Unbiased Teacher for Semi-Supervised Object Detection笔记

- Paper: Unbiased Teacher for Semi-Supervised Object Detection
- Code: facebookresearch/unbiased-teacher

1. Introduction

1.1 Why

1.1.1 Motivation

- 海量数据能提升深度神经网络在不同任务中的表现；但是标注大量的数据很昂贵，半监督学习可作为一种减小标注代价的手段。

1.1.2 Challenges

- 目标检测中**类别不均衡**的特点阻碍了伪标签的使用。(The nature of **class-imbalance** in object detection tasks impedes the usage of pseudo-labeling.)
    - foreground-background imbalance
    - foreground classes imbalance

1.2 What

- **无偏老师**同时处理**伪标签问题**(标注数据中固有的类别不均衡引起的)和**过拟合问题**(标注数据稀缺引起的)。(We propose **Unbiased Teacher** to jointly address the pseudo-labeling bias issue and the overfitting issue in semi-supervised object detection, and our model performs favorably against existing works on COCO-standard, COCO-additional, and VOC.)
- 该方法以**互利**的方式同时训练学生和缓慢进步的老师，老师生成伪标签用于训练学生，学生通过指数滑动平均逐步更新老师。(**Unbiased Teacher**: an approach that jointly trains a Student and a slowly progressing Teacher in a mutually-beneficial manner, in which the Teacher generates pseudo-labels to train the Student, and the Student gradually updates the Teacher via Exponential Moving Average (EMA).)
- 分别老师和学生提供弱增广(为了生成可靠的伪标签)、强增广(为了提升学生模型的多样性)的输入。(The Teacher and Student are given weakly(to provide reliable pseudo-labels) and strongly(the diversity of Student models) augmented inputs, respectively.)

1.3 How

- 利用伪标签对RPN和ROIhead进行显式的监督学习，因此能减轻RPN和ROIhead的过拟合问题。(We utilize the pseudo-labels as explicit supervision for both RPN and ROIhead and thus alleviate the overfitting issues in both RPN and ROIhead.)
- 利用老师-学生互利模型来防止噪声伪标签产生的不利影响。(We also prevent detrimental effects due to noisy pseudo-labels by exploiting the Teacher-Student dual models.)
- 利用EMA training and Focal loss，解决因类别不均衡引起的伪标签偏差问题，提升伪标签的质量。(With the use of EMA training and the Focal loss (Lin et al., 2017b), we can address the pseudo-labeling bias problem caused by class-imbalance and thus improve the quality of pseudo-labels.)

1.4 Contributions

- 发现了目标检测任务中类别不均衡问题对半监督目标检测任务中伪标签方法的作用的限制。(By analyzing object detectors trained with limited-supervision, we identify that the nature of class-imbalance in object detection tasks impedes the effectiveness of pseudo-labeling method on SS-OD task.)
- 提出了无偏老师的方法，解决伪标签问题和过拟合问题。(We thus proposed a simple yet effective method, Unbiased Teacher, to address the pseudo-labeling bias issue caused by class-imbalance existing in ground-truth labels and the overfitting issue caused by the scarcity of labeled data.)
- 表现SOTA。(Our Unbiased Teacher achieves state-of-the-art performance on SS-OD across COCOstandard, COCO-additional, and VOC datasets. We also provide an ablation study to verify the effectiveness of each proposed component.)

