# AIRO Lab 1-2: Object Detection

## 1 Introduction

Along side with image classification, object detection has been a long standing challenge in computer vision. While classification aims to answer the question of what type of objects present in an image, object detection provide much more detail information including the number of objects, their types, their locations and their size. These information are often presented as bounding boxes as in Figure.1.

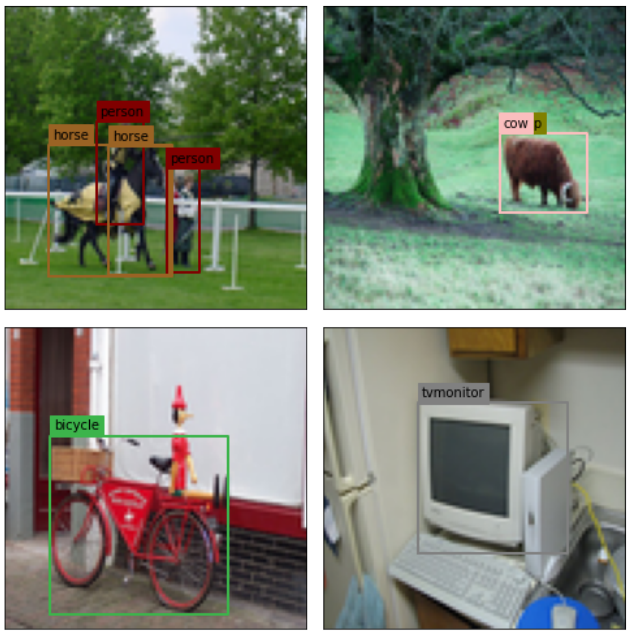


Figure 1: Example of object detection result presented as bounding boxes

Initially, object detection is addressed by anchor-based methods (e.g. Faster R-CNN [4], SSD [2], YOLO [3]) which use an initial guess made of a set of bounding boxes, called anchors, to narrow down the search space. Positive anchors that are anchors having significantly large overlap with ground truth boxes are classified. In parallel, their sizes are adjusted so that they fit as tightly as possible with their associated ground truth boxes. These anchor-based methods have proven their accuracy and efficiency. However, their design are complex and the placement of anchors is highly empirical and somewhat inelegant.

Recently, the so-called anchor-free object detectors have emerged. As mentioned by their names, these methods depart from the idea of using anchors and address object detection as detecting keypoints. Depend on the methods, keypoints to be detected can refer to different entities (e.g. corners [1], or centers [5]). A good representative of this class of detectors is CenterNet [5] which achieve state-of-the-art result on a number of challenging datasets. Its main idea is

1. Localize objects by localizing their centers

2. From these centers, objects' sizes are found through regression

The goal of this lab is to perform object detection on the PASCAL VOC dataset using the framework of CenterNet [5].

# 2 Theory

## 2.1 Encode bounding boxes according to CenterNet

As mentioned in the end of Section.1, the main idea of CenterNet is to localize objects via their centers and then construct bounding boxes around these centers by regression. This idea is formulated as a deep learning problem as the following

- Localize object can be regarded as classifying pixels which is to check for every pixel whether it is a center of an object or not
- Finding object sizes is trivially a regression problem

### 2.1.1 Cool the classification with heat maps

Classifying pixels into center and non-center class has one major drawback which is the sparseness of the supervision signal. In other word, the negative class (non-center) overwhelms the positive class. This is a deal-breaker for any neural nets because the trained network will be severely biased toward the negative class.

A remedy for this drawback is originated from the following observations

- Given a fixed box size, pixels in a region around the true center can still generate a box having large overlap with the ground truth box. Therefore, we can increase the positive class by making a region around a center positive.
- To ensure that centers predicted by trained model is as close as possible to their ground truth counterparts, positive pixels far away from centers should be less positive than those close.


Figure 2: Examples of heat map generated for a single class of objects in different images in VOC'07. The hotter color, the higher probability.

