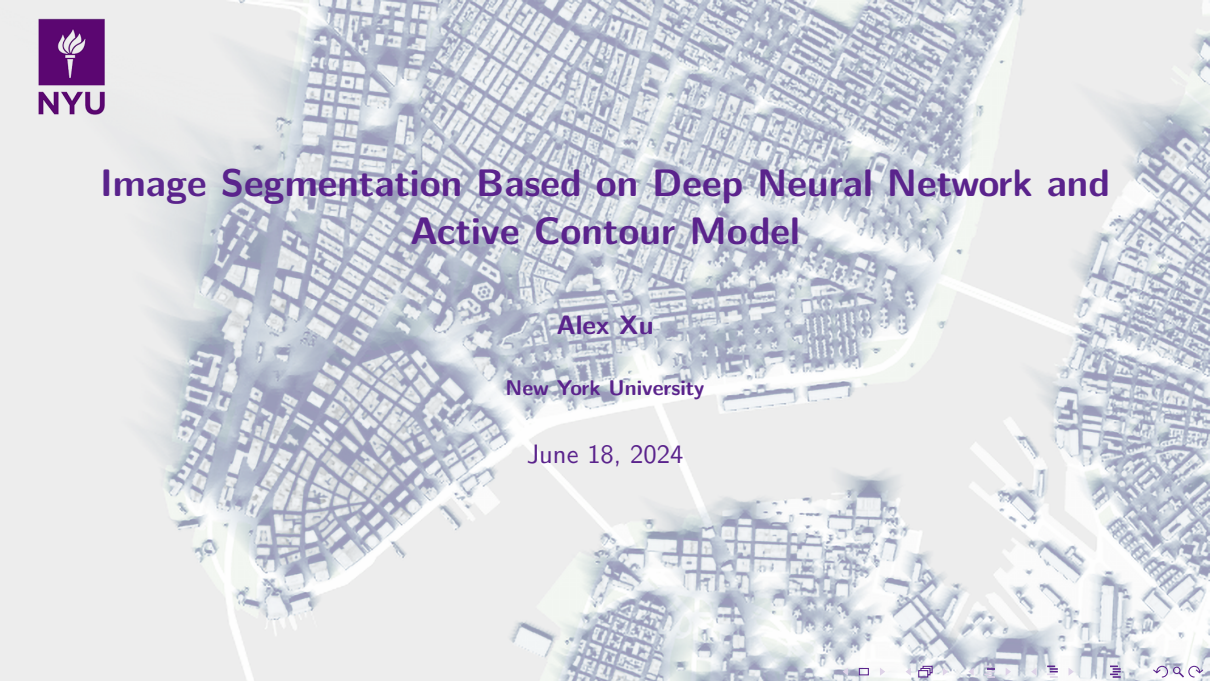

Image Segmentation Based on Deep Neural Network and Active Contour Model

Alex Xu

New York University

June 18, 2024

Content

Background

- Overview of image segmentation.

Background

- Overview of image segmentation.
- Traditional: Accurate segmentation by Active Contour Model.

Background

- Overview of image segmentation.
- Traditional: Accurate segmentation by Active Contour Model.
- Innovation: Deep learning methods.

Active Contour Model

Curve evolution

We define curve as

$$\mathrm{C}(\theta, t) = (x(\theta, t), y(\theta, t)) \tag{1}$$

Then we have curvature

$$\frac{\partial \mathrm{C}(\theta, t)}{\partial t} = \mathrm{F}(k)N \tag{2}$$

As constant evolution

$$\frac{\partial \mathrm{C}(\theta, t)}{\partial t} = V_0 N \tag{3}$$

As curvature evolution

$$\frac{\partial \mathrm{C}(\theta, t)}{\partial t} = \alpha k N \tag{4}$$

Active Contour Model

Level-set method

In curve $\mathrm{C} = \{(x, y), \phi(x, y) = c\}$ where $\phi(x, y)$ defined as level-set function, we introduce time t and apply partial derivatives

$$\frac{\mathrm{d}\phi}{\mathrm{d}t} = \frac{\partial\phi}{\partial t} + \frac{\partial\phi}{\partial x} \cdot \frac{\partial x}{\partial t} + \frac{\partial\phi}{\partial y} \cdot \frac{\partial y}{\partial t} = \frac{\partial\phi}{\partial t} + \nabla\phi \cdot \frac{\partial(x, y)}{\partial t} = 0 \tag{5}$$

$\nabla\phi = (\frac{\partial\phi}{\partial x}, \frac{\partial\phi}{\partial y})$ is the gradient of level set function $\phi(x, y)$, and $\frac{\partial(x,y)}{\partial t} = \frac{\partial \mathrm{C}}{\partial t} = (\frac{\partial x}{\partial t}, \frac{\partial y}{\partial t}) = \vec{V}$. Then

$$\frac{\partial\phi}{\partial t} = -\nabla\phi \cdot \vec{V} = -|\nabla\phi| \cdot \frac{\vec{V}}{\nabla\phi} = |\nabla\phi| \cdot \left(-\frac{\nabla\phi}{|\nabla\phi|}\right) \cdot \vec{V} = |\nabla\phi| \cdot \vec{U} \cdot \vec{V} \tag{6}$$

