mAP 평가지표

2022. 5. 18

충북대학교 산업인공지능학과
[21-8조] 이지호, 이충현

mAP가 무엇인지 알아내기 위한 과정

1. IOU (Intersection over Union)
2. Precision & Recall
3. AP (Average Precision)
4. mAP (mean Average Precision)

IOU (Intersection over Union)

- IOU란 Intersection over Union의 약자로 객체 검출을 평가하기 위한 지표로 두개의 박스의 겹침 정도를 계산하여 얼마나 실제 값을 예측하였는지 나타냅니다.

- IOU_{50}의 경우 IOU 값이 50%이상인 결과만 검출에 성공한 것으로 평가 합니다.

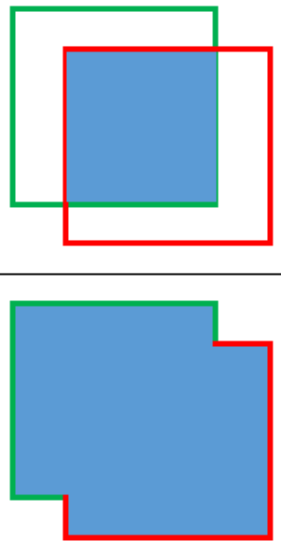
- 예측 값과 실제 값의 교집합을 합집합으로 나누어 계산 합니다.

B_p = 실제 (*Gound Truth*)

B_{gt} = 예측 (*Prediction*)

$$IOU = \frac{area\ of\ overlap}{area\ of\ union} = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})}$$

$$= \frac{\text{실제} \cap \text{예측 중복 영역}}{\text{실제} \cup \text{예측 전체 영역}}$$



waytoliah.com

IOU 코드

```python
def iou(bbox1: list, bbox2: list) -> float:
    cx1, cy1, w1, h1 = bbox1
    cx2, cy2, w2, h2 = bbox2

    # 바운딩박스 넓이
    bbox1_area = w1 * h1
    bbox2_area = w2 * h2

    # 교집합
    intersection_x1 = max(cx1 - (w1 / 2), cx2 - (w2 / 2))
    intersection_y1 = max(cy1 - (h1 / 2), cy2 - (h2 / 2))
    intersection_x2 = min(cx1 + (w1 / 2), cx2 + (w2 / 2))
    intersection_y2 = min(cy1 + (h1 / 2), cy2 + (h2 / 2))
    intersection_w = max(0, intersection_x2 - intersection_x1)
    intersection_h = max(0, intersection_y2 - intersection_y1)

    # 교집합, 합집합 넓이
    intersection_area = intersection_w * intersection_h
    union_area = bbox1_area + bbox2_area - intersection_area

    return intersection_area / union_area

```

Precision & Recall (정밀도 & 재현률)

Coufusion Matrix		**예측 결과 (Predict result)**	
		Positive	Negative
정답 (Ground truth)	Positive	TP (True Positive) 정상 예측	FN (False Negative) 예측 실패 (미 탐지)
	Negative	FP (False Positive) 잘못된 예측 (오 탐지)	TN (True Negative) 정상 미예측

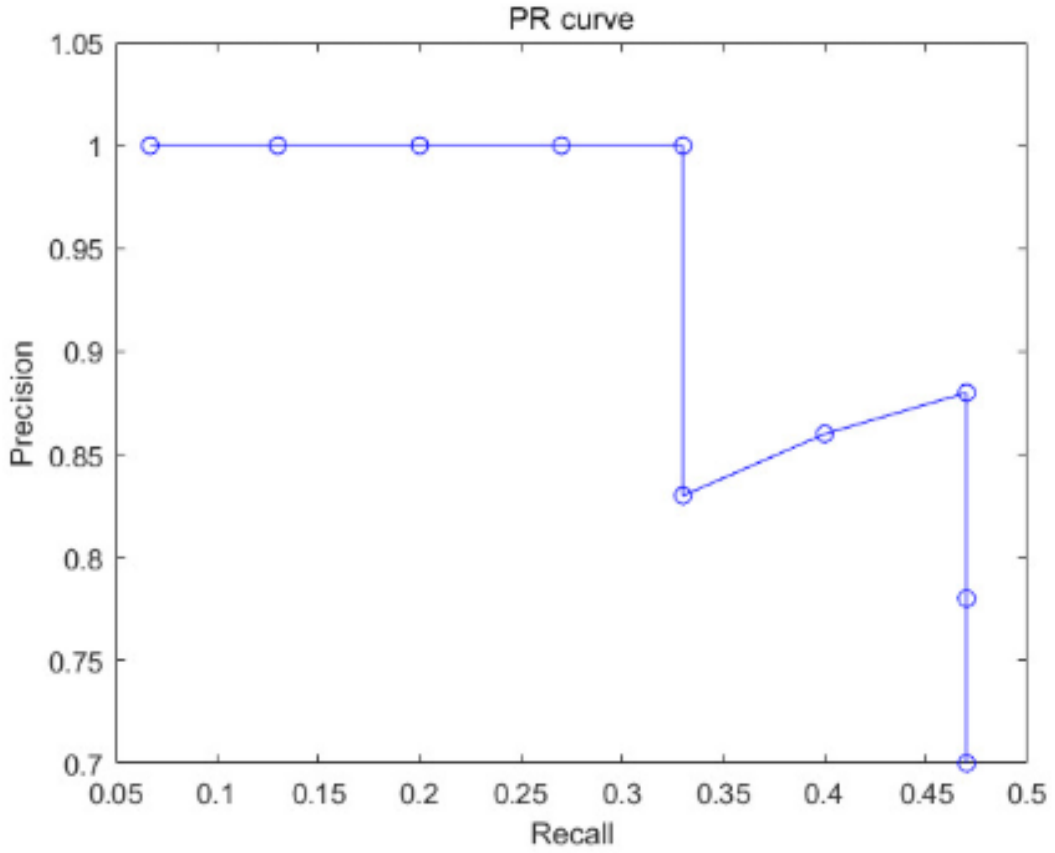
$$Precision = \frac{TP}{TP + FP} = \frac{TP}{Detection\ result}$$