Synthesizing two observations above results in the reformulation of ***localizing objects as predicting the probability of being a center for each pixel***. The target being-center probability, or **probability** (to be short), of each pixel in an image is defined by placing a

Gaussian around the center of each object as in Figure.2. To distinguish one class of objects from the others, multiple target heat maps are generated for each image, as shown in Figure.3. Heat map `i` denote the probability of pixels being centers of objects of class `i`.


Figure 3: Multiple heat maps for different classes generated for one image.

Formally, let's $C$ denote the total number of classes in the dataset of interest, the target heat map of an input image $\mathbf{I} \in \mathcal{R}^{3\times H\times W}$ is $\mathbf{Y} \in [0,1]^{C\times H'\times W'}$. It's worth to notice that the size each channel of the target heat map $H' \times W'$ is different than the size of the image $H \times W$. Usually, $(H', W') = \left(\frac{H}{R}, \frac{W}{R}\right)$ with $R = 4$ which means the target heat map is a low-resolution (down sampled by the factor of $R$) version of the image. This downsampling is to reduce the computation for generating of the predicted heat map $\hat{\mathbf{Y}} \in [0,1]^{C\times \frac{H}{R}\times \frac{W}{R}}$.

### 2.1.2 Construct heat maps with Gaussians

**Gaussians** The target heat map $\mathbf{Y}$ for each image $\mathbf{I}$ is generated by placing a Gaussian around the center of every object. Formally, let $\mathbf{p} = (p_x, p_y)$ denote the center of a ground truth box of class `i` in an image $\mathbf{I}$ and $\tilde{\mathbf{p}} = \lfloor \frac{\mathbf{p}}{R} \rfloor$ be its location in the channel `i` of the target heat map $\mathbf{Y} \in [0,1]^{C\times \frac{H}{R}\times \frac{W}{R}}$. Recall $R$ is downsampling factor applied to the image $\mathbf{I}$ which usually takes the value of 4. A pixel $\mathbf{q} = (q_x, q_y)$ on the same heat map's channel within the square of size $6\sigma_P + 1$ centered on $\tilde{\mathbf{p}}$ has the probability defined by Equation.1

$$\mathbf{Y}[\mathtt{i}, q_y, q_x] = \exp\left(-\frac{(q_x - \tilde{p}_x)^2 + (q_y - \tilde{p}_y)^2}{2\sigma_P^2}\right) \tag{1}$$

Here and the subsequent equations, $[\cdot,\cdot,\cdot]$ is the indexing operation. The Gaussian's standard deviation $\sigma_P$ is calculated based on the size of objects.

**Computing $\sigma_P$** CenterNet follows CornerNet [1]'s convetion in computing $\sigma_P$. The convention, illustrated in Figure.4, is

- For each ground truth box define two circles of radius $r$ centering on its top-left and bottom-right corner.
- The radius $r$ is identified such that any rectangular box constructed from two points respectively chosen from those two circles has its IoU with the ground truth box no smaller than a threshold $t$.
- The value of $\sigma_P$ is set to $\frac{r}{3}$