Active Contour Model

Calculus of variations

The 1D energy function can be written as

$$\mathrm{E}(\phi) = \int_{x_0}^{x_1} \mathrm{F}(x, \phi, \phi')\mathrm{d}x \tag{7}$$

In order to apply Taylor's Formula, we introduce a very small $v(x)$

$$\mathrm{F}(x, \phi + v, \phi' + v') = \mathrm{F}(x, \phi, \phi') + \frac{\partial \mathrm{F}}{\partial \phi} \cdot v + \frac{\partial \mathrm{F}}{\partial \phi'} \cdot v + \cdots \tag{8}$$

Intergrate both sides and collate to get

$$\mathrm{E}(\phi + v) - \mathrm{E}(\phi) = \int_{x_0}^{x_1} \left(v\frac{\partial \mathrm{F}}{\partial \phi} + v'\frac{\partial \mathrm{F}}{\partial \phi'} \right) \mathrm{d}x \tag{9}$$

Active Contour Model

Calculus of variations

As $\phi(x_0) + v(x_0) = a, \phi(x_1) + v(x_1) = b$,

$$\int_{x_0}^{x_1} v' \frac{\partial \mathrm{F}}{\partial \phi'} \mathrm{d}x = \int_{x_0}^{x_1} \frac{\partial \mathrm{F}}{\partial \phi} \mathrm{d}v = v \frac{\partial \mathrm{F}}{\partial \phi'} \Big|_{x_0}^{x_1} - \int_{x_0}^{x_1} v \frac{\mathrm{d}}{\mathrm{d}x} \frac{\partial \mathrm{F}}{\partial \phi'} \mathrm{d}x = - \int_{x_0}^{x_1} v \frac{\mathrm{d}}{\mathrm{d}x} \frac{\partial \mathrm{F}}{\partial \phi'} \mathrm{d}x \tag{10}$$

Ignoring very small changes in the independent variable, we have

$$\frac{\partial \mathrm{F}}{\partial \phi} - \frac{\mathrm{d}}{\mathrm{d}x} \left(\frac{\partial \mathrm{F}}{\partial \phi'} \right) = 0 \tag{11}$$

This is the Euler equation for 1D energy functional. Similarly, The Euler equation for 2D is

$$\frac{\partial \mathrm{F}}{\partial \phi} - \frac{\partial \mathrm{F}}{\partial \phi_x} - \frac{\partial \mathrm{F}}{\partial \phi_y} = 0 \tag{12}$$

Active Contour Model

Calculus of variations

Take (11) as an example. To solve the Euler equation, we introduce a small $v(\cdot)$

$$\mathrm{E}(\phi, t+\Delta t) = \mathrm{E}(\phi, t) + \Delta t \int_{x_0}^{x_1} \frac{\partial \phi}{\partial t} \left(\frac{\partial \mathrm{F}}{\partial \phi} - \frac{\mathrm{d}}{\mathrm{d}x} \left(\frac{\partial \mathrm{F}}{\partial \phi'} \right) \right) \mathrm{d}x \tag{13}$$

Let

$$\frac{\partial \phi}{\partial t} = -\left(\frac{\partial \mathrm{F}}{\partial \phi} - \frac{\mathrm{d}}{\mathrm{d}x} \left(\frac{\partial \mathrm{F}}{\partial \phi'} \right) \right) = \frac{\mathrm{d}}{\mathrm{d}x} \left(\frac{\partial \mathrm{F}}{\partial \phi'} \right) - \frac{\partial \mathrm{F}}{\partial \phi} \tag{14}$$

Then

$$\Delta \mathrm{E} = \mathrm{E}(\phi, t+\Delta t) - \mathrm{E}(\phi, t) = -\Delta t \int \left(\frac{\partial \mathrm{F}}{\partial \phi} - \frac{\mathrm{d}}{\mathrm{d}x} \left(\frac{\partial \mathrm{F}}{\partial \phi'} \right) \right)^2 \mathrm{d}x \tag{15}$$

The solution to the Euler equation of an energy functional tends to stabilize as the curve evolves, corresponding to the gradient flow of the given variation.

Active Contour Model

Chan Vese Model

Chan and Vese proposed the Chan-Vese active contour model. Given a grayscale image $I(x)$, the target Ω_1 and background Ω_2 are detected with characteristic uniformity and a clear difference, represented by the approximate grayscale constants c_1 and c_2. The energy functional is defined as

$$\begin{aligned} \mathrm{F}\left(c_1, c_2, \phi\right) = \lambda_1 \cdot \int_{\Omega} \left(\mathrm{I}(x) - c_1\right)^2 \cdot \mathrm{H}(\phi(x))\mathrm{d}x + \lambda_2 \cdot \int_{\Omega} \left(\mathrm{I}(x) - c_2\right)^2 \cdot (1 - \mathrm{H}(\phi(x)))\mathrm{d}x \\ + \mu \cdot \int_{\Omega} \delta_0(\phi(x))|\nabla\phi(x)|\mathrm{d}x + v \cdot \int_{\Omega} \mathrm{H}(\phi(x))\mathrm{d}x \end{aligned} \tag{16}$$

Where $\lambda_1, \lambda_2 > 0, \mu, v \geq 0$. $\mathrm{H}(\phi)$ is the unit step function(Heaviside function). δ_0 is its derivative, from which the Dirac function can be derived.