2. Approach

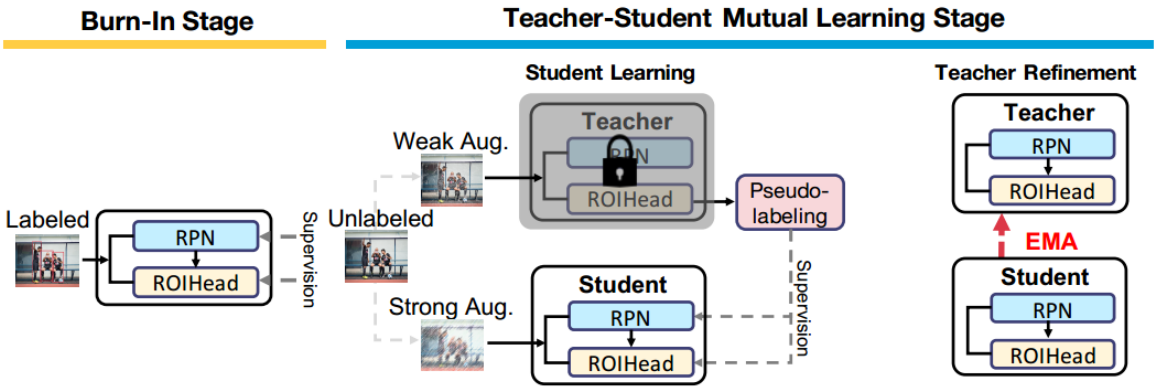

Figure 3: Overview of ***Unbiased Teacher***. Unbiased Teacher consists of two stages. ***Burn-In***: we first train the object detector using available labeled data. ***Teacher-Student Mutual Learning*** consists of two steps. **Student Learning**: the fixed teacher generates pseudo-labels to train the Student, while Teacher and Student are given weakly and strongly augmented inputs, respectively. **Teacher Refinement**: the knowledge that the Student learned is then transferred to the slowly progressing Teacher via exponential moving average (EMA) on network weights. When the detector is trained until converge in the Burn-In stage, we switch to the Teacher-Student Mutual Learning stage.

- **预热阶段**：用标注数据进行监督学习来初始化老师模型。(In the Burn-In stage, we simply train the object detector using the available supervised data to initialize the detector.)
- **老师-学生相互学习阶段**：(Our Teacher-Student Mutual Learning stage aims at evolving both Teacher and Student models via a mutual learning mechanism)
  - 老师教、学生学：老师生成伪标签来训练学生；(the Teacher generates pseudo-labels to train the Student)
  - 学生回馈、老师改善；(the Student updates the knowledge it learned back to the Teacher via exponential moving average (EMA))

2.1 BURN-IN

- 目的：**初始化检测模型**，用于生成可靠的伪标签供学生模型学习。该阶段模型收敛后，得到的模型在老师-学生相互学习阶段的初始时刻，会被复制给老师模型和学生模型。

$$\mathcal{L}_{sup} = \sum_i \mathcal{L}_{cls}^{rpn}(\boldsymbol{x}_i^s, \boldsymbol{y}_i^s) + \mathcal{L}_{reg}^{rpn}(\boldsymbol{x}_i^s, \boldsymbol{y}_i^s) + \mathcal{L}_{cls}^{roi}(\boldsymbol{x}_i^s, \boldsymbol{y}_i^s) + \mathcal{L}_{reg}^{roi}(\boldsymbol{x}_i^s, \boldsymbol{y}_i^s). \quad (1)$$

2.2 TEACHER-STUDENT MUTUAL LEARNING

- 目的：**提升伪标签生成的模型(improve the pseudo-label generation model)**

2.2.1 Student Learning with Pseudo-Labeling

- 目的：**the Student is optimized by using the pseudo-labels generated from the Teacher**

$$\theta_s \leftarrow \theta_s + \gamma \frac{\partial(\mathcal{L}_{sup} + \boldsymbol{\lambda}_u \mathcal{L}_{unsup})}{\partial \theta_s}, \quad \mathcal{L}_{unsup} = \sum_i \mathcal{L}_{cls}^{rpn}(\boldsymbol{x}_i^u, \hat{\boldsymbol{y}}_i^u) + \mathcal{L}_{cls}^{roi}(\boldsymbol{x}_i^u, \hat{\boldsymbol{y}}_i^u) \tag{2}$$

- 要点：
  - We first set a **confidence threshold** of predicted bounding boxes to **filter low-confidence predicted bounding boxes** to **prevent the consecutively detrimental effect of noisy pseudo-labels**.
  - We **remove the repetitive predictions** by applying **class-wise non-maximum suppression (NMS)** before the use of confidence thresholding as performed in STAC.
  - **Only the learnable weights of the Student model is updated** via back-propagation, because noisy pseudo-labels can affect the pseudo-label generation model (Teacher).
  - We do not apply unsupervised losses for the bounding box regression since the naive confidence thresholding is not able to filter the pseudo-labels that are potentially incorrect for bounding box regression.