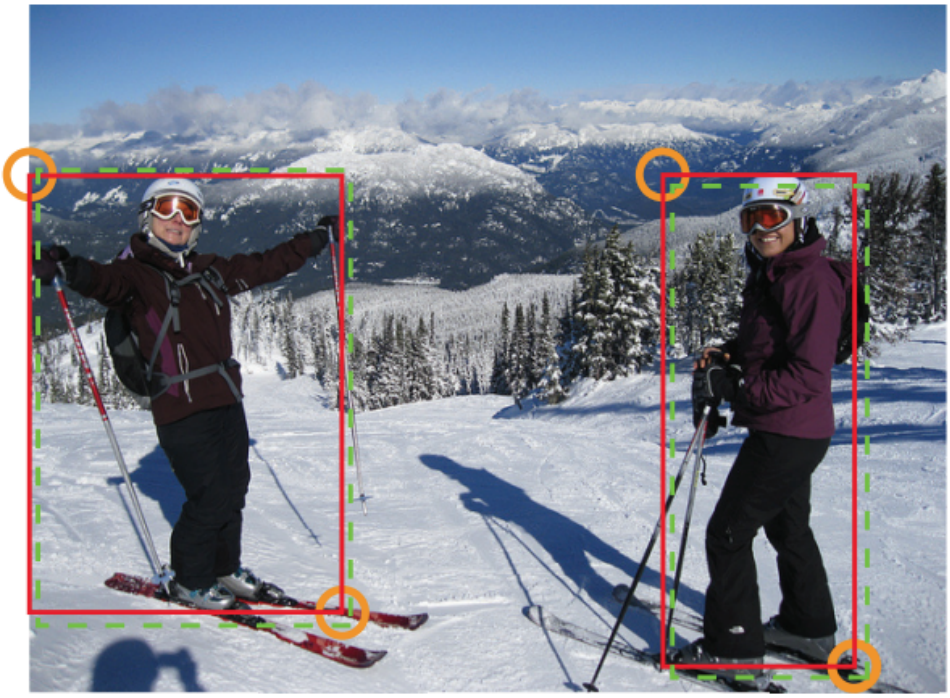

Figure 4: Convetion for computing $\sigma_P$. Two orange circle are placed on top-left and bottom-right corners of each ground truth boxes denoted continuous red line. The radius $r$ of each pair of orange circles are defined such that any green-dash box constructed from a pair of points on these circles has IoU with the red box no smaller than a threshold $t$. This image is from [1].

### 2.1.3 Regression target

**Boxes' Size** Similar to the probability, the width and height of bounding boxes are encoded as 2D grids to form width map $\mathbf{W} \in \mathcal{R}^{1\times\frac{H}{R}\times\frac{W}{R}}$ and height map $\mathbf{H} \in \mathcal{R}^{1\times\frac{H}{R}\times\frac{W}{R}}$. However, these maps have two differences compared to the heat map

- The width and height map are class agnostic. In other words, there is only one width map and one height map shared among $C$ classes.
- The width map $\mathbf{W}$ and height map $\mathbf{H}$ contains nonzero value for the true center only.

Recall from the Section.2.1.2, a ground-truth center $\mathbf{p} = (p_x, p_y)$ in the image $\mathbf{I}$ has the position on the heat map defined by $\tilde{\mathbf{p}} = \lfloor \frac{\mathbf{p}}{R} \rfloor$. The value of width map $\mathbf{W}$ and height map $\mathbf{H}$ at $\tilde{\mathbf{p}}$ is

$$\begin{aligned} \mathbf{W}[0, \tilde{p}_y, \tilde{p}_x] &= \frac{w}{R} \\ \mathbf{H}[0, \tilde{p}_y, \tilde{p}_x] &= \frac{h}{R} \end{aligned} \tag{2}$$

Here, $(w, h)$ is the width and height of the ground box whose center in the image $\mathbf{I}$ is $\mathbf{p}$.

**Center Offset** Because of the $\lfloor \cdot \rfloor$ operator in computing the low-resolution center $\tilde{\mathbf{p}}$, even when $\tilde{\mathbf{p}}$ is correctly located from the predicted heat map $\hat{\mathbf{Y}}$, the ground-truth center $\mathbf{p}$ can be anywhere in the region of $R$ pixels centering around $R \times \tilde{\mathbf{p}}$. To improve the detection accuracy, such offset between $R \times \tilde{\mathbf{p}}$ and $\mathbf{p}$ need to be predicted. As a result, the offset map $\mathbf{O} \in \mathcal{R}^{2\times\frac{H}{R}\times\frac{W}{R}}$ are created as following

