Perceptual Edge

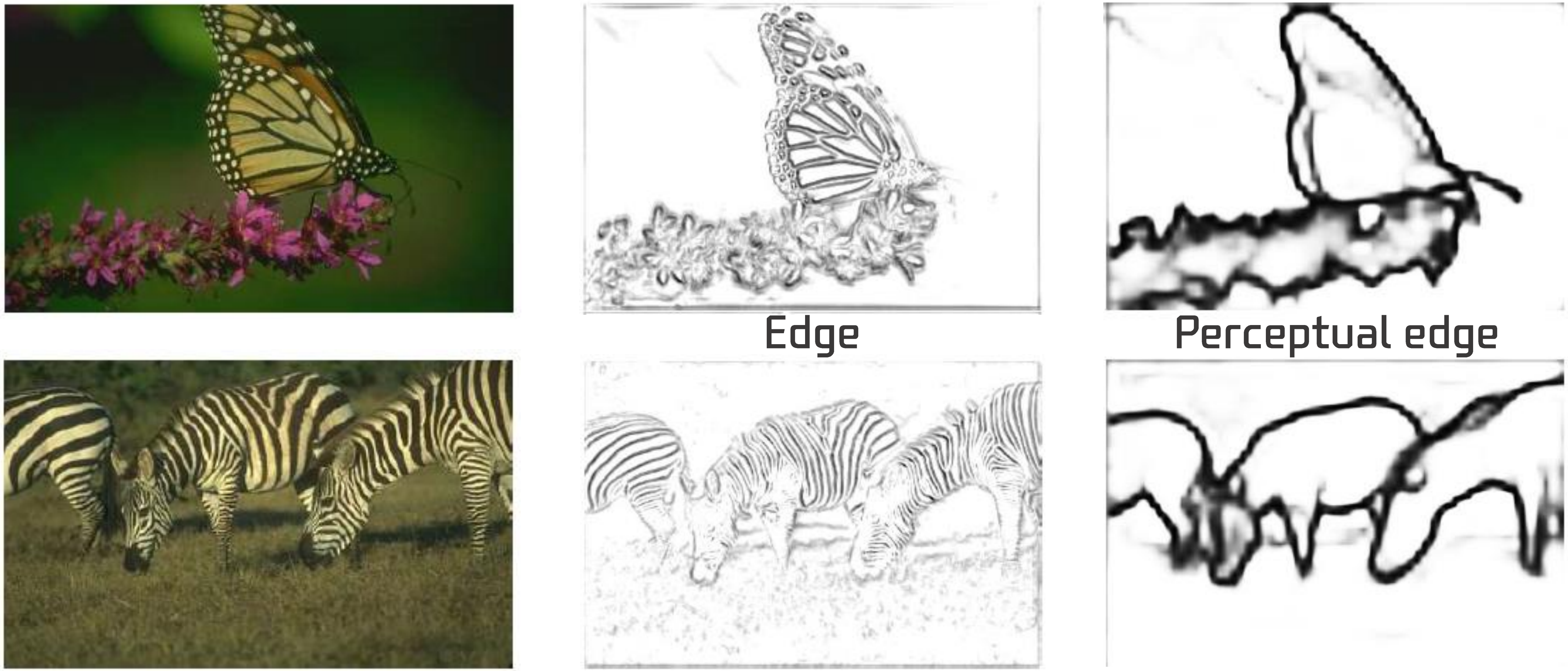
Digital signal processing Lab
Presenter: KIM JONGHYUN