2.2.2 Teacher Refinement via Exponential Moving Average

- 目的：**the Teacher is updated by gradually transferring the weights of continually learned Student model**
- 要点：The slowly progressing Teacher model can be regarded as the temporal ensemble of the Student models in different training iterations.

$$\theta_t \leftarrow \alpha \theta_t + (1 - \alpha)\theta_s. \tag{3}$$

2.3 BIAS IN PSEUDO-LABEL

- 目的：**解决生成的伪标签中的类别不均衡问题(address the crucial imbalance issue in generated pseudo-labels by Focal loss and EMA training)**
- 要点：
  - Multi-class Focal loss makes the model focus on hard samples
  - The EMA training can also alleviate the imbalanced pseudo-labeling biased issue due to the conservative property of the EMA training

$$\theta_t^i = \hat{\theta} - \gamma \sum_{k=1}^{i-1} (1 - \alpha^{-k+(i-1)}) \frac{\partial(\mathcal{L}_{sup} + \boldsymbol{\lambda}_u \mathcal{L}_{unsup})}{\partial \theta_s^k}, \tag{4}$$

3. Code=f(method)

- 以Train the Unbiased Teacher under 1% COCO-supervision为例：

```
python train_net.py \
      --num-gpus 8 \
      --config configs/coco_supervision/faster_rcnn_R_50_FPN_sup1_run1.yaml \
       SOLVER.IMG_PER_BATCH_LABEL 16 SOLVER.IMG_PER_BATCH_UNLABEL 16
```

- 以configs/coco_supervision/faster_rcnn_R_50_FPN_sup1_run1.yaml为例：

```
_BASE_: "../Base-RCNN-FPN.yaml"
MODEL:
  META_ARCHITECTURE: "TwoStagePseudoLabGeneralizedRCNN"
  WEIGHTS: "detectron2://ImageNetPretrained/MSRA/R-50.pkl"
  MASK_ON: False
  RESNETS:
    DEPTH: 50
  PROPOSAL_GENERATOR:
    NAME: "PseudoLabRPN"
  RPN:
    POSITIVE_FRACTION: 0.25
    LOSS: "CrossEntropy"
  ROI_HEADS:
    NAME: "StandardROIHeadsPseudoLab"
    LOSS: "FocalLoss"
SOLVER:
  LR_SCHEDULER_NAME: "WarmupMultiStepLR"
  STEPS: (179990, 179995)
  MAX_ITER: 180000  ##论文第16页的A4中的Training部分有说明
  IMG_PER_BATCH_LABEL: 32  ##论文第16页的A4中的Training部分有说明
  IMG_PER_BATCH_UNLABEL: 32  ##论文第16页的A4中的Training部分有说明
  BASE_LR: 0.01  ##论文第16页的A4中的Training部分有说明
DATALOADER:
  SUP_PERCENT: 1.0  ##训练数据中label数据的比例为1%
  RANDOM_DATA_SEED: 1
DATASETS:
  CROSS_DATASET: False
  TRAIN: ("coco_2017_train",)
  TEST: ("coco_2017_val",)
SEMISUPNET:
  Trainer: "ubteacher"
  BBOX_THRESHOLD: 0.7  ##论文第16页的A4中的Hyper-parameters部分有说明
  TEACHER_UPDATE_ITER: 1  ##Teacher模型的更新频率
  BURN_UP_STEP: 2000  ##论文第16页的A4中的Training部分有说明
  EMA_KEEP_RATE: 0.9996  ##论文第16页的A4中的Hyper-parameters部分有说明
  UNSUP_LOSS_WEIGHT: 4.0  ##论文第16页的A4中的Hyper-parameters部分有说明
TEST:
  EVAL_PERIOD: 1000
  EVALUATOR: "COCOeval"
```

3.1 Input

/ubteacher/engine/trainer.py

```
# data_q and data_k from different augmentations (q:strong, k:weak)
# labeled_strong, labeled_weak, unlabeled_strong, unlabeled_weak
label_data_q, label_data_k, unlabel_data_q, unlabel_data_k = data
```