- The offset map $\mathbf{O}$ contains nonzero value of center pixels only

- Given a ground truth center $\mathbf{p}$, the value of the offset map $\mathbf{O}$ at its low resolution version $\tilde{\mathbf{p}} = \lfloor \frac{\mathbf{p}}{R} \rfloor$ is

$$\begin{aligned} \mathbf{O}[0, \tilde{p}_y, \tilde{p}_x] &= \frac{p_x}{R} - \tilde{p}_x \\ \mathbf{O}[1, \tilde{p}_y, \tilde{p}_x] &= \frac{p_y}{R} - \tilde{p}_y \end{aligned} \tag{3}$$

#### 2.1.4 Overall target for an image

To sum up, an image has three targets

- The heat map $\mathbf{Y} \in [0,1]^{C \times \frac{H}{R} \times \frac{W}{R}}$ denotes the probability of each pixel being a center of a certain class
- The size map (made of width map $\mathbf{W}$ and height map $\mathbf{H}$) $\mathbf{S} \in \mathcal{R}^{2 \times \frac{H}{R} \times \frac{W}{R}}$ denotes the size of the box at each center pixel
- The offsetmap $\mathbf{O} \in [0,1]^{C \times \frac{H}{R} \times \frac{W}{R}}$ denotes the offset between ground truth centers and their low resolution counterparts

Concatenating these maps along the channel dimension results in the target $\mathbf{T}$ for each image. This tensor has the size of $(C+4) \times \frac{H}{R} \times \frac{W}{R}$.

### 2.2 Decode CenterNet prediction

Since CenterNet's target is defined in form of heat map $\mathbf{Y}$, size map $\mathbf{S}$, and offset $\mathbf{O}$, the output of a model in CenterNet's framework has the same form. Such an output is interpreted into bounding by first detecting all pixels whose probability is greater or equal to theirs 8-connected neighbor on each heat map's channel and keep the top 100 peaks across all channels. Let the triplet $(\hat{x}, \hat{y}, \mathtt{c})$ denote a peak in the channel $\mathtt{c}$ of $\mathbf{Y}$, the corresponding box in the center-width-height form is retrieved using Equation.4

$$\left[\hat{x} + \mathbf{O}[0, \hat{x}, \hat{y}] \quad \hat{y} + \mathbf{O}[1, \hat{x}, \hat{y}] \quad \mathbf{S}[0, \hat{x}, \hat{y}] \quad \mathbf{S}[1, \hat{x}, \hat{y}]\right] \times R \tag{4}$$

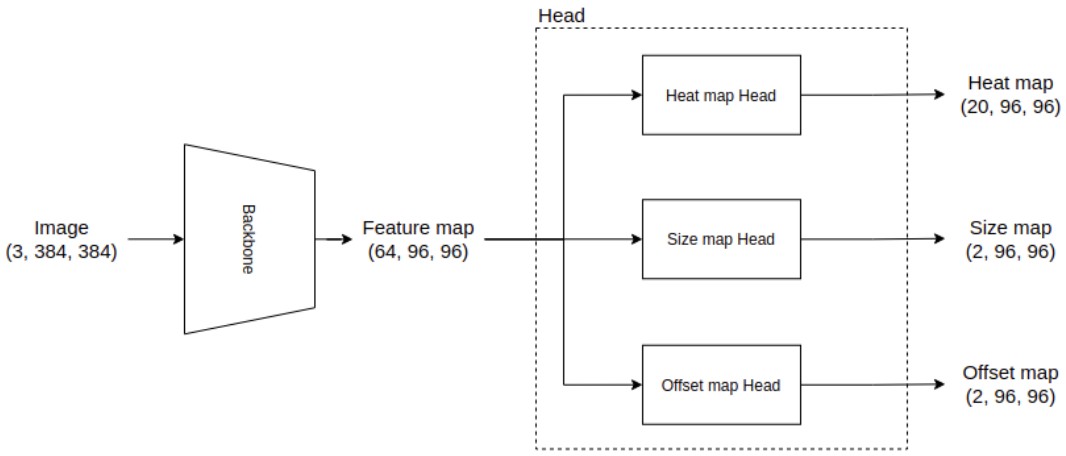


Figure 5: An architecture in the CenterNet framework.