Contents

001 Concept

Digital Signal Processing Lab

Perceptual edge



Perceptual edge
: Object-level edge

002 Proposed method

Digital Signal Processing Lab

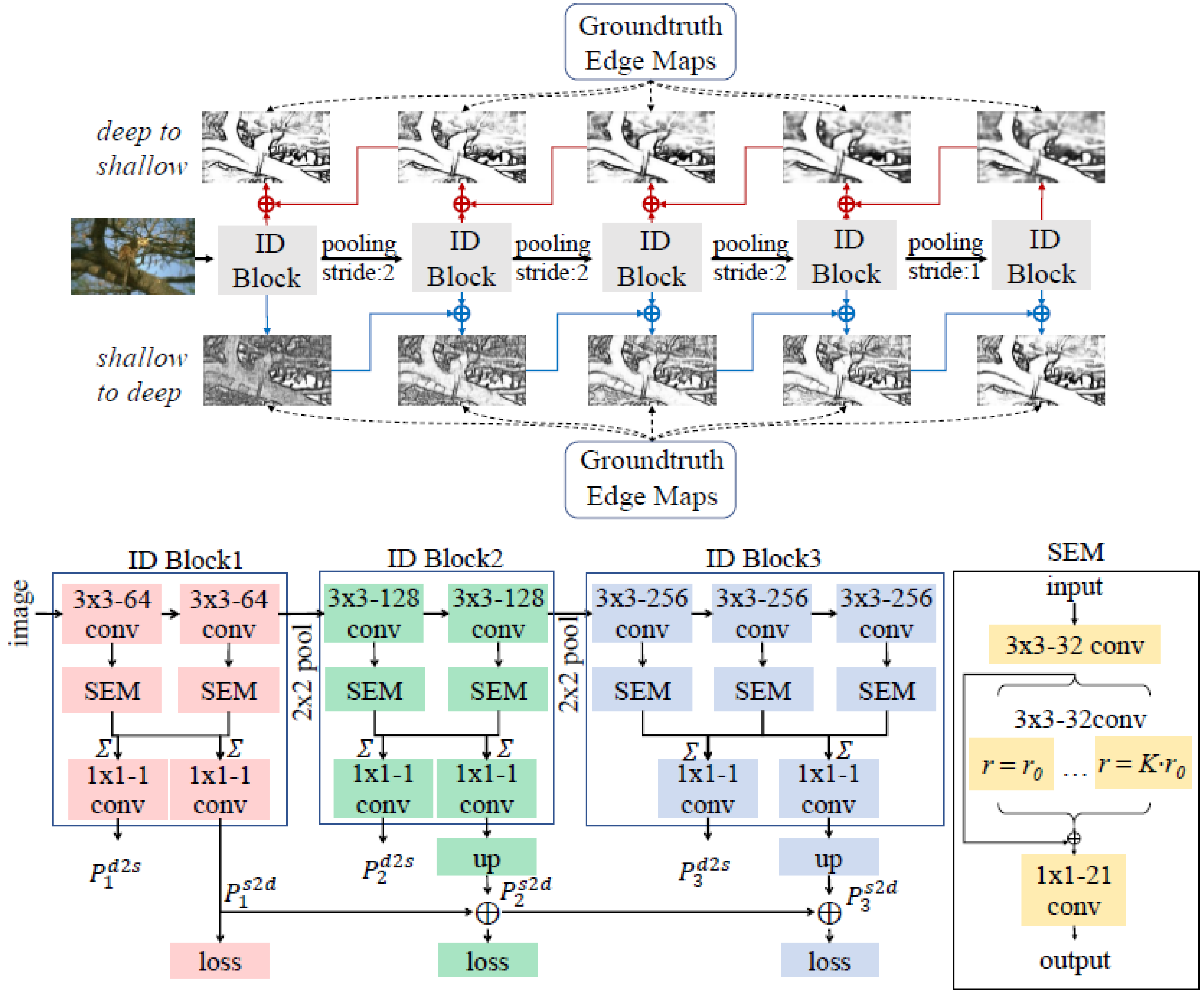
2019, CVPR

Bi-Directional Cascade Network for Perceptual Edge Detection

Jianzhong He[1],Shiliang Zhang[1],Ming Yang[2],Yanhu Shan[2],Tiejun Huang[1]
[1]Peking University, [2]Horizon Robotics, Inc.
{jianzhonghe,slzhang.jdl,tjhuang}@pku.edu.cn,m-yang4@u.northwestern.edu, yanhu.shan@gmail.com

002 Proposed method

Contour detection



Bi-directional & Cascaded

Contour detection

- Y : Ground truth edge map
- Ys : Annotated edges corresponding to a scale s

$$Y = \sum_{s=1}^{S} Y_s, \quad (1)$$

Sum of different scales of edge maps = GT

Contour detection

- *Ps* : Edge prediction at scale *s*
- *Ls* : Training loss
- *X* : Input image
- *T* : Train set