/ubteacher/data/build.py

```
# used by unbiased teacher trainer
def build_detection_semisup_train_loader_two_crops(cfg, mapper=None):
    if cfg.DATASETS.CROSS_DATASET:  # cross-dataset (e.g., coco-additional)
        label_dicts = get_detection_dataset_dicts(
            cfg.DATASETS.TRAIN_LABEL,
            filter_empty=cfg.DATALOADER.FILTER_EMPTY_ANNOTATIONS,
            min_keypoints=cfg.MODEL.ROI_KEYPOINT_HEAD.MIN_KEYPOINTS_PER_IMAGE
            if cfg.MODEL.KEYPOINT_ON
            else 0,
            proposal_files=cfg.DATASETS.PROPOSAL_FILES_TRAIN
            if cfg.MODEL.LOAD_PROPOSALS
            else None,
        )
        unlabel_dicts = get_detection_dataset_dicts(
            cfg.DATASETS.TRAIN_UNLABEL,
            filter_empty=False,
            min_keypoints=cfg.MODEL.ROI_KEYPOINT_HEAD.MIN_KEYPOINTS_PER_IMAGE
            if cfg.MODEL.KEYPOINT_ON
            else 0,
            proposal_files=cfg.DATASETS.PROPOSAL_FILES_TRAIN
            if cfg.MODEL.LOAD_PROPOSALS
            else None,
        )
    else:  # different degree of supervision (e.g., COCO-supervision)
        dataset_dicts = get_detection_dataset_dicts(
            cfg.DATASETS.TRAIN,
            filter_empty=cfg.DATALOADER.FILTER_EMPTY_ANNOTATIONS,
            min_keypoints=cfg.MODEL.ROI_KEYPOINT_HEAD.MIN_KEYPOINTS_PER_IMAGE
            if cfg.MODEL.KEYPOINT_ON
            else 0,
            proposal_files=cfg.DATASETS.PROPOSAL_FILES_TRAIN
            if cfg.MODEL.LOAD_PROPOSALS
            else None,
        )

        ##根据训练数据中label数据的比例划分数据集
        # Divide into labeled and unlabeled sets according to supervision
percentage
        label_dicts, unlabel_dicts = divide_label_unlabel(
            dataset_dicts,
            cfg.DATALOADER.SUP_PERCENT,
            cfg.DATALOADER.RANDOM_DATA_SEED,
            cfg.DATALOADER.RANDOM_DATA_SEED_PATH,
        )

    label_dataset = DatasetFromList(label_dicts, copy=False)
```

```
    # exclude the labeled set from unlabeled dataset
    unlabel_dataset = DatasetFromList(unlabel_dicts, copy=False)
    # include the labeled set in unlabel dataset
    # unlabel_dataset = DatasetFromList(dataset_dicts, copy=False)

    ##对数据集进行强、弱增广
    if mapper is None:
        mapper = DatasetMapper(cfg, True)
    label_dataset = MapDataset(label_dataset, mapper)
    unlabel_dataset = MapDataset(unlabel_dataset, mapper)

    sampler_name = cfg.DATALOADER.SAMPLER_TRAIN
    logger = logging.getLogger(__name__)
    logger.info("Using training sampler {}".format(sampler_name))
    if sampler_name == "TrainingSampler":
        label_sampler = TrainingSampler(len(label_dataset))
        unlabel_sampler = TrainingSampler(len(unlabel_dataset))
    elif sampler_name == "RepeatFactorTrainingSampler":
        raise NotImplementedError("{} not yet supported.".format(sampler_name))
    else:
        raise ValueError("Unknown training sampler: {}".format(sampler_name))
    return build_semisup_batch_data_loader_two_crop(
        (label_dataset, unlabel_dataset),
        (label_sampler, unlabel_sampler),
        cfg.SOLVER.IMG_PER_BATCH_LABEL,  ##训练过程中每个batch中的label数据量(论文中为
32)
        cfg.SOLVER.IMG_PER_BATCH_UNLABEL,  ##训练过程中每个batch中的unlabel数据量(论
文中为32)
        aspect_ratio_grouping=cfg.DATALOADER.ASPECT_RATIO_GROUPING,
        num_workers=cfg.DATALOADER.NUM_WORKERS,
    )
```

/ubteacher/data/build.py

```
def divide_label_unlabel(
    dataset_dicts, SupPercent, random_data_seed, random_data_seed_path
):
    num_all = len(dataset_dicts)
    num_label = int(SupPercent / 100.0 * num_all)

    # read from pre-generated data seed
    with open(random_data_seed_path) as COCO_sup_file:
        coco_random_idx = json.load(COCO_sup_file)

    labeled_idx = np.array(coco_random_idx[str(SupPercent)]
[str(random_data_seed)])
    assert labeled_idx.shape[0] == num_label, "Number of READ_DATA is
mismatched."

    label_dicts = []
    unlabel_dicts = []
    labeled_idx = set(labeled_idx)

    for i in range(len(dataset_dicts)):
        if i in labeled_idx:
            label_dicts.append(dataset_dicts[i])
        else:
```

```
            unlabel_dicts.append(dataset_dicts[i])

    return label_dicts, unlabel_dicts
```