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{Ground\ truth}$$

PR 곡선

- 이전 페이지에서 구한 Precision & Recall 을 이용하여 그래프를 그리게 됩니다.

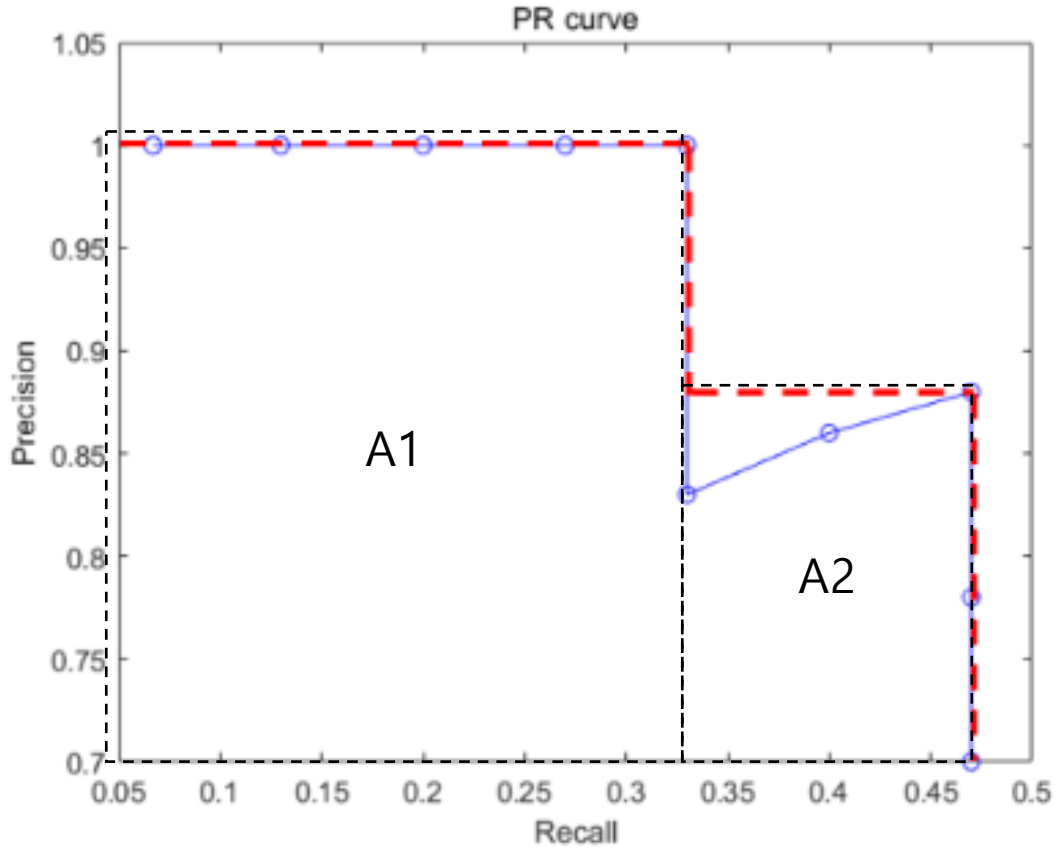
- PR곡선을 그릴 때 Confidence값에 대한 Threshold값을 조절 하면 Precision, Recall 도 그에 따라 변하게 되는데 이것을 그리게 됩니다.