$$\mathrm{H}(z) = \begin{cases} 0 & z < 0 \\ 1 & z \geq 0 \end{cases}, \delta_0(z) = \frac{\mathrm{d}}{\mathrm{d}z}\mathrm{H}(z) \tag{17}$$

Active Contour Model

Chan Vese Model

Substituting it into the Euler-Lagrange equation, the level set gradient flow of the Chan-Vese model is solved.

$$\frac{\partial \phi}{\partial \mathrm{t}} = -\frac{\partial \mathrm{F}}{\partial \phi} = \delta_{\varepsilon}\left[\mu \operatorname{div}\left(\frac{\nabla \phi}{|\nabla \phi|}\right) - v - \lambda_1 \left(\mathrm{I}(x) - c_1\right)^2 + \lambda_2 \left(\mathrm{I}(x) - c_2\right)^2\right] \tag{18}$$

where

$$c_1 = \frac{\int \mathrm{I}(x) \cdot H(\phi(x))\mathrm{d}x}{\int \mathrm{H}(\phi(x))\mathrm{d}x}, \quad c_2 = \frac{\int \mathrm{I}(x) \cdot (1 - \mathrm{H}(\phi(x)))\mathrm{d}x}{\int (1 - \mathrm{H}(\phi(x)))\mathrm{d}x} \tag{19}$$

Neural Network

Convolutional Neural Networks

- convolutional layer
- pooling layer
- flatten layer
- fully connected layer
- activation layer

Neural Network

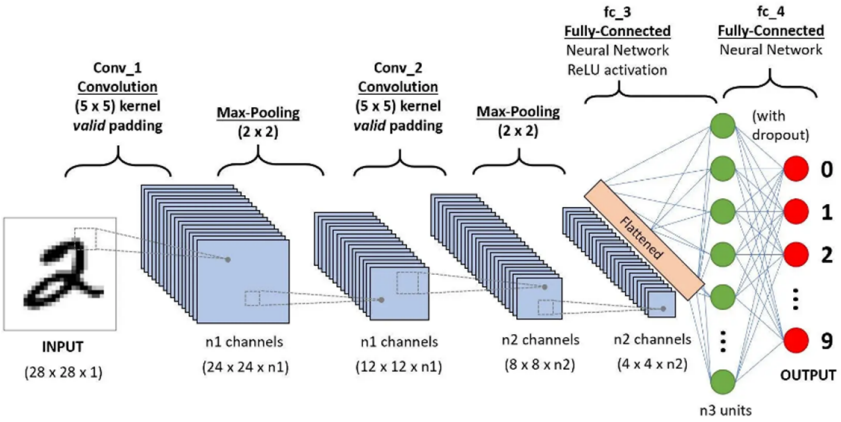

Figure 1: Handwritten number recognition pipeline

Loss Functions

Softmax

$$\text{Softmax}(x) = \frac{e^{x_i}}{\sum_i e^{x_i}} \tag{20}$$

Loss Functions

Softmax

$$\text{Softmax}(x) = \frac{e^{x_i}}{\sum_i e^{x_i}} \tag{20}$$

Sigmoid

$$\text{Sigmoid}(x) = \frac{1}{1 + e^{-x}} \tag{21}$$

Loss Functions

Softmax

$$\text{Softmax}(x) = \frac{e^{x_i}}{\sum_i e^{x_i}} \tag{20}$$

Sigmoid

$$\text{Sigmoid}(x) = \frac{1}{1 + e^{-x}} \tag{21}$$

tanh

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{22}$$

Loss Functions

Softmax

$$\text{Softmax}(x) = \frac{e^{x_i}}{\sum_i e^{x_i}} \tag{20}$$

Sigmoid

$$\text{Sigmoid}(x) = \frac{1}{1 + e^{-x}} \tag{21}$$

tanh

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{22}$$

ReLU

$$\text{ReLU}(x) = \max(0, x) \tag{23}$$

Optimizations

Methods

- Batch normalization

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}} \tag{24}$$

- Stochastic gradient descent

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta, x^{(i)}, y^{(i)}) \tag{25}$$

Classic Models

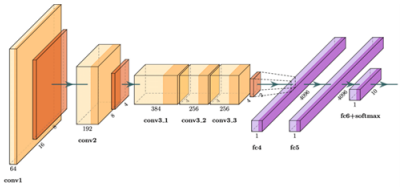

Figure 2: AlexNet

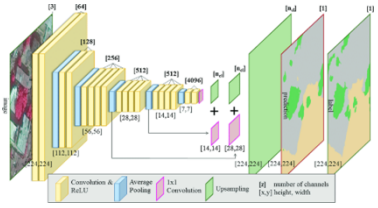

Figure 3: FCN

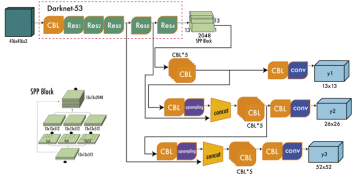

Figure 4: DarkNet

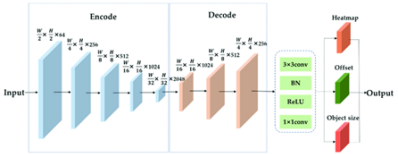

Figure 5: CenterNet

Content

R-CNN Series

R-CNN is an object detection model that uses selective search for region proposals, feeds these into a pre-trained CNN for feature extraction, and then classifies and refines the regions using SVMs and bounding box regressors.