/ubteacher/data/dataset_mapper.py

```
# apply strong augmentation
# We use torchvision augmentation, which is not compatiable with
# detectron2, which use numpy format for images. Thus, we need to
# convert to PIL format first.
image_pil = Image.fromarray(image_weak_aug.astype("uint8"), "RGB")
image_strong_aug = np.array(self.strong_augmentation(image_pil))
dataset_dict["image"] = torch.as_tensor(
np.ascontiguousarray(image_strong_aug.transpose(2, 0, 1))
)

dataset_dict_key = copy.deepcopy(dataset_dict)
dataset_dict_key["image"] = torch.as_tensor(
np.ascontiguousarray(image_weak_aug.transpose(2, 0, 1))
)
assert dataset_dict["image"].size(1) == dataset_dict_key["image"].size(1)
assert dataset_dict["image"].size(2) == dataset_dict_key["image"].size(2)
return (dataset_dict, dataset_dict_key)
```

3.2 Process

3.2.1 Initialize the detector

/ubteacher/engine/trainer.py

```
##进行预热阶段的模型训练
# burn-in stage (supervised training with labeled data)
if self.iter < self.cfg.SEMISUPNET.BURN_UP_STEP:

    # input both strong and weak supervised data into model
    label_data_q.extend(label_data_k)
    ##这里的record_dict是losses = {}，包含RPN的proposal_losses和ROIHeads的
detector_losses
    record_dict, _, _, _ = self.model(label_data_q, branch="supervised")  ##见
ubteacher/modeling/meta_arch/rcnn.py

    # weight losses
    loss_dict = {}
    for key in record_dict.keys():
        if key[:4] == "loss":
            loss_dict[key] = record_dict[key] * 1
    losses = sum(loss_dict.values())
```