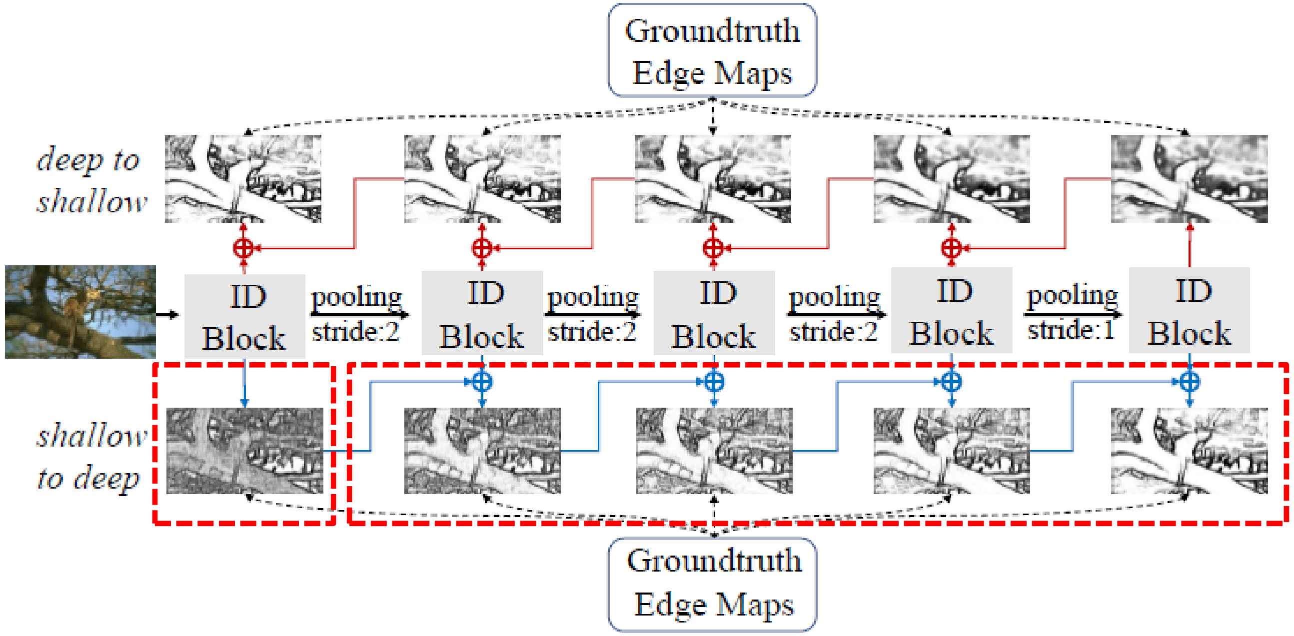
$$\mathcal{L}_s = \sum_{X \in \mathbb{T}} |P_s - Y_s|, \tag{2}$$

To find approximate Ys by training Eq(2)

How to get *Ys* ?

002 Proposed method

Contour detection



$$Y_s \sim Y - \sum_{i \neq s} P_i. \tag{3}$$

To get Y_1, $Y - \sum_{i \neq 1} P_i$

edges P_s at layer s should approximate Y_s, *i.e.*, $P_s \sim Y - \sum_{i \neq s} P_i$.

Contour detection

edges P_s at layer s should approximate Y_s, *i.e.*, $P_s \sim Y - \sum_{i \neq s} P_i$.

Consider all scales, then training equation approximate $Y \sim \sum_i P_i$.

- L : Training objective
- $\hat{Y}$: Prediction at all scales
- Y : Ground truth

$$\mathcal{L} = \mathcal{L}(\hat{Y}, Y), \text{ where } \hat{Y} = \sum_i P_i.$$

$$\frac{\partial(L)}{\partial(P_s)} = \frac{\partial(L(\hat{Y}, Y))}{\partial(P_s)} = \frac{\partial(L(\hat{Y}, Y))}{\partial(\hat{Y})} \cdot \frac{\partial(\hat{Y})}{\partial(P_s)}. \qquad (4)$$

Calculate the gradient at a scale s

Contour detection

- $\hat{Y}$: Prediction at all scales
- P_s, P_i : Prediction at s and i scales
- ∂ : Partial derivative

$$\frac{\partial(L)}{\partial(P_s)} = \frac{\partial(L(\hat{Y}, Y))}{\partial(P_s)} = \frac{\partial(L(\hat{Y}, Y))}{\partial(\hat{Y})} \cdot \frac{\partial(\hat{Y})}{\partial(P_s)}. \qquad (4)$$

$$\frac{\partial(\hat{Y})}{\partial(P_s)} = \frac{\partial(\hat{Y})}{\partial(P_i)} = 1$$

