GCN and NAS in Semantic Segmentation

Speaker: Xia Li
Date: 7th, Apr, 2019
Outline

1. GCN in Semantic Segmentation
   1. A^2Net
   2. GloRe
   3. SGR
   4. GCU

2. NAS in Semantic Segmentation
   1. DPC
   2. Auto-DeepLab
1. GCN in Semantic Segmentation

**Characteristics of GCN**

1. Non-grid structure
2. Well-defined adjacency matrix $A$

**How to apply on semantic segmentation task?**

$$H^{(l+1)} = \sigma \left( A^{(l)} H^{(l)} W^{(l)} \right)$$

Or

$$H^{(l+1)} = \sigma \left( (I - \hat{A}^{(l)}) H^{(l)} W^{(l)} \right)$$

Where

$$\hat{A} = D^{-\frac{1}{2}} A D^{-\frac{1}{2}}, \quad D = \text{diag} \left( A \times \bar{1} \right)$$
1.0 Prerequisites

In general, 3 steps are needed:
1. Pixels to entities
2. GCN upon entities
3. Entities to pixels
1.0 Prerequisites

In general, 3 steps are needed:
1. Pixels to entities
2. GCN upon entities
3. Entities to pixels

Difficulties:
1. How to proj and re-proj?
2. How to define A?
1.1 A$^2$Net

For every spatial input location $i$

\[
  z_i = F_{\text{distr}} \left( G_{\text{gather}}(X), v_i \right)
\]

Proj

Re-proj

1.1 A^2Net

For every spatial input location $i$

$$z_i = F_{\text{distr}} \left( G_{\text{gather}}(X), v_i \right)$$

1.1 A^2Net

H: Height
W: Width
C: Number of the channels
K: Number of global descriptors
1.1 A^2Net

Using four $1 \times 1$ convolutions
1. Two construct bottleneck
2. Another two used for attention, Which can be merged as one.

Comparison with Nonlocal:
Reduce the complexity by using the multiplication law.
1.2 GloRe

First paper about GCN in Seg published on Arxiv.org.

1.2 GloRe

First paper about GCN in Seg published on Arxiv.org.

Using five $1 \times 1$ convolutions
To execute global reasoning.

Chen, Yunpeng, et al. "Graph-Based Global Reasoning Networks."
1.2 GloRe

Interaction Space

Step 1: information diffusion

Step 2: state update

How?

\[ O = HWC^2K \]

\[ \mathbf{Z} = G\mathbf{V}W_g = ((I - A_g)\mathbf{V})W_g \]

\[ A_g \text{ : learned parameter of a Conv1d} \]

\[ W_g \text{ : learned parameter of a Conv1d} \]

\[ O(\text{HWCK} + K^2C + C^2K) \]
1.2 GloRe

Table 3: Semantic segmentation results on Cityscapes validation set. ImageNet pre-trained ResNet-50 is used as the backbone CNN.

<table>
<thead>
<tr>
<th>FCN</th>
<th>multi-grid</th>
<th>+1 GloRe unit</th>
<th>+2 GloRe unit</th>
<th>mIoU</th>
<th>Δ mIoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>75.79%</td>
<td></td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>76.45%</td>
<td>0.66%</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>78.25%</td>
<td>2.46%</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>77.84%</td>
<td>2.05%</td>
</tr>
</tbody>
</table>

Table 4: Semantic segmentation results on Cityscapes test set. All networks are evaluated by the testing server. Our method is trained without using extra “coarse” training set.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>IoU cla.</th>
<th>iIoU cla.</th>
<th>IoU cat.</th>
<th>iIoU cat.</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeepLab-v2 [4]</td>
<td>ResNet101</td>
<td>70.4%</td>
<td>42.6%</td>
<td>86.4%</td>
<td>67.7%</td>
</tr>
<tr>
<td>PSPNet [36]</td>
<td>ResNet101</td>
<td>78.4%</td>
<td>56.7%</td>
<td>90.6%</td>
<td>78.6%</td>
</tr>
<tr>
<td>PSANet [37]</td>
<td>ResNet101</td>
<td>80.1%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DenseASPP [35]</td>
<td>ResNet101</td>
<td>80.6%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FCN + 1 GloRe unit</td>
<td>ResNet50</td>
<td>79.5%</td>
<td>60.3%</td>
<td>91.3%</td>
<td>81.5%</td>
</tr>
<tr>
<td>FCN + 1 GloRe unit</td>
<td>ResNet101</td>
<td>80.9%</td>
<td>62.2%</td>
<td>91.5%</td>
<td>82.1%</td>
</tr>
</tbody>
</table>