## 2.3 Architecture

Section.2.1.4 defines the target of each image is a 3D tensor of size $(C+4) \times \frac{H}{R} \times \frac{W}{R}$. As a result, the architecture of choice should produce its prediction of the same format - a 3D tensor, in other words a stack of 2D grids. To produce such an output, a popular option is *fully convolutional* architectures which can be describe holistically as neural nets that receive an image and produce another "image" probably with smaller size and larger depth.

In the framework of CenterNet, a network, as shown in Figure.5, is made of two sub network: a backbone and a head. The backbone is a fully convolutional architecture that in charge of building a feature map $\mathbf{F} \in \mathcal{R}^{D \times \frac{H}{R} \times \frac{W}{R}}$ from input image $\mathbf{I} \in \mathcal{R}^{3 \times H \times W}$. The value of $D$ depends on the backbone's details. The head made of a stack of convolution layers converts the backbone's output into model's prediction which is a tensor of shape $(C+4) \times \frac{H}{R} \times \frac{W}{R}$.

### 2.3.1 Backbone - ResNet18 with upsampling path

Among different backbone architectures presented in [5], the scope of this lab is limited the simplest one - ResNet18 with up-sample convolutions because this architecture achieves relatively good result on the targeted dataset - PASCAL VOC.

The detail of ResNet18 with upsample convolution is shown in Figure.6

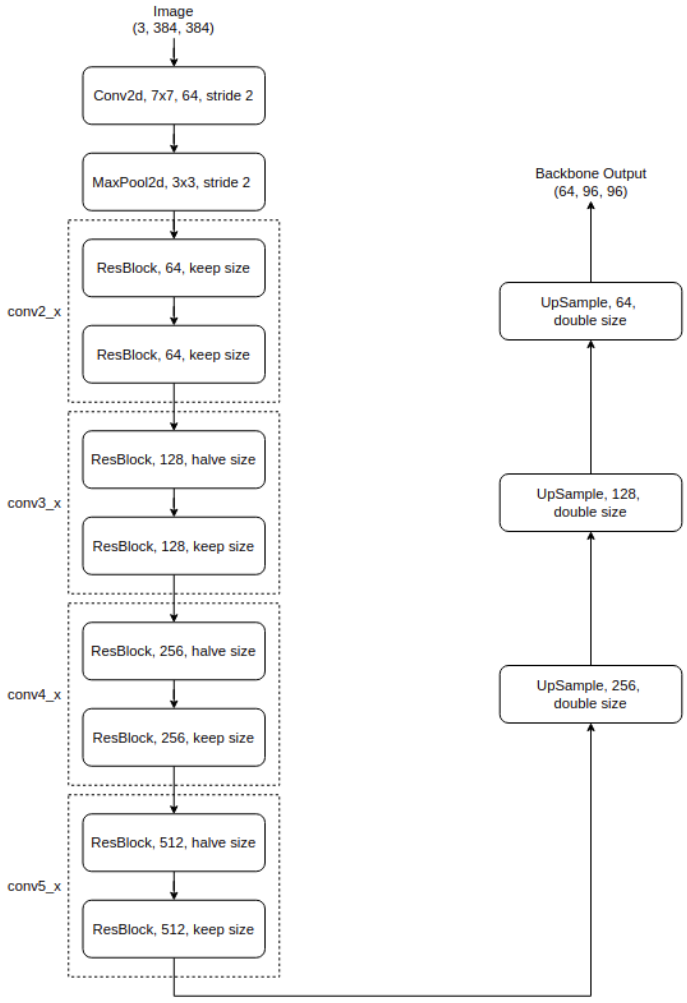


Figure 6: Architecture of backbone made of ResNet18 with up-sample convolution. The number written on each block the number of channels of block's output. The phrase halve size in a block's name indicates that this height and width of this block's output are half of its input.