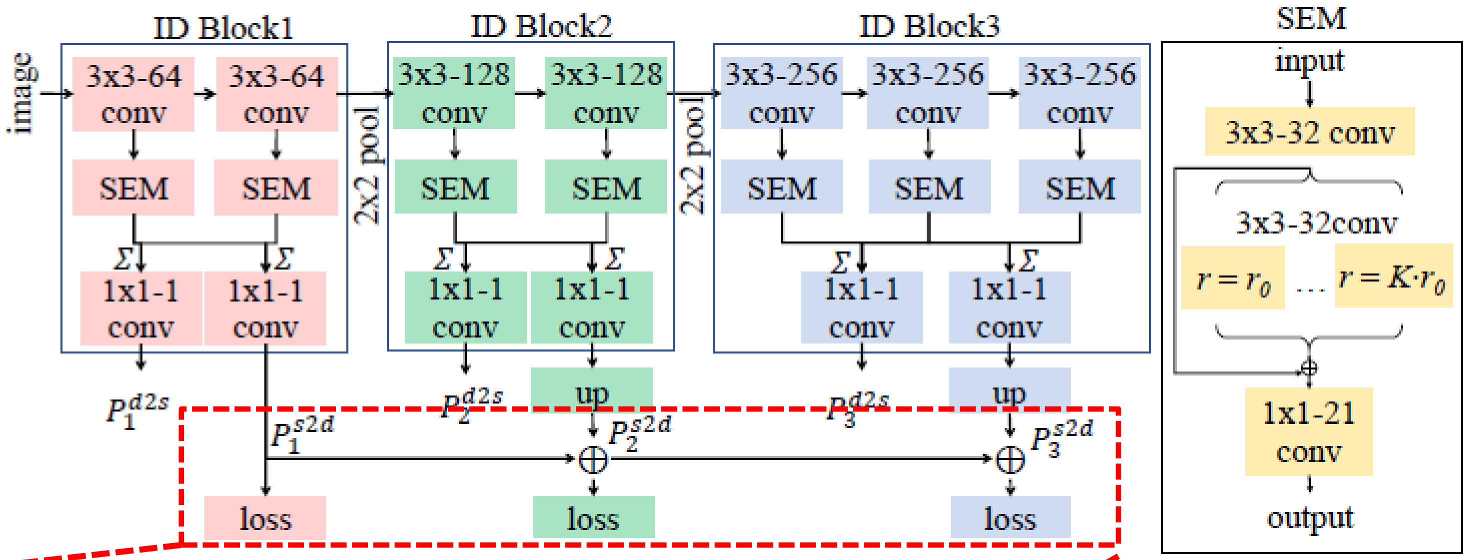
All gradient at any scale is the same as “1”

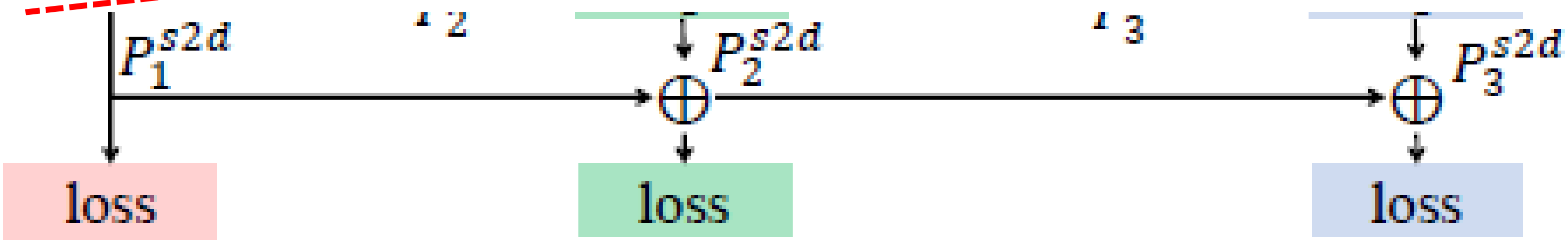
$$\hat{Y} = \textstyle\sum_i P_i. \qquad \frac{\partial(P_1 + P_2 + \cdots + P_i + \cdots P_s)}{\partial(P_i) \; or \; \partial(P_s)}$$