It laid the foundation for deep learning-based object detection but was computationally intensive.

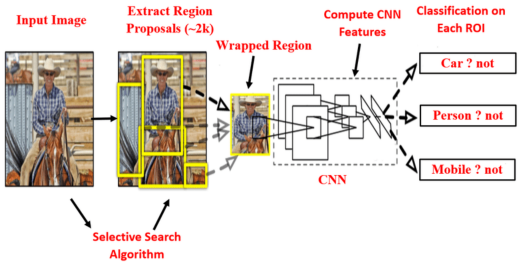

Figure 6: R-CNN

R-CNN series

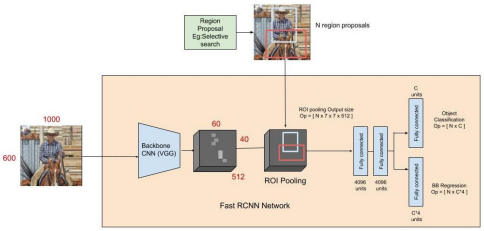

Figure 7: Fast R-CNN

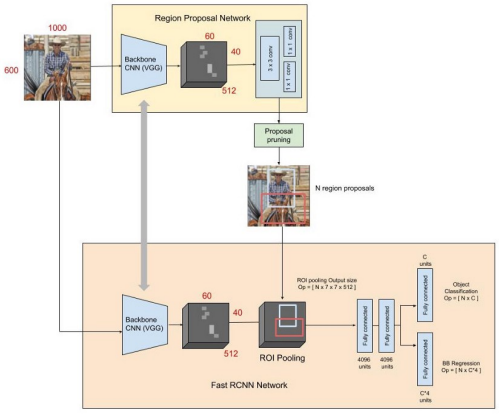

Figure 8: Faster R-CNN

You Only Look Once

Innovations

- Unified detection framework
- Real-time detection
- Global contextual prediction
- Spatially separated bounding boxes
- End-to-end training

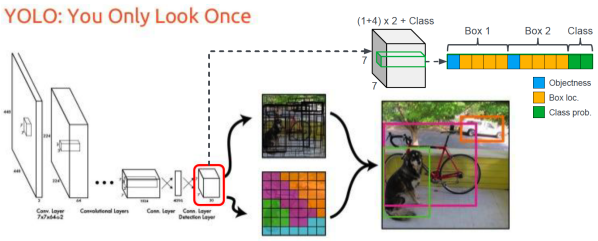

Figure 9: YOLOv1 architecture

Loss Function Improvement

YOLOv1

$$
\begin{aligned}
\text{Loss} =& \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{\text{obj}} \left[\left(x_i^j - \hat{x}_i^j \right)^2 + \left(y_i^j - \hat{y}_i^j \right)^2 \right] \\
&+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{\text{obj}} \left[\left(\sqrt{w_i^j} - \sqrt{\hat{w}_i^j} \right)^2 + \left(\sqrt{h_i^j} - \sqrt{\hat{h}_i^j} \right)^2 \right] \\
&+ \sum_{i=0}^{s^2} \sum_{j=0}^{B} I_{ij}^{\text{obj}} \left(C_i^j - \hat{C}_i^j \right)^2 \\
&+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{\text{noobj}} \left(C_i^j - \hat{C}_i^j \right)^2 \\
&+ \sum_{i=0}^{s^2} I_{ij}^{\text{obj}} \sum_{c \in \text{ classes}} \left(P_i^j(c) - \hat{C}_i^j(c) \right)^2
\end{aligned}
\tag{26}
$$

Loss Function Improvement

YOLOv3

$$
\begin{aligned}
\text{Loss} =& \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{\text{obj}} \left[\left(x_i^j - \hat{x}_i^j \right)^2 + \left(y_i^j - \hat{y}_i^j \right)^2 \right] \\
&+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{\text{obj}} \left[\left(\sqrt{w_i^j} - \sqrt{\hat{w}_i^j} \right)^2 + \left(\sqrt{h_i^j} - \sqrt{\hat{h}_i^j} \right)^2 \right] \\
&- \sum_{i=0}^{s^2} \sum_{j=0}^{B} I_{ij}^{\text{obj}} \left[\hat{C}_i^j \log \left(C_i^j \right) + \left(1 - \hat{C}_i^j \right) \log \left(1 - C_i^j \right) \right] \\
&- \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{\text{noobj}} \left[\hat{C}_i^j \log \left(C_i^j \right) + \left(1 - \hat{C}_i^j \right) \log \left(1 - C_i^j \right) \right] \\
&- \sum_{i=0}^{s^2} I_{ij}^{\text{obj}} \sum_{c \in \text{ classes}} \left[\hat{P}_i^j \log \left(P_i^j \right) + \left(1 - \hat{P}_i^j \right) \log \left(1 - P_i^j \right) \right]
\end{aligned}
\tag{27}
$$