**Residual block** The architecture of a *ResBlock* in the case of keep size and halve size are respectively shown in Figure.7 and Figure.8. In the "keep size" case, two `Conv2d` on the Main Pass produce outputs having the same size (height and width) as the input. On the other hand, when the block halves the size the first `Conv2d` on the Main Pass and the `Conv2d` on the Skip Connection have their `padding` and `stride` set such that their outputs' size is just half of the input.

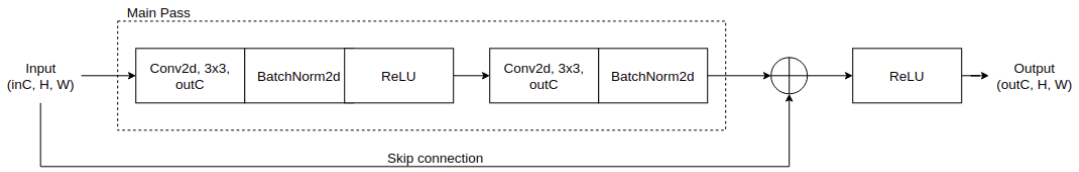


Figure 7: Architecture of a residual block in the case of output having the same size as input. Here `outC` in the number of channels of the output of this entire block.

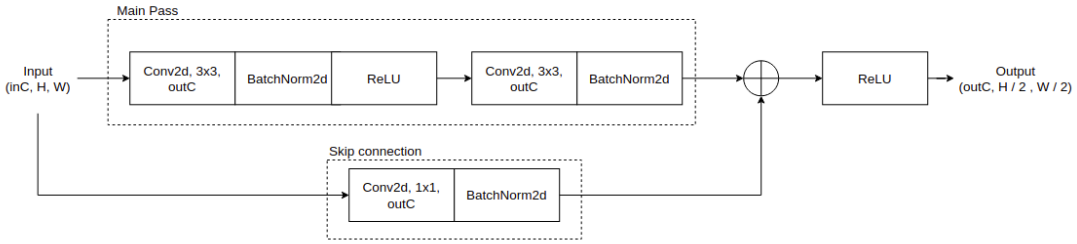


Figure 8: Architecture of a residual block in the case of output having half the size of input. Here `outC` in the number of channels of the output of this entire block.The downsampling takes place in the first `Conv2d` of the Main Pass and the `Conv2d` on the Skip Connection by setting the appropriate `padding` and `stride`.

It is important to notice that when a `Conv2d` is followed by a `BatchNorm2d` it must not have bias, because the bias is brought by `BatchNorm2d`. On the other hand, when a `Conv2d` is not accompanied by a a `BatchNorm2d`, it should have bias.

**Up-sample block** An up-sample block's architecture is shown in Figure.9 . It is made by combining a `Conv2d` layer with a `ConvTranspose2d` layer.

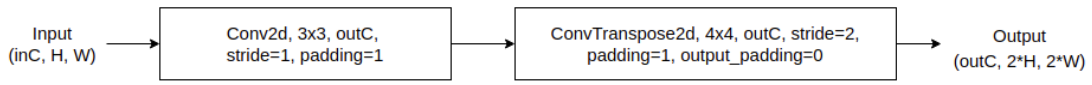


Figure 9: Architecture of an up-sample block which produces the output of twice the size of the input.

### 2.3.2 A three-branch Head

The expected output of the architecture comprises of 3 maps: heat map, size map, and offset map. As a result 3 dedicated heads whose architecture is shown in Figure.10 are employed to predict each of these maps.

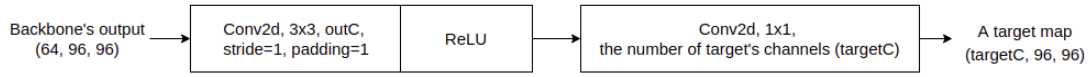


Figure 10: Architecture of a branch of the Head.