Contour detection

$$\boxed{Y_s^{s2d} = Y - \sum_{i<s} P_i^{s2d},}$$

$$Y_s^{d2s} = Y - \sum_{i>s} P_i^{d2s},$$





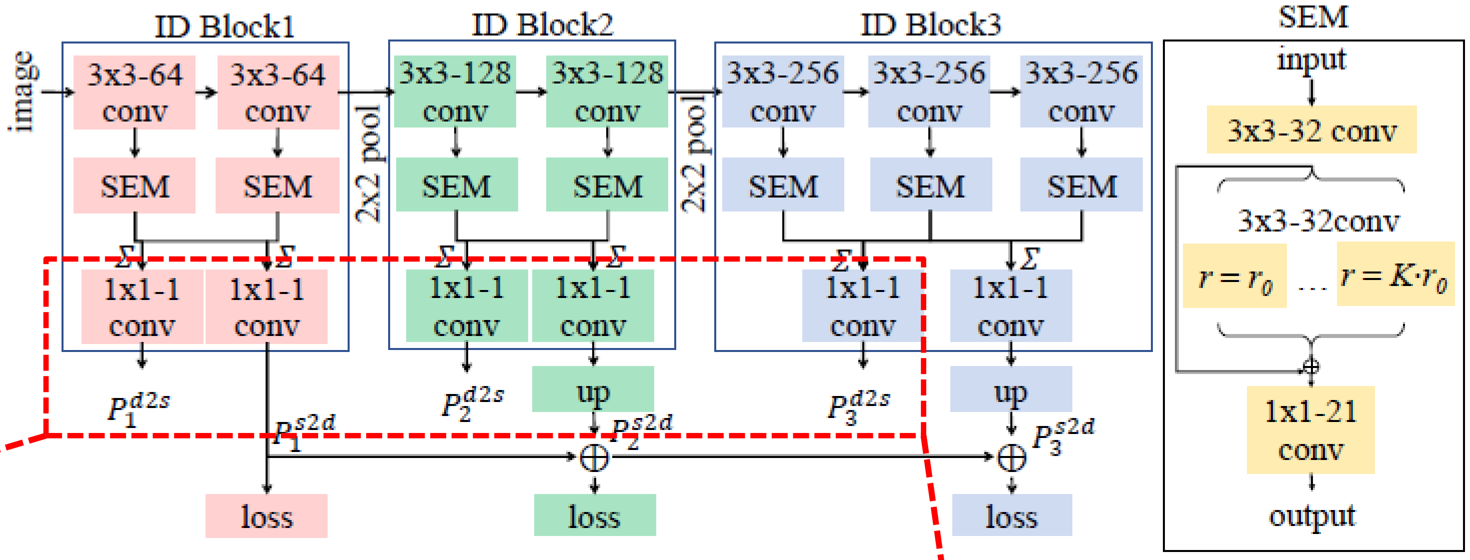
if “s” is 3, then use predictions in 1, 2 scales

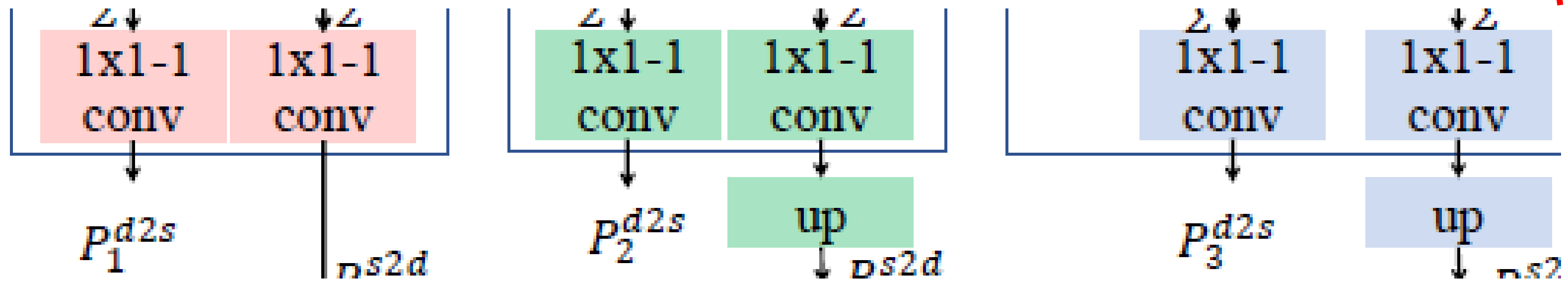
Why do not use scale 4 ? $P_4 \sim Y - (P_1 + P_2 + P_3)$

