# DataLab Cup 2: Object Detection

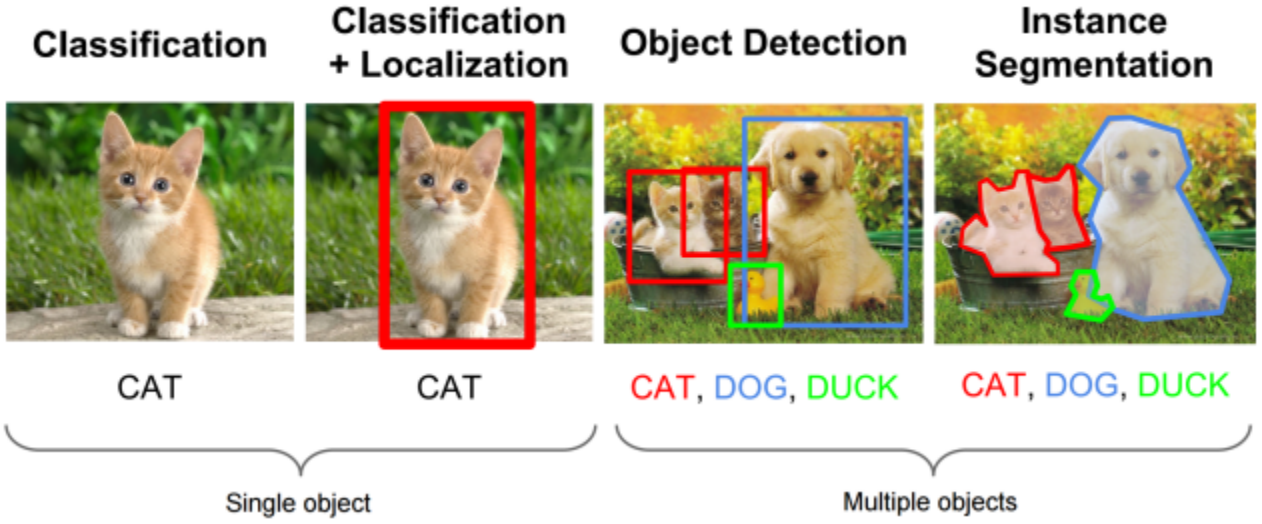
Bing-Han Chiang & Datalab

# Outline

- Competition Information
- Object detection model
  - You Only Look Once (YOLO)
- Evaluation metric
  - Mean Average Precision (mAP)
- Hints
- Precautions
- Competition Timeline

# Competition Information

- Object Detection
  - In this competition, we are going to train an object detection model to detect objects in an image.



# Competition Information

- Dataset
  - PASCAL VOC 2007
    - Train/Val data: 5011
      - Each row contains one image and its bounding boxes.
      - filename, $(x_{min}, y_{min}, x_{max}, y_{max}, \text{label}) * \text{object\_num}$

```
000012.jpg 156 97 351 270 6
000016.jpg 92 72 305 473 1
000017.jpg 185 62 279 199 14 90 78 403 336 12
000019.jpg 231 88 483 256 7 11 113 266 259 7
```

    - Test data: 4952
      - filename

```
000001.jpg
000002.jpg
000003.jpg
```

# Object Detection Model

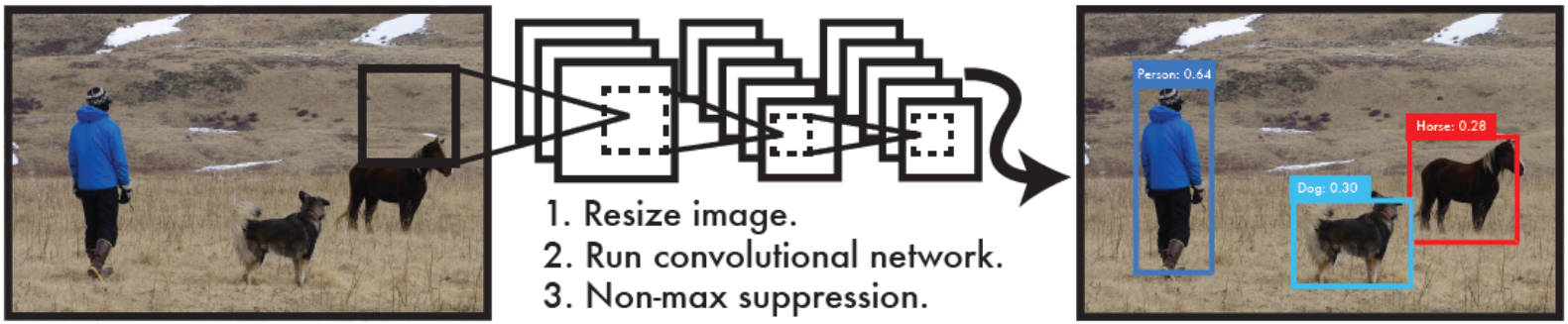
YOLO

- Motivation
  - Conventional object detection methods require two steps:
    1. Propose some regions that might contain object
    2. Doing classification on all proposed regions
  - It is inefficient to scan the image twice and costly to do classification on all proposed regions.
  - What about scanning image once and propose bounding boxes and labels accordingly?