Masterpieces

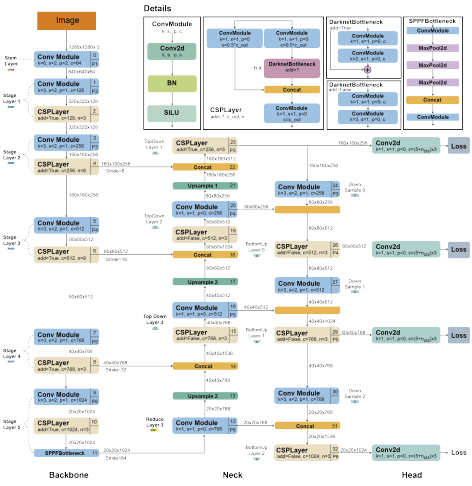

Figure 10: YOLOv5 architecture

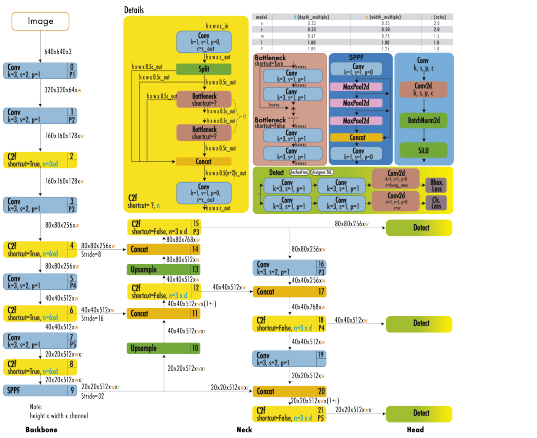

Figure 11: YOLOv8 architecture

Applications

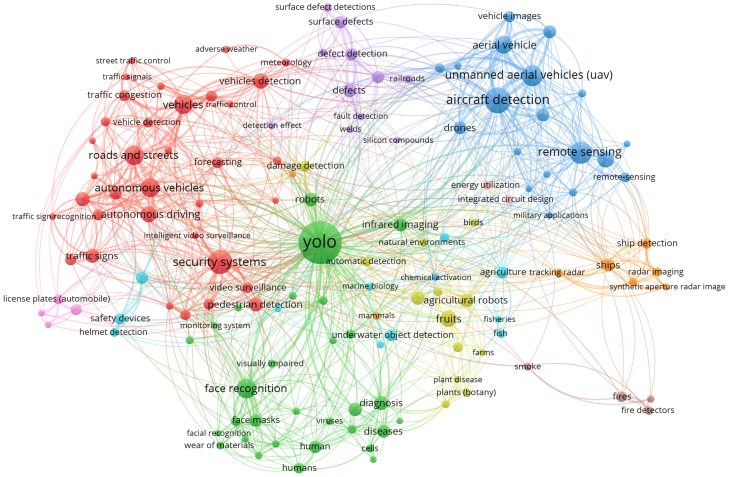

Figure 12: Applications on YOLO

Deep Snake

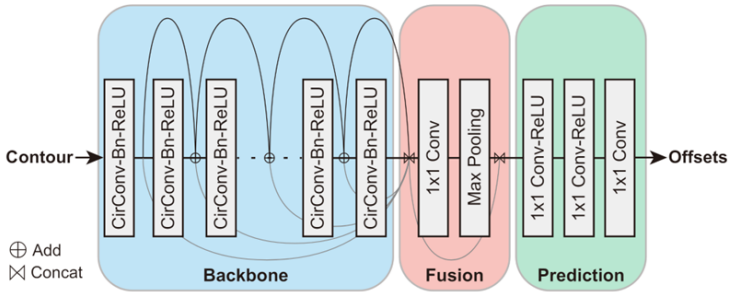

Figure 13: Deep Snake architecture

Content

Inprovement: Efficient Multi-scale Attention

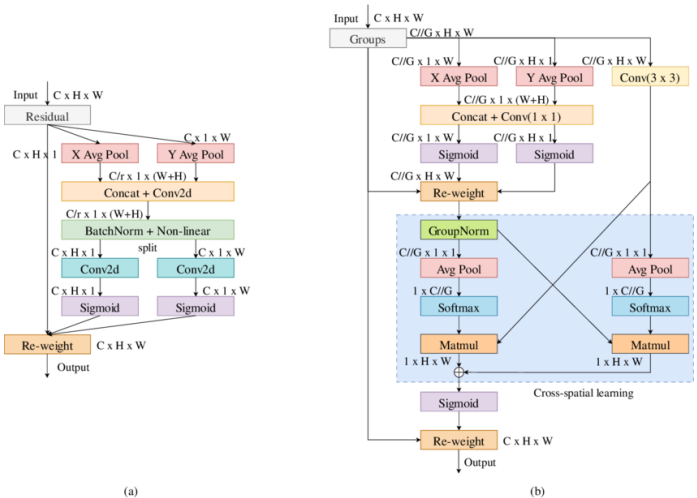

Figure 14: Structure of CA and EMA

Inprovement: Efficient Multi-scale Attention

Pooling output comparison

In coordinate attention(CA), the 1D global average-pooling for encoding information along the horizontal dimension direction in C at height H can be denoted by