Contour detection

$$Y_s^{s2d} = Y - \sum_{i<s} P_i^{s2d},$$

$$\boxed{Y_s^{d2s} = Y - \sum_{i>s} P_i^{d2s},}$$





if “s” is 1, then use predictions in 2, 3 scales

Contour detection

For scale s, the predicted edges $P_s{}^{s2d}$ and $P_s{}^{d2s}$ approximate to Y_s^{s2d} and $Y_s{}^{d2s}$, respectively. Their combination is a reasonable approximation to Y_s, *i.e.*,

$$P_s{}^{s2d} + P_s{}^{d2s} \sim 2Y - \sum_{i<s} P_i{}^{s2d} - \sum_{i>s} P_i{}^{d2s}, \quad (6)$$

where the edges predicted at scales $i \neq s$ are depressed. Therefore, we use $P_s{}^{s2d} + P_s{}^{d2s}$ to interpolate the edge prediction at scale s.

Final loss using bi-directional method

Contour detection

- *Dataset* : BSDS500, NYUDv2, Multicue
- *Optimizer* : SGD
- *Batch size* : 10
- *Learning rate* : 1e−6 (decrease 10 times after every 10k iterations.)
- *Total iterations* : 40k for BSDS500 and NYUDv2, 4k for Multicue
- *Augmentation* : randomly cropping 500x500