### 2.4 Loss Function

The loss function $L$ the weighted sum of heat loss $L_{\text{heat}}$, size loss $L_{\text{size}}$, and offset loss $L_{\text{offset}}$

$$L = L_{\text{heat}} + \lambda_{\text{size}} L_{\text{size}} + \lambda_{\text{offset}} L_{\text{offset}} \tag{5}$$

Here, $\lambda_{\text{size}} = 0.1$ and $\lambda_{\text{offset}} = 1.0$.

**Heat loss** $L_{\text{heat}}$ is the focal loss calculated for every pixel across every channel of the predicted heat map $\hat{\mathbf{Y}} \in \mathcal{R}^{C \times \frac{H}{R} \times \frac{W}{R}}$

$$L_{\text{heat}} = \frac{-1}{N} \sum_{cxy} \begin{cases} \left(1 - \hat{\mathbf{Y}}[c,y,x]\right)^{\alpha} \log\left(\hat{\mathbf{Y}}[c,y,x]\right) & \text{if } \mathbf{Y}[c,y,x] == 1 \\ \left(1 - \mathbf{Y}[c,y,x]\right)^{\beta} \left(\hat{\mathbf{Y}}[c,y,x]\right)^{\alpha} \log\left(1 - \hat{\mathbf{Y}}[c,y,x]\right) & \text{otherwise} \end{cases} \tag{6}$$

Here, $c, x, y$ are respectively the channel index, the column index, and the row index. $N$ is the total number of ground truth centers.

**Size loss** $L_{\text{size}}$ is only calculated for low resolution centers $\tilde{\mathbf{p}}^k = \left(\tilde{p}_x^k, \tilde{p}_y^k\right) (k = 1, ..., N)$

$$L_{\text{size}} = \frac{1}{N} \sum_{k=1}^{N} \sum_{i=0}^{1} |\mathbf{S}[i, \tilde{p}_y^k, \tilde{p}_x^k] - \hat{\mathbf{S}}[i, \tilde{p}_y^k, \tilde{p}_x^k]| \tag{7}$$

Here, $\hat{\mathbf{S}}$ denote the predicted size map.

**Offset loss** $L_{\text{offset}}$ is calculated from target offset map $\mathbf{O}$ and predicted offset map $\hat{\mathbf{O}}$ in the same manner as size loss.

## 3 Implementation

This lab is made of multiple files with multiple functions to fill in. The places to fill in are marked with `TODO` in the skeleton code and sometimes are accompanied with hints.

To help you make concrete progress in this lab, test cases are provided in the directory `test`. Test cases concerning functions or classes in the file `a_file_name.py` are placed in the file `test_a_file_name.py`. Upon finishing a single function, you can run the corresponding test case to validate your implementation.

For your convenience, the workload of this lab is summarized in Table.1.

## References

[1] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 734–750.

| File (*.py) in directory | Description | Functions or Classes |
|---|---|---|
| parse_raw_dataset in tools | Parse raw dataset and dump result into a list of image paths and a list of annotations | create_data_lists |
| dataset_utils in tools | Provide functionalities for creating target maps $\mathbf{Y}, \mathbf{S}, \mathbf{O}$ given an image and its annotation | box_comp_cornerNet_radius, draw_gaussian_on_matrix, box_to_label |
| dataset in tools | Create the class VOC by inheriting from torch.utils.data.Dataset | VOC |
| model_utils in model | Provide functionalities for creating model and decode model's prediction | idx1d_to_indices3d, draw_gaussian_on_matrix, nms_heat_map, get_topK_peaks, decode_prediction |
| resnet in model | Define Residual block (referred to as BasicLayer in the code), Up-sample block, and the backbone | BasicLayer, UpSampleLayer, ResNet18 |
| centernet_head in model | Define the Head of CenterNet | conv_block, CenterNetHead |
| loss_function in model | Construct the loss function defined in Equation.5 | CenterNetLoss |

Table 1: Unfinished functions and where to find them


[2] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[3] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[4] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *arXiv preprint arXiv:1506.01497*, 2015.

[5] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," *arXiv preprint arXiv:1904.07850*, 2019.