translate = 0.1
self.mosaic_scale = (0.1, 2)
# apply mixup aug or not
self.enable_mixup = True
self.mixup_scale = (0.5, 1.5)
# shear angle range, for example, if set to 2, the true range is (-2, 2)
self.shear = 2.0
```

再使用[1]yolox 官方 github 的 data\_augment.py 進圖片前處理，如下圖。

```
def preproc(img, input_size, swap=(2, 0, 1)):
    if len(img.shape) == 3:
        padded_img = np.ones((input_size[0], input_size[1], 3), dtype=np.uint8) * 114
    else:
        padded_img = np.ones(input_size, dtype=np.uint8) * 114

    r = min(input_size[0] / img.shape[0], input_size[1] / img.shape[1])
    resized_img = cv2.resize(
        img,
        (int(img.shape[1] * r), int(img.shape[0] * r)),
        interpolation=cv2.INTER_LINEAR,
    ).astype(np.uint8)
    padded_img[: int(img.shape[0] * r), : int(img.shape[1] * r)] = resized_img

    padded_img = padded_img.transpose(swap)
    padded_img = np.ascontiguousarray(padded_img, dtype=np.float32)
    return padded_img, r
```

### (2) Hyperparameters

- Original

```
self.num_classes = 1
```

```
# activation name. For example, if using "relu", then "silu" will be replaced to "relu".
self.act = "silu"

# ----- dataloader config ----- #
# set worker to 4 for shorter dataloader init time
# If your training process cost many memory, reduce this value.
self.data_num_workers = 4
self.input_size = (640, 640) # (height, width)
# Actual multiscale ranges: [640 - 5 * 32, 640 + 5 * 32].
# To disable multiscale training, set the value to 0.
self.multiscale_range = 5
```

```
# ----- training config ----- #
# epoch number used for warmup
self.warmup_epochs = 5
# max training epoch
self.max_epoch = 300
# minimum learning rate during warmup
self.warmup_lr = 0
self.min_lr_ratio = 0.05
# learning rate for one image. During training, lr will multiply batchsize.
self.basic_lr_per_img = 0.01 / 64.0
# name of LRScheduler
self.scheduler = "yoloxwarmcos"
# last #epoch to close augmentation like mosaic
self.no_aug_epochs = 15
# apply EMA during training
self.ema = True

# weight decay of optimizer
self.weight_decay = 5e-4
# momentum of optimizer
self.momentum = 0.9
# log period in iter, for example,
# if set to 1, user could see log every iteration.
self.print_interval = 10
# eval period in epoch, for example,
# if set to 1, model will be evaluate after every epoch.
self.eval_interval = 10
# save history checkpoint or not.
# If set to False, yolox will only save latest and best ckpt.
self.save_history_ckpt = True
# name of experiment
self.exp_name = os.path.split(os.path.realpath(__file__))[1].split(".")[0]
```

```
# ----- testing config ----- #
# output image size during evaluation/test
self.test_size = (640, 640)
# confidence threshold during evaluation/test,
# boxes whose scores are less than test_conf will be filtered
self.test_conf = 0.01
# nms threshold
self.nmsthre = 0.65
```

- with SE module  
Hyperparameters 都跟 Original 的一樣，只有 no\_aug\_epochs 改成設為 20 這個地方不一樣而已。

2. Explain which layer you add SE modules to and compare the corresponding results



3. car\_coco 資料夾移至 HW2\_312551093/Code/Original/datasets 與 HW2\_312551093/Code/SE/datasets 資料夾裡
4. 將 car\_coco 資料夾裡的 train 跟 val 資料夾改名成 train2017、val2017
5. 在 car\_coco 資料夾裡創 1 個名字叫 annotations 的資料夾，裡面放 train\_labels 跟 val\_labels 的 JSON 檔

## (2) Installation

```
cd HW2_312551093/Code/Original

conda env create -f environment.yml

pip3 install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu118

pip install -v -e .
```

## (3) Train

with SE module:

```
cd HW2_312551093/Code/SE

python -m yolox.tools.train -f exps/example/custom/yolox_s.py -b 64 --fp16 -o -c yolox_s.pth
```

without SE module:

```
cd HW2_312551093/Code/Original

python -m yolox.tools.train -f exps/example/custom/yolox_s.py -b 64 --fp16 -o -c yolox_s.pth
```

## (4) Validation

with SE module:

```
cd HW2_312551093/Code/SE

python -m yolox.tools.demo --path datasets/car_coco/val2017 --device {gpu} --save_result -f
exps/example/custom/yolox_s.py -c best_ckpt.pth --fp16 image
```

without SE module:

```
cd HW2_312551093/Code/Original

python -m yolox.tools.demo --path datasets/car_coco/val2017 --device {gpu} --save_result -f
exps/example/custom/yolox_s.py -c best_ckpt.pth --fp16 image
```

## (5) Test

with SE module:

```
cd HW2_312551093/Code/SE
```

```
python -m yolox.tools.demo --path datasets/car_coco/test --device {gpu} --save_result -f  
exps/example/custom/yolox_s.py -c best_ckpt.pth --fp16 image
```

without SE module:

```
cd HW2_312551093/Code/Original
```

```
python -m yolox.tools.demo --path datasets/car_coco/test --device {gpu} --save_result -f  
exps/example/custom/yolox_s.py -c best_ckpt.pth --fp16 image
```

## (6) Evaluating validation results

with SE module:

```
git clone https://github.com/rafaelpadilla/Object-Detection-Metrics
```

```
cd Object-Detection-Metrics
```

把(4)部分輸出的 txt 檔全部丟到 Object-Detection-Metrics/detections/se 資料夾裡(資料夾要自己建)

```
python pascalvoc.py -gt ./groundtruths/val_labels -det ./detections/se -t 0.85 -gtformat xywh -  
detformat xyrb -gtcoords rel -detcoords abs -imgsize "(1920,1080)" -sp results
```

without SE module:

```
git clone https://github.com/rafaelpadilla/Object-Detection-Metrics
```

```
cd Object-Detection-Metrics
```

把(4)部分輸出的 txt 檔全部丟到 Object-Detection-Metrics/detections/original 資料夾裡(資料夾要自己建)

```
python pascalvoc.py -gt ./groundtruths/val_labels -det ./detections/original -t 0.85 -gtformat xywh -  
detformat xyrb -gtcoords rel -detcoords abs -imgsize "(1920,1080)" -sp results
```

## Reference

[1] <https://github.com/Megvii-BaseDetection/YOLOX>

[2] [https://blog.csdn.net/qq\\_44824148/article/details/122855288?spm=1001.2101.3001.6650.1&utm\\_medium=distribute.pc\\_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7ERate-1-122855288-blog-](https://blog.csdn.net/qq_44824148/article/details/122855288?spm=1001.2101.3001.6650.1&utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7ERate-1-122855288-blog-)

126002838.235%5Ev38%5Epc\_relevant\_anti\_t3\_base&depth\_1-  
utm\_source=distribute.pc\_relevant.none-task-blog-  
2%7Edefault%7ECTRLIST%7ERate-1-122855288-blog-  
126002838.235%5Ev38%5Epc\_relevant\_anti\_t3\_base&utm\_relevant\_index=2