003 Result

Digital Signal Processing Lab

Prediction at scale s

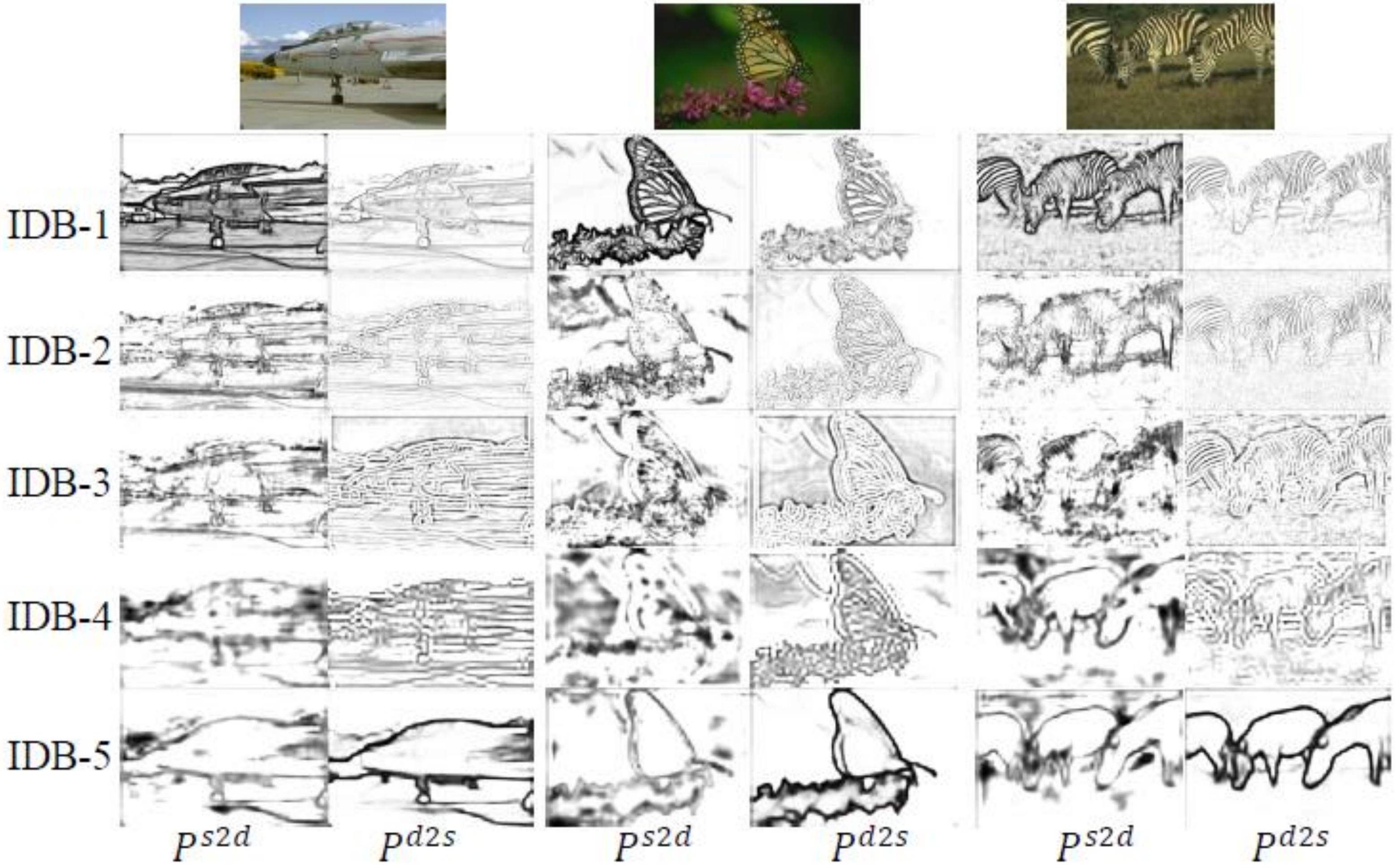
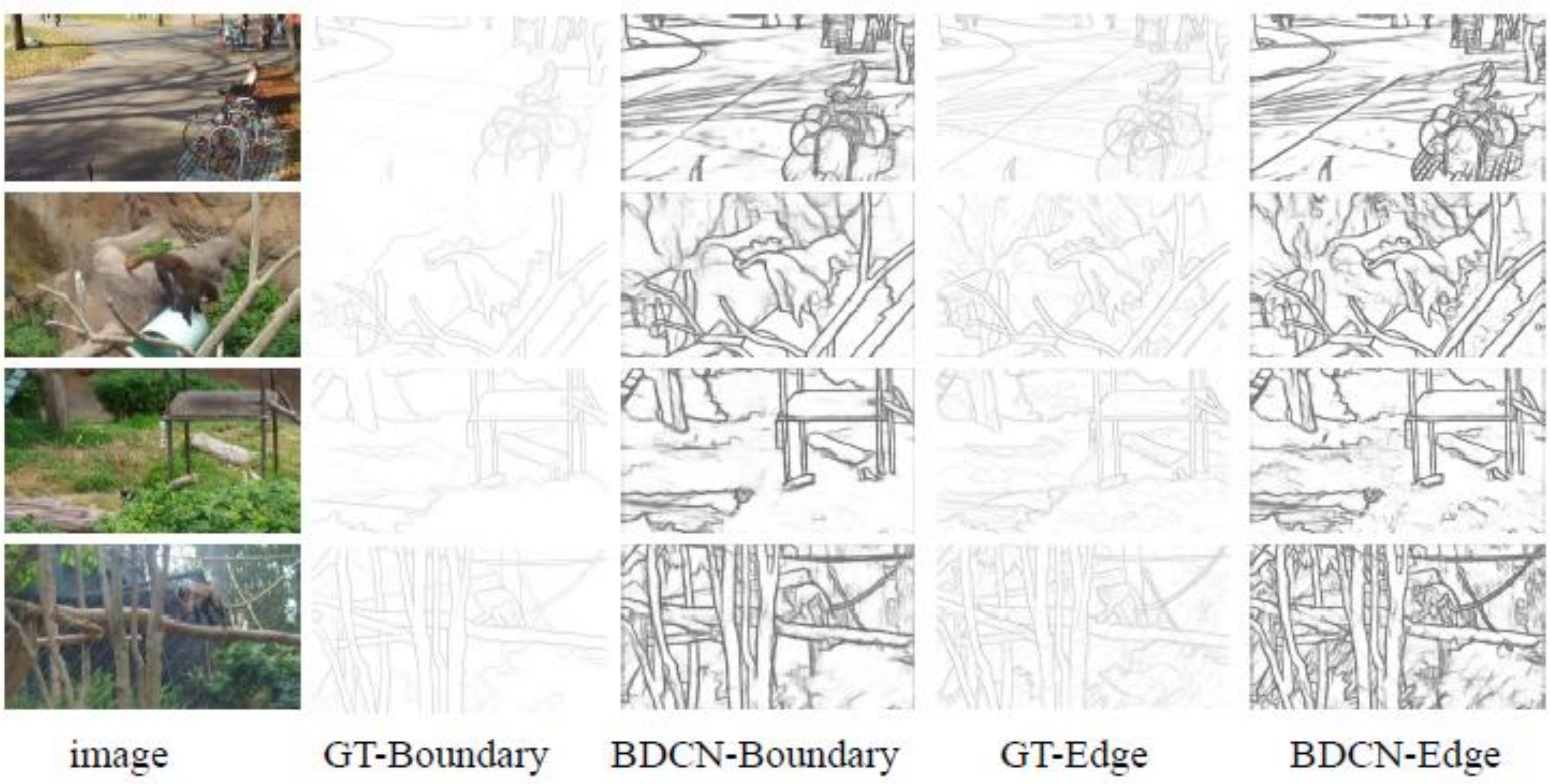


Figure 4. Examples of edges detected by different ID Blocks (IDB for short). Each ID Block generates two edge predictions, P^{s2d} and P^{d2s}, respectively.