# Object Detection Model

YOLO

- Main idea
  - Reframe object detection problem as a single regression problem.
  - Predict bounding box coordinates and class probabilities from image pixels straightly.



# Object Detection Model

YOLO

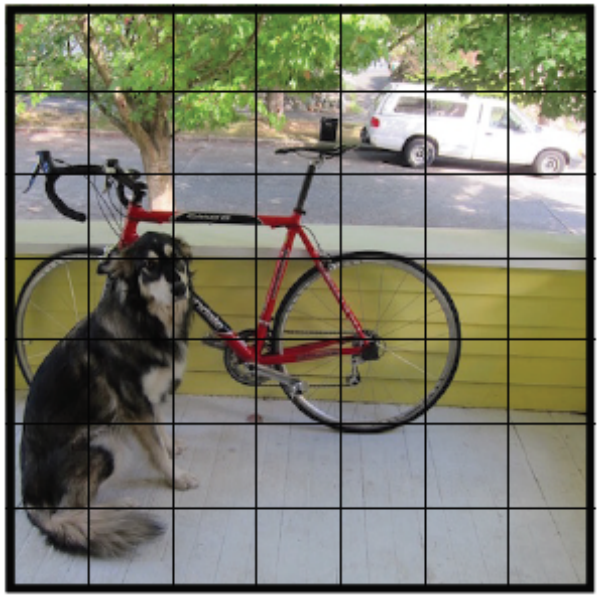
- Main idea
  - YOLO is extremely fast since it requires no complex pipelines to output bounding boxes and object labels.

| Real-Time Detectors | Train | mAP | FPS |
|---|---|---|---|
| 100Hz DPM [30] | 2007 | 16.0 | 100 |
| 30Hz DPM [30] | 2007 | 26.1 | 30 |
| Fast YOLO | 2007+2012 | 52.7 | **155** |
| YOLO | 2007+2012 | **63.4** | 45 |
| Less Than Real-Time | | | |
| Fastest DPM [37] | 2007 | 30.4 | 15 |
| R-CNN Minus R [20] | 2007 | 53.5 | 6 |
| Fast R-CNN [14] | 2007+2012 | 70.0 | 0.5 |
| Faster R-CNN VGG-16[27] | 2007+2012 | 73.2 | 7 |
| Faster R-CNN ZF [27] | 2007+2012 | 62.1 | 18 |

# Object Detection Model

YOLO

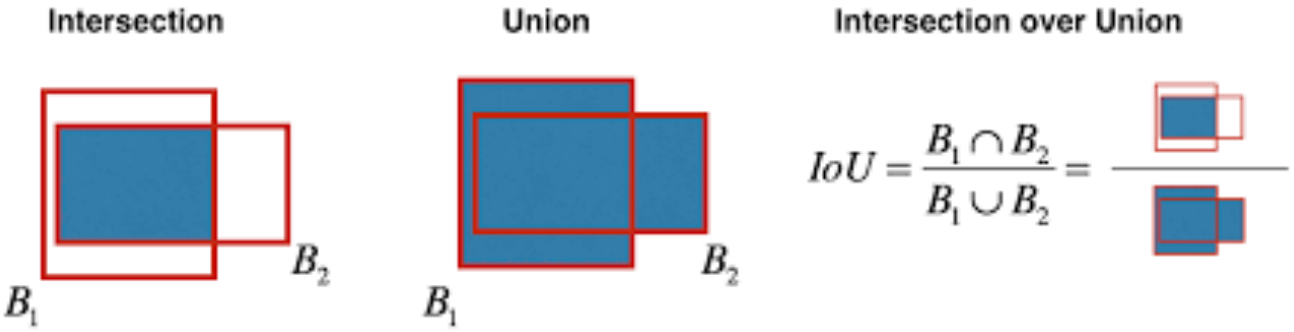
- Objective function
  - YOLO first divides the input image into a $S \times S$ grid.
  - If the center of an object falls into a grid cell, the grid cell is responsible for detecting that object.