AP (Average Precision)

- AP는 Precision-Recall 그래프 에서 그래프 면적을 뜻합니다.
- 일반적으로 계산전에 PR곡선을 단조적으로 감소하는 그래프가 되게 하기 위해 PR 곡선을 보정해 줍니다.
- 오른쪽 예시 그래프 경우 아래와 같습니다.

$$AP = A1 + A2$$
$$= 1 \times 0.33 + 0.88 \times (0.47 - 0.33)$$
$$= 0.4532$$



AP 예제

rec: recall
prec: precision

mrec: recall 리스트
mpre: precision 리스트

[출처]
rafaelpadilla/**review_object_detection_metrics** **(Github)**

[URL]
https://github.com/rafaelpadilla/review_object_detection_metrics/blob/56d8969739d4774b4bab5b8122870e7e4c833021/src/evaluators/pascal_voc_evaluator.py#L13-L31

```python
def calculate_ap_every_point(rec, prec):
    mrec = []
    mrec.append(0)
    [mrec.append(e) for e in rec]
    mrec.append(1)
    mpre = []
    mpre.append(0)
    [mpre.append(e) for e in prec]
    mpre.append(0)
    for i in range(len(mpre) - 1, 0, -1):
        mpre[i - 1] = max(mpre[i - 1], mpre[i])
    ii = []
    for i in range(len(mrec) - 1):
        if mrec[1:][i] != mrec[0:-1][i]:
            ii.append(i + 1)
    ap = 0
    for i in ii:
        ap = ap + np.sum((mrec[i] - mrec[i - 1]) * mpre[i])
    return [ap, mpre[0:len(mpre) - 1], mrec[0:len(mpre) - 1], ii]
```

mAP (mean Average Precision)

- 1개의 클래스당 1개의 AP 값을 구하고, 여러 클래스에 대해서 mean 값을 구한 것입니다.

$$mAP = \frac{1}{cls}\sum_{1}^{cls} AP(cls)$$

mAP 예제

gt: Ground Truth (실제 값)
preds: Predictions (예측 값)
cls: class (부류)
vid: video (영상)

rec: recall
prec: precision

[출처]
rafaelpadilla/**review_object_detection_metrics** **(Github)**

[URL]
https://github.com/rafaelpadilla/review_object_detection_metrics/blob/main/src/evaluators/tube_evaluator.py#L63-L130

```python
def evaluate(self, thr=0.5):
    """Evaluate the predictions according to the chosen IOU threshold
    Args:
        thr (float, optional): IOU threshold 0 < thr < 1. Defaults to 0.5.
    Returns:
        res, mAP: return a dictionary (res) with results per class. Also, returns the mAP.
    """
    if not 0 < thr <= 1:
        raise ValueError("IOU threshold must be 0 < thr <= 1: ", thr)

    self.__process()

    # loop over classes
    # TODO: group detection on videos and classes basis to avoid loop multiple times
    for obj_cls in self._classes:
        gt_cls = [gt for gt in self._gt if gt.category_id == obj_cls['id']]
        preds_cls = [pred for pred in self._predictions if pred.category_id == obj_cls['id']]

        # sort detections by decreasing confidence
        preds_cls = sorted(preds_cls, key=lambda tube: tube.confidence, reverse=True)

        # initialize true positive as zeros of length of predictions of a giving class

        # loop over videos
        for vid_id in self._videos:
            gts = [gt for gt in gt_cls if gt.video_id == vid_id['id']]
            preds = [pred for pred in preds_cls if pred.video_id == vid_id['id']]

            n_tp, n_fp, n_fn = self._classify_tubes(preds, gts, thr)

        TP = np.array([int(tube.isTP) for tube in preds_cls])
        FP = np.logical_not(TP).astype(int)

        # compute precision, recall and average precision
        acc_TP = np.cumsum(TP)
        acc_FP = np.cumsum(FP)
        rec = acc_TP / len(gt_cls)
        prec = np.divide(acc_TP, (acc_FP + acc_TP))

        # Depending on the method, call the right implementation
        if self._method == MethodAveragePrecision.EVERY_POINT_INTERPOLATION:
            [ap, mpre, mrec, ii] = calculate_ap_every_point(rec, prec)
        elif self._method == MethodAveragePrecision.ELEVEN_POINT_INTERPOLATION:
            [ap, mpre, mrec, _] = calculate_ap_11_point_interp(rec, prec)
        else:
            raise ValueError(f'Invalid interpolation method: {self._method}')

        # add class result in the dictionary to be returned
        self._res[obj_cls['name']] = {
            'precision': prec,
            'recall': rec,
            'AP': ap,
            'interpolated precision': mpre,
            'interpolated recall': mrec,
            'total TP': n_tp,
            'total FP': n_fp,
            'total FN': n_fn,
        }
    # For mAP, only the classes in the gt set should be considered
    mAP = 0.0
    for c, r in self._res.items():
        if any(cat['name'] == c for cat in self._classes):
            mAP += r['AP']
    mAP /= len(self._classes)

    return self._res, mAP
```