Figure 7: Visualization of the learned projection weights (best viewed in color). Red color denotes positive and green negative values, color brightness denotes magnitude.
1.3 SGR

1.3 SGR

\[
H_{n}^{ps} = \sum_{x_{i}} a_{x_{i}} x_{i} W_{n}^{ps}
\]

\[
a_{x_{i}} = \frac{\exp(W_{n}^{T} x_{i})}{\sum_{n \in N} \exp(W_{n}^{T} x_{i})}
\]

\[
\phi(X; W_{\phi}) \text{softmax} \left( \theta(X; W_{\theta}) \right)^{T} \text{softmax} \left( \rho(X; W_{\rho}) \right)
\]
Different definitions of A:
- **SGR** - pre-defined according to priors
- **GloRe** - learnable parameters

\[ H^g = \sigma(A^g BW^g) \]

\[ B = [\sigma(H^{ps}), S] \in \mathbb{R}^{M \times (D^c + K)} \]

**linguistic embedding**

\[ Z = GV W_g = ((I - A_g) V) W_g \]
1.3 SGR

\[
\alpha_{h^g} \rightarrow x_i = \frac{\exp(W_{s}^{\top} [h^g, x_i])}{\sum_{x_i} \exp(W_{s}^{\top} [h^g, x_i])}
\]

\[
X^{l+1} = \sigma(A^g H^g W^g) + X^l.
\]

\[
Z = GVW_g = ((I - A_g)V)W_g
\]

\[
\left[ \phi(X; W_\phi) \text{softmax} \left( \theta(X; W_\theta) \right) \right] \text{softmax} \left( \rho(X; W_\rho) \right)
\]
1.3 SGR

![Diagram](image.png)

Figure 3: Qualitative comparison results on Coco-stuff dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class acc.</th>
<th>acc.</th>
<th>mean IoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCN [31]</td>
<td>38.5</td>
<td>60.4</td>
<td>27.2</td>
</tr>
<tr>
<td>DeepLabv2 (ResNet-101)</td>
<td>45.5</td>
<td>65.1</td>
<td>34.4</td>
</tr>
<tr>
<td>DAG RNN + CRF [42]</td>
<td>42.8</td>
<td>63.0</td>
<td>31.2</td>
</tr>
<tr>
<td>OHE + DC + FCN [15]</td>
<td>45.8</td>
<td>66.6</td>
<td>34.3</td>
</tr>
<tr>
<td>DSSPN (ResNet-101) [27]</td>
<td>47.0</td>
<td>68.5</td>
<td>36.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Class acc.</th>
<th>acc.</th>
<th>mean IoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>SGR (w/o residual)</td>
<td>47.9</td>
<td>68.4</td>
<td>38.1</td>
</tr>
<tr>
<td>SGR (scene graph)</td>
<td>49.1</td>
<td>69.6</td>
<td>38.3</td>
</tr>
<tr>
<td>SGR (concurrency graph)</td>
<td>48.6</td>
<td>69.5</td>
<td>38.4</td>
</tr>
<tr>
<td>SGR (w/o mapping)</td>
<td>47.3</td>
<td>67.9</td>
<td>37.2</td>
</tr>
<tr>
<td>SGR (ConvBlock4)</td>
<td>47.6</td>
<td>68.3</td>
<td>37.5</td>
</tr>
<tr>
<td>Our SGR (ResNet-101)</td>
<td>49.3</td>
<td>69.9</td>
<td>38.7</td>
</tr>
<tr>
<td>Our SGR (ResNet-101 2-layer)</td>
<td>49.4</td>
<td>69.7</td>
<td>38.8</td>
</tr>
<tr>
<td>Our SGR (ResNet-101 Hybrid)</td>
<td>49.8</td>
<td>70.5</td>
<td>39.1</td>
</tr>
</tbody>
</table>

Table 1: Comparison on Coco-Stuff test set (%). All our models are based on ResNet-101.