# Object Detection Model

YOLO

- *Intersection over Union* (IoU)
  - A metric to evaluate the effectiveness of predict bounding box comparing to the ground truth.



# Object Detection Model

YOLO

- Objective function
  - Each grid cell predicts $B$ bounding boxes and confidence score for those boxes.
    - Each bounding box consists of 5 predictions: $x, y, w, h$ and confidence.

# Object Detection Model

YOLO

- Objective function
  - Each grid cell predicts $B$ bounding boxes and confidence score for those boxes.
    - The confidence score is defined as

$$\Pr(\text{Object}) * \text{IoU}_{pred}^{truth}.$$

    - The score should be zero when there is no object in bounding box.
    - Otherwise, the score should be equal to $\text{IoU}_{pred}^{truth}$.

# Object Detection Model

YOLO

- Objective function
  - Each grid cell also predicts $C$ conditional class probabilities,

$$\Pr(\text{Class}_i|\text{Object})$$

  - Multiplying the conditional class probabilities and the individual box confidence predictions gives us the *class-specific scores for each box*.

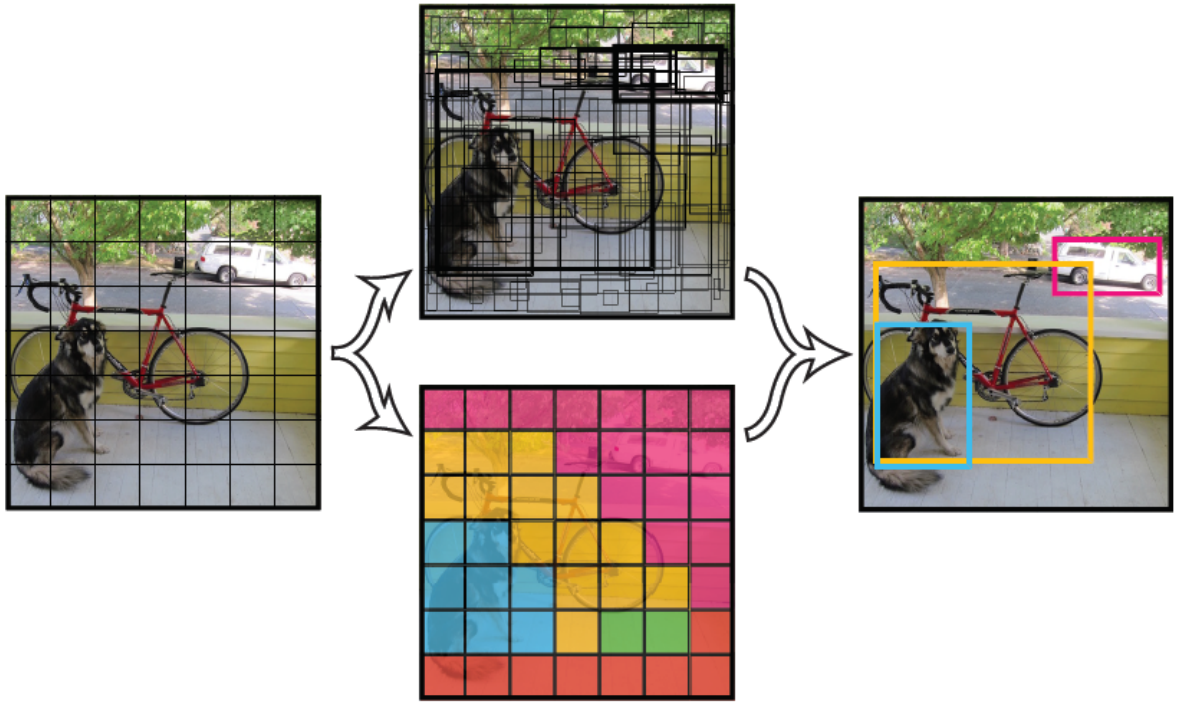
$$\Pr(\text{Class}_i|\text{Object}) * \Pr(\text{Object}) * \text{IoU}_{pred}^{truth} = \Pr(\text{Class}_i) * \text{IoU}_{pred}^{truth}$$

Conditional Class Prob. Bounding box confidence

# Object Detection Model

YOLO

- Objective function
  - The predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.