003 Result

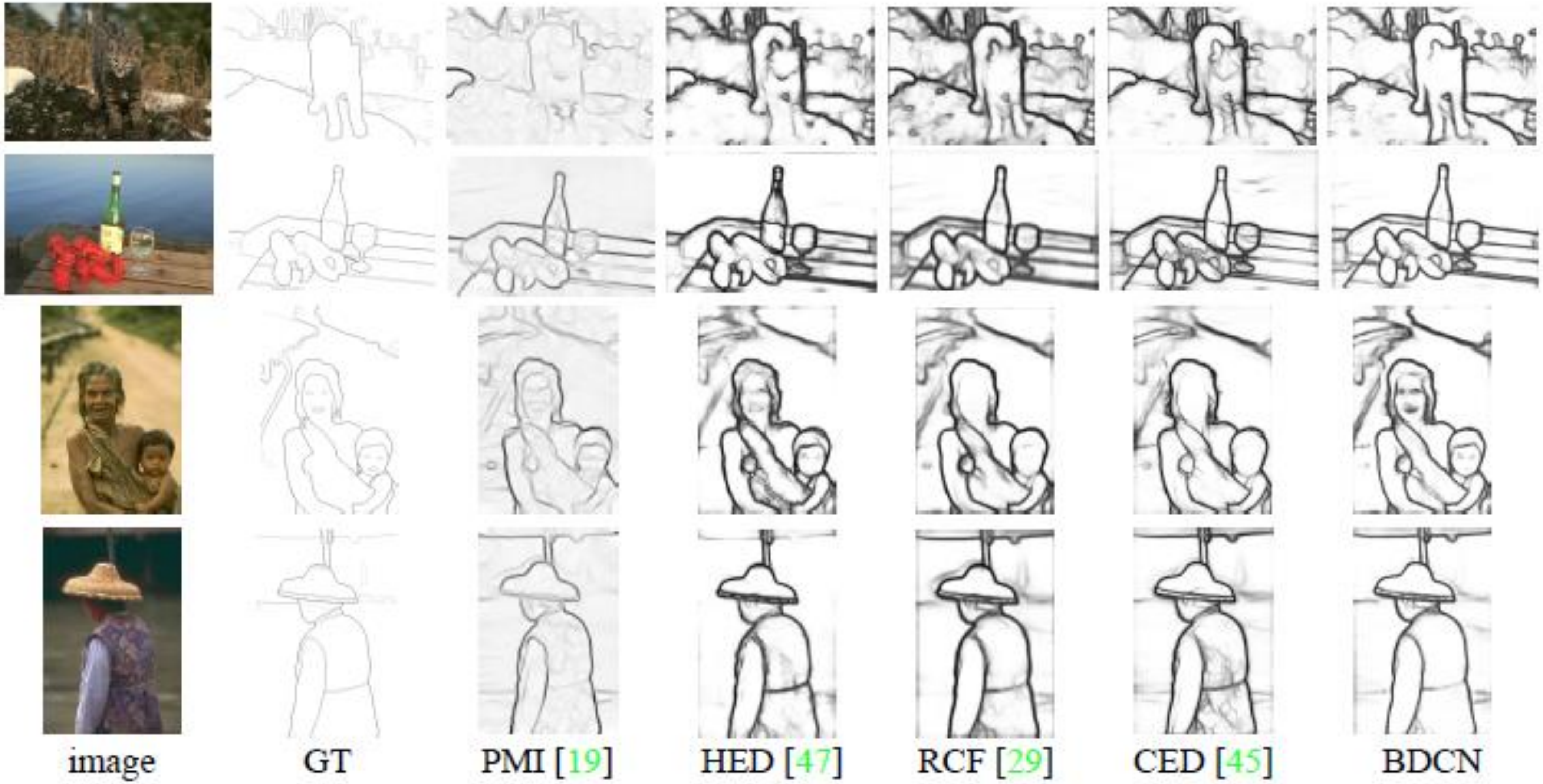
Boundary and edge



3

003 Result

Comparison with SOTA



THANK

YOU