$$z_c^H(H) = \frac{1}{W} \sum_{0 \le i \le W} x_c(H, i) \quad (28)$$

The pooling output in C at width W can be formulated as

$$z_c^W(W) = \frac{1}{H} \sum_{0 \le j \le H} x_c(j, W) \quad (29)$$

Efficient multi-scale attention(EMA) employs 2D global average pooling to aggregate cross-spatial information, encoding it into the 1×1 branch output. Outputs of the least branch will be transformed to the correspond dimension shape before the joint activation mechanism of channel features, i.e., $\mathbb{R}_1^{1 \times C//G} \times \mathbb{R}_3^{C//G \times HW}$. The 2D global pooling operation is

$$z_c = \frac{1}{H \times W} \sum_j^H \sum_i^W x_c(i, j) \quad (30)$$

Inprovement: Wise-IoU

In YOLOv2, we have bounding box regression(BBR) loss expressed as

$$\mathcal{L}(\vec{B}, \vec{B_{gt}}) = \left|\left|\vec{B} - \vec{B_{gt}}\right|\right| \tag{31}$$

But it cannot shield the inference of the size of the bounding box. So we come up with intersection over union(IoU)

$$\mathcal{L}_{\text{IoU}} = 1 - \text{IoU} = 1 - \frac{W_i H_i}{S_i} \tag{32}$$

In order to prevent circumstances like $W_i = 0$ or H_i, we add penalty term $\mathcal{R}_i$

$$\mathcal{L}_i = \mathcal{L}_{\text{IoU}} + \mathcal{R}_i \tag{33}$$

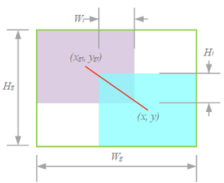

Figure 15: An IoU example

Inprovement: Wise-IoU

Distance-IoU

Consider normalized distance between central points of two bounding boxes

$$\mathcal{R}_{\text{DIoU}} = \frac{(x - x_{gt})^2 + (y - y_{gt})^2}{W_g^2 + H_g^2} \tag{34}$$

Inprovement: Wise-IoU

Distance-IoU

Consider normalized distance between central points of two bounding boxes

$$\mathcal{R}_{\text{DIoU}} = \frac{(x - x_{gt})^2 + (y - y_{gt})^2}{W_g^2 + H_g^2} \tag{34}$$

Complete-IoU

Add the consideration of aspect ratio

$$\mathcal{R}_{\text{CIoU}} = \mathcal{R}_{\text{DIoU}} + \alpha v, \alpha = \frac{v}{\mathcal{L}_{\text{IoU}} + v} \tag{35}$$

Inprovement: Wise-IoU

Efficient-IoU

Increase the punishment for distance metric

$$\mathcal{R}_{\text{EIoU}} = \mathcal{R}_{\text{DIoU}} + \frac{(x - x_{gt})^2}{W_g^2} + \frac{(y - y_{gt})^2}{H_g^2} \tag{36}$$

Inprovement: Wise-IoU

Efficient-IoU

Increase the punishment for distance metric

$$\mathcal{R}_{\text{EIoU}} = \mathcal{R}_{\text{DIoU}} + \frac{(x - x_{gt})^2}{W_g^2} + \frac{(y - y_{gt})^2}{H_g^2} \tag{36}$$

Scylla-IoU

We define angle cost as

$$\Lambda = \sin\left(2\sin^{-1}\frac{\min\left(|x - x_{gt}|, |y - y_{gt}|\right)}{\sqrt{(x - x_{gt})^2 + (y - y_{gt})^2} + \varepsilon}\right) \tag{37}$$

Then the distance cost comes to be

$$\Delta = \frac{1}{2}\sum_{t=w,h}\left(1 - e^{-\gamma\rho_t}\right), \gamma = 2 - \Lambda \tag{38}$$

Therefore, we have

$$\mathcal{R}_{\text{SIoU}} = \Delta + \Sigma \tag{39}$$

Inprovement: Wise-IoU

Wise-IoU

Based on SIoU and EIou, we reconsider low-quality examples and geometric factors, and construct Wise-IoU with 2 layers of attention mechanism

$$\mathcal{L}_{\text{WIoU}} = \mathcal{R}_{\text{WIoU}} \mathcal{L}_{\text{IoU}} \tag{40}$$

where

$$\mathcal{R}_{\text{WIoU}} = \exp\left(\frac{(x - x_{gt})^2 + (y - y_{gt})^2}{(W_g^2 + H_g^2)^*}\right) \tag{41}$$

$\mathcal{R}_{\text{WIoU}} \in [1, e)$ which will significantly amplify $\mathcal{L}_{\text{IoU}}$ of the ordinary-quality anchor box.
$\mathcal{L}_{\text{IoU}} \in [0, 1]$, which will significantly reduce $\mathcal{R}_{\text{WIoU}}$ of the high-quality anchor box and its focus on the distance between central points when the anchor box coincides well with the target box.
Superscript $*$ indicates W_g, H_g are detached from the computational graph.

Experiment: Where to add EMA?

	Precision(B) (%)	Recall(B) (%)	mAP50(B) (%)	mAP50-95(B) (%)
YOLOv8	**60.047**	68.864	66.999	56.038
YOLOv8+EMA(neck)	58.934	**70.791**	66.75	57.047
YOLOv8+EMA(backbone)	59.085	70.489	**68.187**	**57.669**

	Precision(M) (%)	Recall(M) (%)	mAP50(M) (%)	mAP50-95(M) (%)
YOLOv8	**61.783**	69.719	68.568	55.745
YOLOv8 +EMA(neck)	59.51	71.393	68.149	56.008
YOLOv8 +EMA(backbone)	60.575	**72.123**	**69.953**	**57.032**

Experiment: How's the combination?