# Object Detection Model

YOLO

- Objective function
  - During training, the objective function is formulated as below:

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(x_i - \hat{x}_i\right)^2 + \left(y_i - \hat{y}_i\right)^2$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(\sqrt{w_i} - \sqrt{\hat{w}_i}\right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i}\right)^2$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i\right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i\right)^2$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c)\right)^2$$

# Object Detection Model

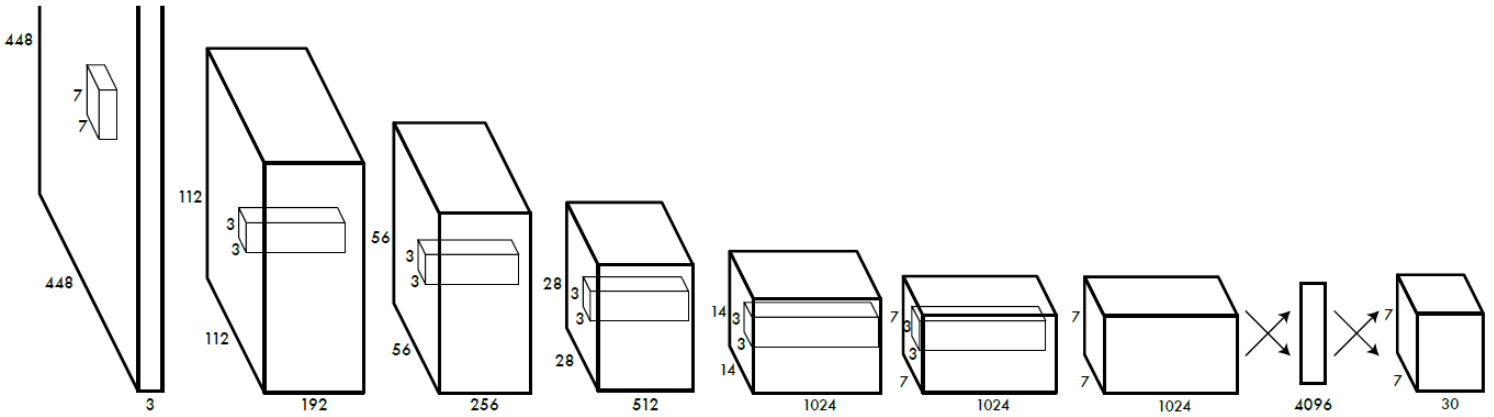
YOLO

- Objective function
  - In our implementation, we choose $S = 7, B = 2$ and $\mathrm{C} = 20$, because PASCAL VOC has 20 labels.
  - As a result, the final prediction is a $7 \times 7 \times 30$ tensor.

# Object Detection Model

YOLO

- Model architecture
  - 24 convolutional layers followed by 2 fully connected layers.



# Object Detection Model

YOLO

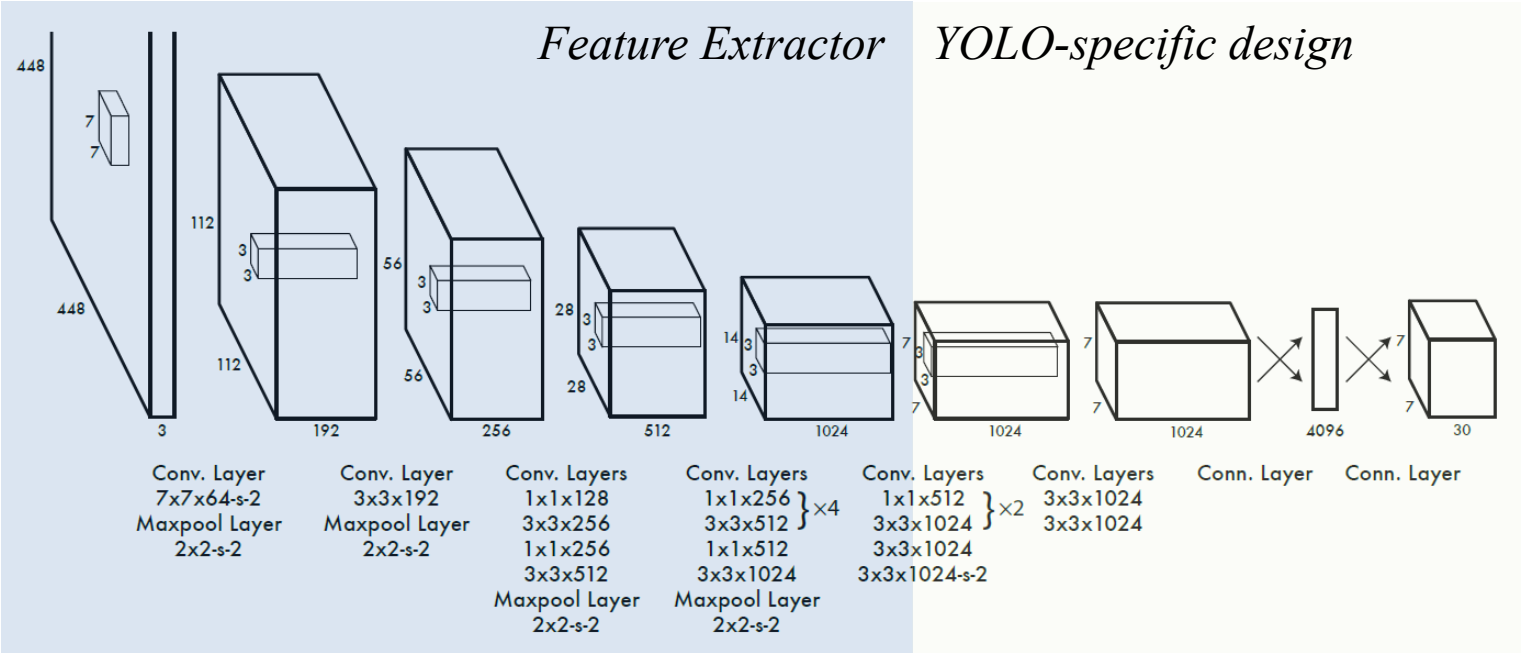
- Model architecture
  - YOLO uses the relu activation function for the final layer and all other layers use the following leaky rectified linear activation function:

$$\phi(x) = \begin{cases} x, & \text{if } x > 0 \\ 0.1x, & \text{otherwise} \end{cases}$$

# Object Detection Model

YOLO

- Training strategy
  - Pre-train the first 20 convolutional layers on ImageNet.
  - Train the whole network end-to-end on PASCAL.



# Object Detection Model

YOLO

- Reference
  - For more detailed explanation, please see the original paper: https://arxiv.org/abs/1506.02640

# Evaluation Metric

Mean Average Precision (mAP)

- Confusion matrix reminder
  - **True positive (TP)**: A correct detection. Detection with $\text{IoU} \geq threshold$.
  - **False positive (FP)** : A wrong detection. Detection with $\text{IoU} < threshold$.
  - **False Negative (FN)**: A ground truth not detected.
  - **True Negative (TN)**: A correct misdetection. Does not apply in evaluation.

# Evaluation Metric

Mean Average Precision (mAP)

- Precision x Recall curve
  - Precision: the percentage of correct positive predictions.

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{all\ detections}$$

  - Recall: the percentage of true positive detected among all ground truths.

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{all\ ground\ truths}$$

# Evaluation Metric

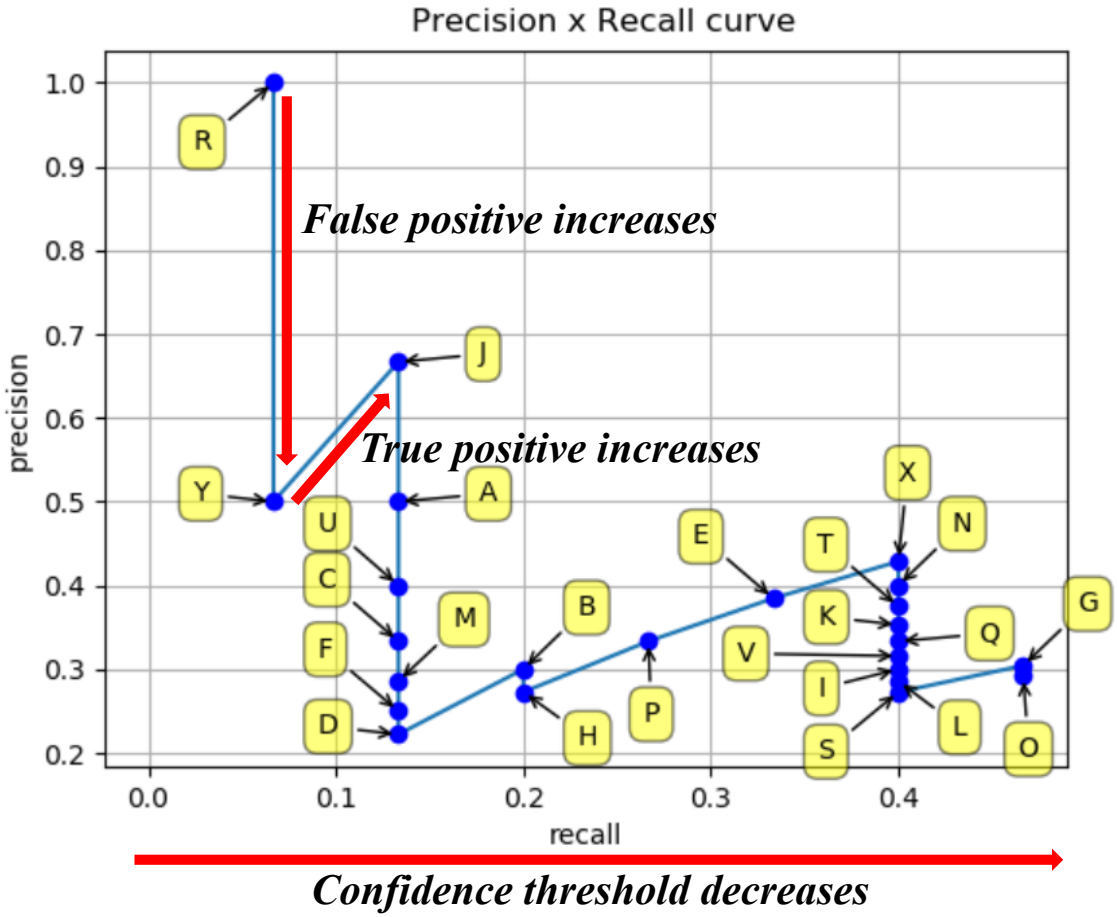
## Mean Average Precision (mAP)