3.2.2 Teacher-Student Mutual Learning

/ubteacher/engine/trainer.py

- **Student Learning**:

```
##进行相互学习阶段的模型训练
else:
    ##预测阶段训练结束，将得到的初始模型复制给老师模型
    if self.iter == self.cfg.SEMISUPNET.BURN_UP_STEP:
        # update copy the the whole model
        self._update_teacher_model(keep_rate=0.00)
    ##相互学习阶段，按照一定的迭代频率更新老师模型
    elif (
        self.iter - self.cfg.SEMISUPNET.BURN_UP_STEP
    ) % self.cfg.SEMISUPNET.TEACHER_UPDATE_ITER == 0:
        self._update_teacher_model(keep_rate=self.cfg.SEMISUPNET.EMA_KEEP_RATE)

    record_dict = {}
    # generate the pseudo-label using teacher model
    # note that we do not convert to eval mode, as 1) there is no gradient
computed in
    # teacher model and 2) batch norm layers are not updated as well
    ##老师模型不进行梯度计算和反向传播，因为噪声伪标签会影响老师模型；论文公式(2)上面有说明
    with torch.no_grad():
        ##老师模型用弱增广数据生成伪标签(RPN和ROIHead)
        (
            _,
            proposals_rpn_unsup_k,
            proposals_roih_unsup_k,
            _,
        ) = self.model_teacher(unlabel_data_k, branch="unsup_data_weak")  ##见
ubteacher/modeling/meta_arch/rcnn.py

    #  Pseudo-labeling
    cur_threshold = self.cfg.SEMISUPNET.BBOX_THRESHOLD  ##文中置信度阈值为0.7，用于过
滤低置信度的bbox

    joint_proposal_dict = {}
    joint_proposal_dict["proposals_rpn"] = proposals_rpn_unsup_k
    (
        pesudo_proposals_rpn_unsup_k,
        nun_pseudo_bbox_rpn,
    ) = self.process_pseudo_label(
        proposals_rpn_unsup_k, cur_threshold, "rpn", "thresholding"
    )
    joint_proposal_dict["proposals_pseudo_rpn"] = pesudo_proposals_rpn_unsup_k
    # Pseudo_labeling for ROI head (bbox location/objectness)
    pesudo_proposals_roih_unsup_k, _ = self.process_pseudo_label(
        proposals_roih_unsup_k, cur_threshold, "roih", "thresholding"
    )
    joint_proposal_dict["proposals_pseudo_roih"] = pesudo_proposals_roih_unsup_k

    #  add pseudo-label to unlabeled data
    unlabel_data_q = self.remove_label(unlabel_data_q)
    unlabel_data_k = self.remove_label(unlabel_data_k)

    unlabel_data_q = self.add_label(
```

```
        unlabel_data_q, joint_proposal_dict["proposals_pseudo_roih"]
    )
    unlabel_data_k = self.add_label(
        unlabel_data_k, joint_proposal_dict["proposals_pseudo_roih"]
    )

    all_label_data = label_data_q + label_data_k
    all_unlabel_data = unlabel_data_q

    ##这里的record_all_label_data是losses = {}，包含RPN的proposal_losses和ROIHeads的
detector_losses
    record_all_label_data, _, _, _ = self.model(
        all_label_data, branch="supervised"
    )  ##见ubteacher/modeling/meta_arch/rcnn.py
    record_dict.update(record_all_label_data)
    ##这里的record_all_unlabel_data是losses = {}，包含RPN的proposal_losses和ROIHeads
的detector_losses
    record_all_unlabel_data, _, _, _ = self.model(
        all_unlabel_data, branch="supervised"
    )  ##见ubteacher/modeling/meta_arch/rcnn.py
    new_record_all_unlabel_data = {}
    for key in record_all_unlabel_data.keys():
        ##给训练过程中伪标签数据产生的loss加上标签"_pseudo"
        new_record_all_unlabel_data[key + "_pseudo"] =
record_all_unlabel_data[key]
    record_dict.update(new_record_all_unlabel_data)
    ##此时的record_dict即包含label数据的loss，也包含pseudo-label数据的loss

    # weight losses
    loss_dict = {}
    for key in record_dict.keys():
        if key[:4] == "loss":
            if key == "loss_rpn_loc_pseudo" or key == "loss_box_reg_pseudo":
                # pseudo bbox regression <- 0
                loss_dict[key] = record_dict[key] * 0  ##论文公式(2)下面有说明，对
bbox回归可能不正确的伪标签无法被置信度阈值过滤，因此非监督loss中不加bbox
            elif key[-6:] == "pseudo":  # unsupervised loss
                loss_dict[key] = (
                    record_dict[key] * self.cfg.SEMISUPNET.UNSUP_LOSS_WEIGHT  ##
对应论文公式(2)
                )
            else:  # supervised loss
                loss_dict[key] = record_dict[key] * 1

    losses = sum(loss_dict.values())
```

- **Teacher Refinement**:

```
@torch.no_grad()
def _update_teacher_model(self, keep_rate=0.996):
    if comm.get_world_size() > 1:
        student_model_dict = {
            key[7:]: value for key, value in self.model.state_dict().items()
        }
    else:
```

```
        student_model_dict = self.model.state_dict()

    new_teacher_dict = OrderedDict()
    for key, value in self.model_teacher.state_dict().items():
        if key in student_model_dict.keys():
            new_teacher_dict[key] = (
                student_model_dict[key] * (1 - keep_rate) + value * keep_rate
##对应论文公式(3)
            )
        else:
            raise Exception("{} is not found in student model".format(key))

    self.model_teacher.load_state_dict(new_teacher_dict)
```

3.3 Output

- train_net.py

```
if args.eval_only:
    if cfg.SEMISUPNET.Trainer == "ubteacher":
        model = Trainer.build_model(cfg)
        model_teacher = Trainer.build_model(cfg)
        ensem_ts_model = EnsembleTSModel(model_teacher, model)

        DetectionCheckpointer(
            ensem_ts_model, save_dir=cfg.OUTPUT_DIR
        ).resume_or_load(cfg.MODEL.WEIGHTS, resume=args.resume)
        res = Trainer.test(cfg, ensem_ts_model.modelTeacher)  ##用老师模型进行推理

    else:
        model = Trainer.build_model(cfg)
        DetectionCheckpointer(model, save_dir=cfg.OUTPUT_DIR).resume_or_load(
            cfg.MODEL.WEIGHTS, resume=args.resume
        )
        res = Trainer.test(cfg, model)
    return res
```

参考资料

1. 半监督目标检测SOTA: Unbiased Teacher(教学相长)
2. Unbiased Teacher for Semi-Supervised Object Detection论文解读

更新于2021-05-08