	Precision(B) (%)	Recall(B) (%)	mAP50(B) (%)	mAP50-95(B) (%)
YOLOv8	**60.047**	68.864	66.999	56.038
YOLOv8+WIoU	57.03	76.804	66.26	52.148
YOLOv8+EMA	59.085	70.489	68.187	57.669
YOLOv8+EMA+WIoU	59.276	**74.3**	**69.169**	**58.039**

	Precision(M) (%)	Recall(M) (%)	mAP50(M) (%)	mAP50-95(M) (%)
YOLOv8	**61.783**	69.719	68.568	55.745
YOLOv8+WIoU	58.219	77.826	68.119	53.059
YOLOv8+EMA	60.575	72.123	69.953	57.032
YOLOv8+EMA+WIoU	60.184	**75.724**	**70.888**	**57.467**

Assessment: EMA and WIoU

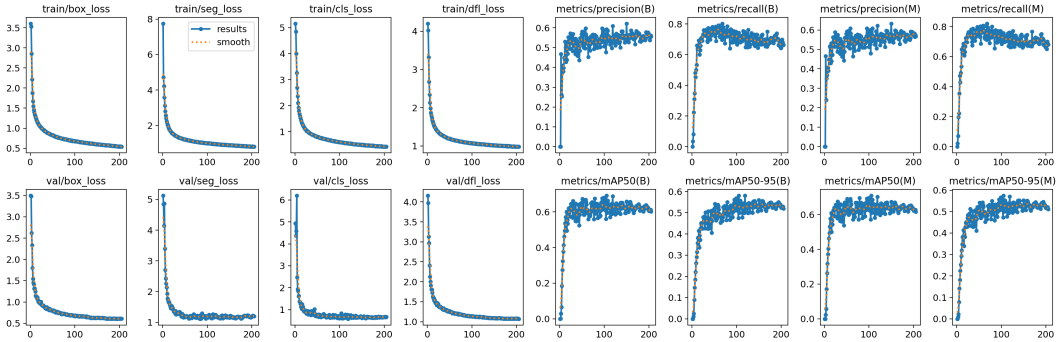

Figure 16: Results

Assessment: EMA and WIoU

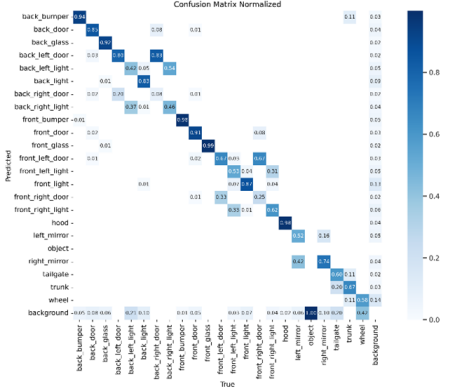

Figure 17: Original YOLOv8 confusion matrix

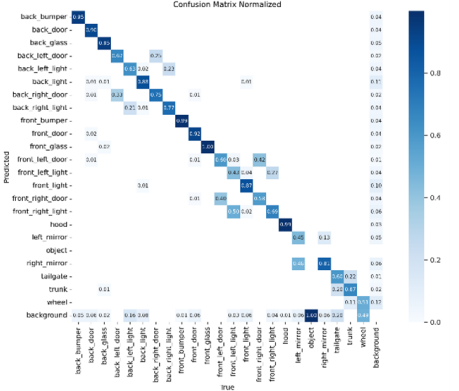

Figure 18: Improved YOLOv8 confusion matrix