- Precision x Recall curve
  - An object detector of a particular class is considered good if its precision stays high as recall increases.
  - It means that if you vary the confidence threshold, the precision and recall will still be high.

# Evaluation Metric

## Mean Average Precision (mAP)

- Precision x Recall curve



# Evaluation Metric

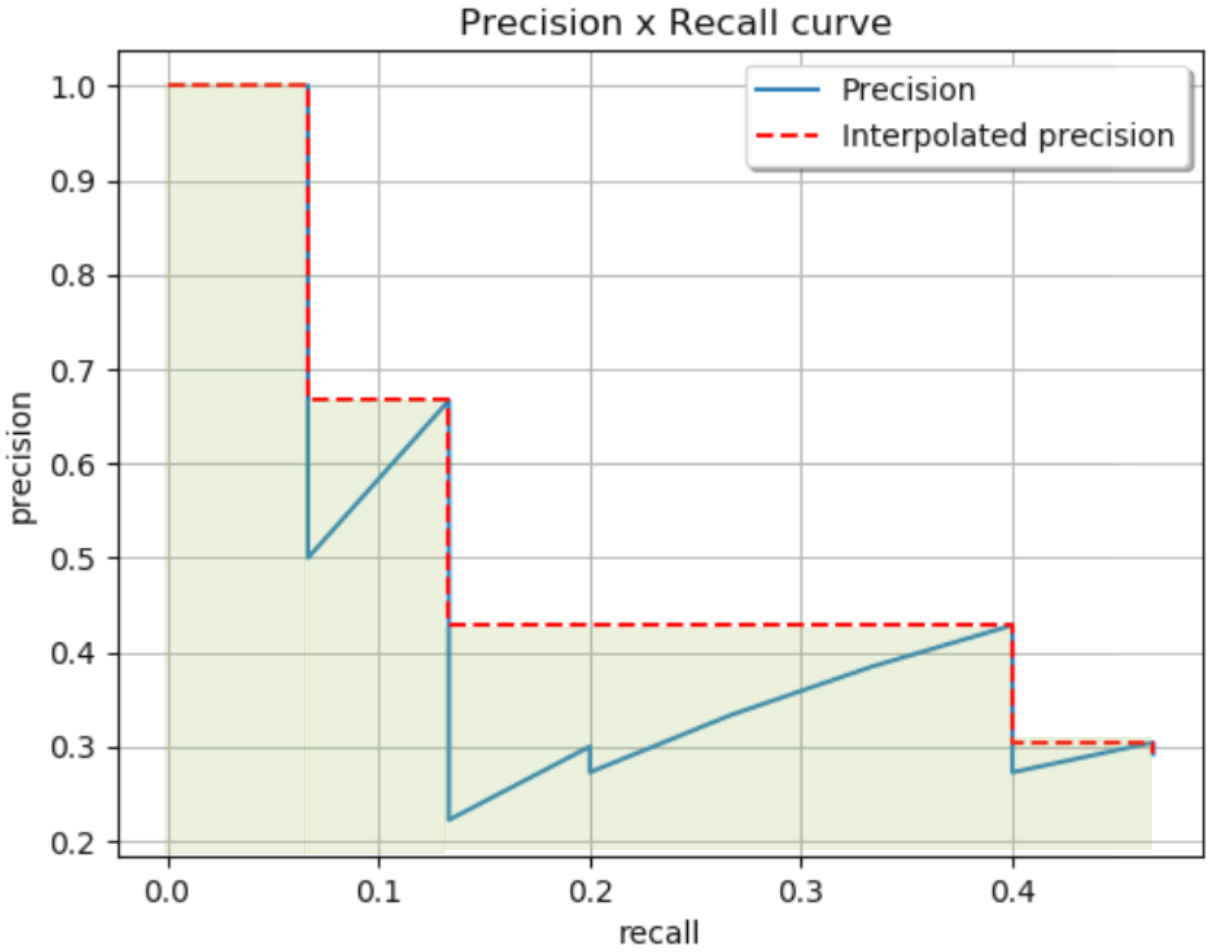
## Mean Average Precision (mAP)

- Average Precision (AP)
  - Represent precision at every recall by the maximum precision whose recall value is greater or equal to current recall and calculate the area under curve (AUC).

# Evaluation Metric

## Mean Average Precision (mAP)

- Average Precision (AP)



# Evaluation Metric

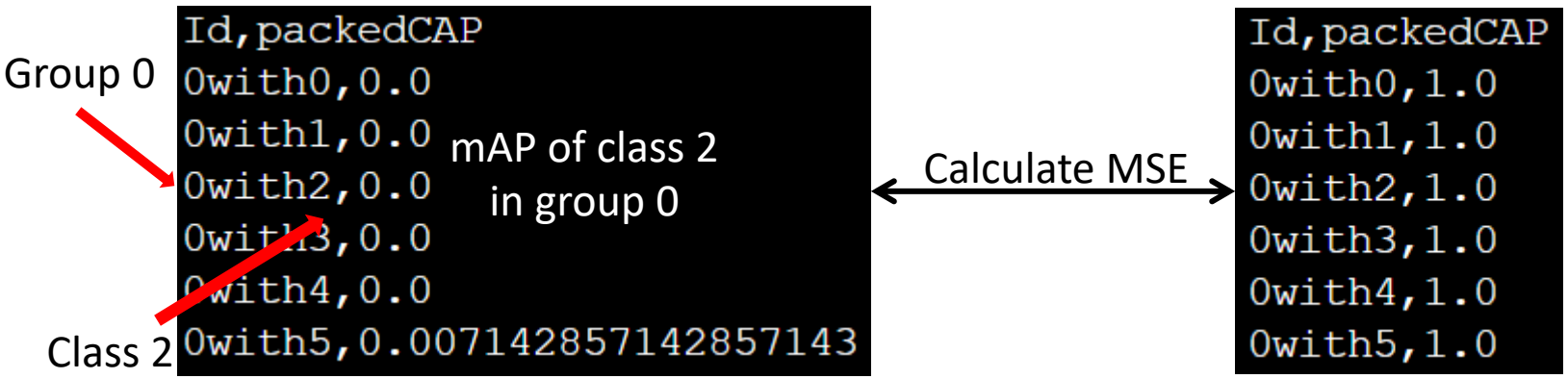
## Mean Average Precision (mAP)

- Mean Average Precision (mAP)
  - Calculate the Average Precision for every class and average them.

# Evaluation Metric

## Mean Average Precision (mAP)

- Mean Average Precision (mAP)
  - In this competition, we divide testing data into 10 groups and calculate the mAP of each class.
  - After deriving the mAP of each class in 10 groups, we compare the result with ground truth and use the mean square error as the final score.