Results: EMA and WIoU

Figure 19: Original labels

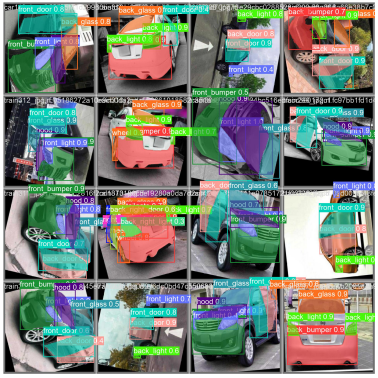

Figure 20: YOLOv8 with EMA and WIoU segmentation

Inprovement: Deep Layer Aggregation

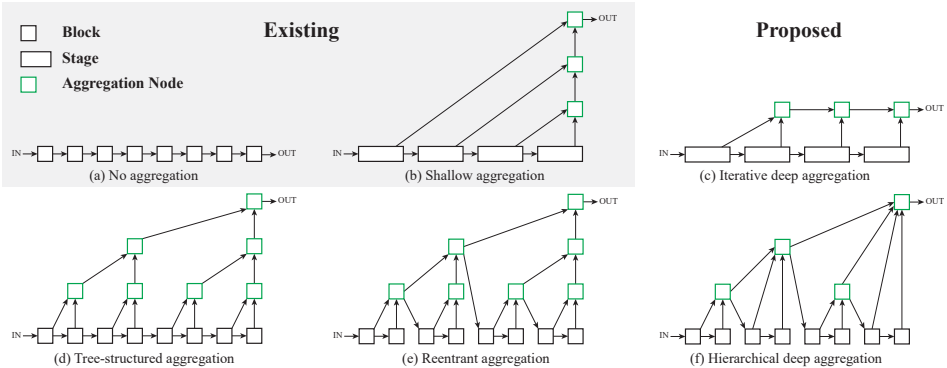

Figure 21: Different ways to aggregate layers

Inprovement: Deep Layer Aggregation

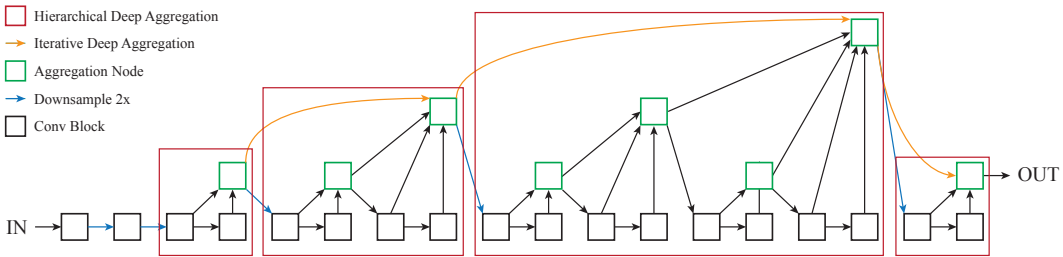

Figure 22: DLA network architecture

Experiment: Deep Snake

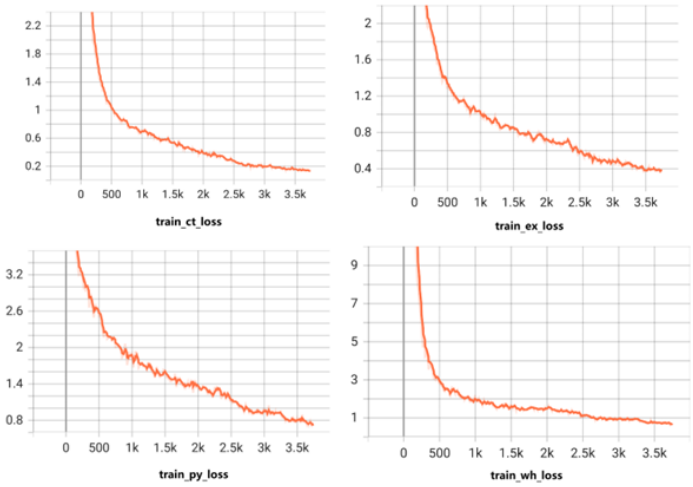

Figure 23: Loss during training

Experiment: Deep Snake

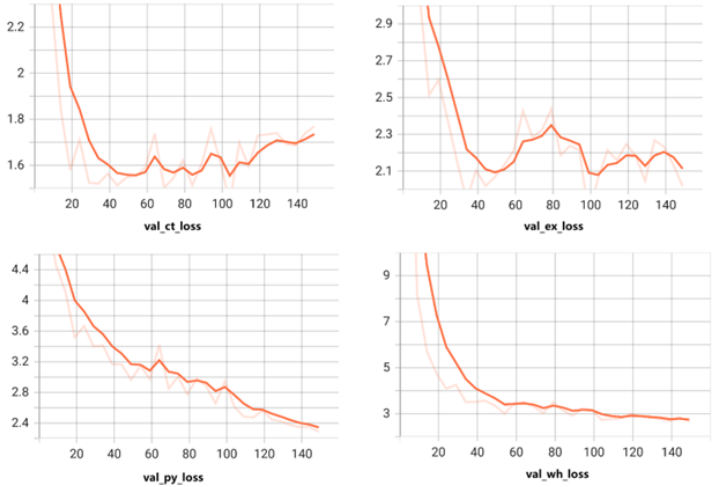

Figure 24: Loss during validation

Experiment: Deep Snake

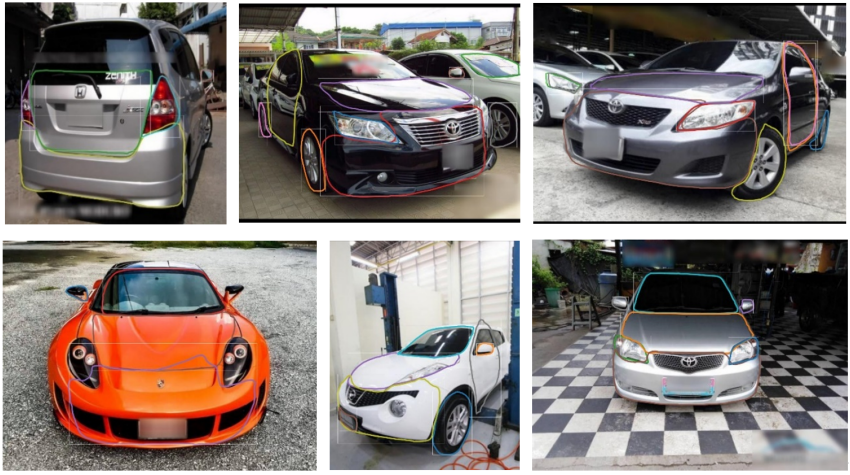

Figure 25: Sample pictures segmented by Deep Snake


NYU
THANK YOU