# Evaluation Metric

## Mean Average Precision (mAP)

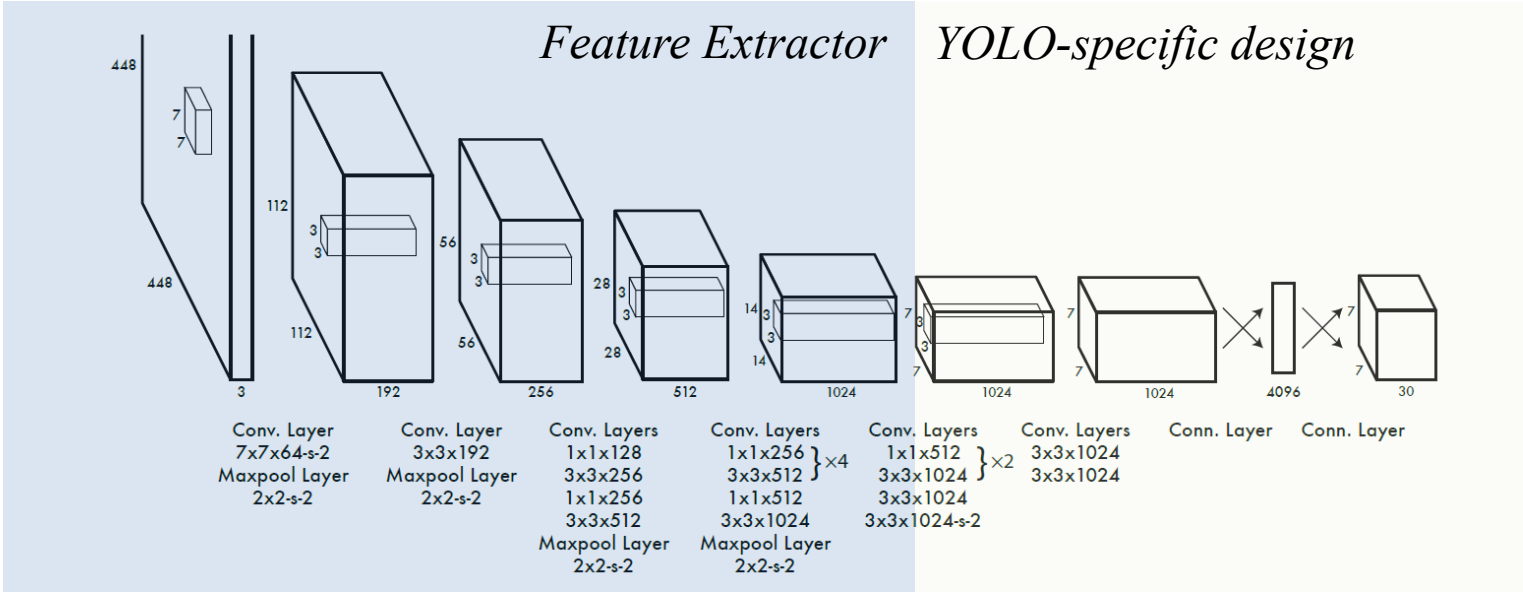
- Mean Average Precision (mAP)
  - For more detailed explanation of mAP, please see https://github.com/rafaelpadilla/Object-Detection-Metrics

# Hints

1. Transfer learning
2. Data augmentation
3. Training strategy
4. Other object detection models

# Hints

1. Transfer learning
   - Training from scratch is nearly impossible for object detection.
   - Feel free to replace the feature extractor with other pre-trained model.



# Hints

1. Transfer learning
   - YOLO pre-trained its feature extractor on ImageNet.
   - How to load pre-trained model is already described in lab12 - style transfer.
   - Be careful that different models require different data preprocess.
   - You can see all the pre-trained models provided by Keras here: https://www.tensorflow.org/api_docs/python/tf/keras/applications

# Hints

2. Data augmentation
   - The dataset we are using in this competition is the combination of training and validation set from VOC 2007.
   - It contains only 5012 images in total. Furthermore, the labels are highly imbalanced.
   - Doing data augmentation not only helps your model generalizing to testing data but also easing the training process.

# Hints

2. Data augmentation
   - Random scaling and translations are applied when training YOLO.
   - Note that the bounding box coordinates have to be changed accordingly if the image was transformed.

# Hints

3. Training strategy
   - It takes more than 2 days to converge when training on *RTX 2080 Ti*. Be patient.
   - Learning rate influence the convergence speed.
   - Shuffle buffer size influence the randomness of dataset.
   - Batch size influence the GPU RAM requirements.

# Hints

4. Other object detection models
   - Feel free to try other object detection models.
   - It is ok to read other's code on github, but you have to implement it in TensorFlow 2.0.
   - It's **not allowed** to load other's pre-trained model which was already trained on object detection task.

# Precautions

1. The final score will be only based on your ranking on private leaderboard(80%) and report(20%).
2. Training on the datasets that were not provided by us is forbidden.
3. Loading the model pre-trained on ImageNet is allowed, while loading the model trained on object detection task is not allowed.
4. Discussion is welcomed, while plagiarism gets you 0 point.
5. Using ground truth to generate output will get you 0 point, too.

# Competition Timeline

- 2019/11/14 competition announced.
- 2019/12/06 23:59(UTC) competition deadline.
- 2019/12/08 23:59(TW) report deadline.
- 2019/12/10 winner team share.