<table>
<thead>
<tr>
<th>Method</th>
<th>mean IoU (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCN [31]</td>
<td>37.8</td>
</tr>
<tr>
<td>CRF-RNN [51]</td>
<td>39.3</td>
</tr>
<tr>
<td>ParseNet [30]</td>
<td>40.4</td>
</tr>
<tr>
<td>BoxSup [8]</td>
<td>40.5</td>
</tr>
<tr>
<td>HO CRF [1]</td>
<td>41.3</td>
</tr>
<tr>
<td>Piecewise [29]</td>
<td>43.3</td>
</tr>
<tr>
<td>VeryDeep [44]</td>
<td>44.5</td>
</tr>
<tr>
<td>DeepLab-v2 (ResNet-101)</td>
<td>45.7</td>
</tr>
<tr>
<td>RefineNet (Res152) [28]</td>
<td>47.3</td>
</tr>
<tr>
<td>Our SGR (ResNet-101)</td>
<td>50.8</td>
</tr>
<tr>
<td>Our SGR (Transfer convs)</td>
<td>51.3</td>
</tr>
<tr>
<td>Our SGR (Transfer SGR)</td>
<td><strong>52.5</strong></td>
</tr>
</tbody>
</table>

Table 2: Comparison on PASCAL-Context test set (%).

<table>
<thead>
<tr>
<th>Method</th>
<th>mean IoU pixel acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCN [31]</td>
<td>39.39 71.32</td>
</tr>
<tr>
<td>SegNet [2]</td>
<td>21.64 71.00</td>
</tr>
<tr>
<td>DilatedNet [47]</td>
<td>32.31 73.55</td>
</tr>
<tr>
<td>CascadeNet [52]</td>
<td>34.90 74.52</td>
</tr>
<tr>
<td>PSPNet (ResNet-101)DA_AL [50]</td>
<td>41.96 80.64</td>
</tr>
<tr>
<td>Conditional Softmax [38]</td>
<td>31.27 72.23</td>
</tr>
<tr>
<td>Word2Vec [10]</td>
<td>29.18 71.31</td>
</tr>
<tr>
<td>Joint-Cosine [49]</td>
<td>31.52 73.15</td>
</tr>
<tr>
<td>DeepLabv2 (ResNet-101)</td>
<td>38.97 79.01</td>
</tr>
<tr>
<td>DSSPN (ResNet-101) [27]</td>
<td>42.03 81.21</td>
</tr>
<tr>
<td><strong>Our SGR (ResNet-101)</strong></td>
<td><strong>44.32</strong> 81.43</td>
</tr>
</tbody>
</table>

Table 3: Comparison on the ADE20K val set [%] (“Conditional Softmax [38]”, “Word2Vec [10]” and “Joint-Cosine [49]” use VGG as backbone. We use “DeepLabv2 (ResNet-101) [6]” as baseline.)
1.4 GCU

Graph Projection

**GCU**

\[
q_{ij}^k = \frac{\exp \left( -\frac{\|x_{ij} - w_k\|^2}{2\sigma_k} \right)}{\sum_k \exp \left( -\frac{\|x_{ij} - w_k\|^2}{2\sigma_k} \right)}
\]

\[
z_k = \frac{z_k'}{\|z_k'\|^2}, \quad z_k' = \frac{1}{\sum_{ij} q_{ij}^k} \sum_{ij} q_{ij}^k (x_{ij} - w_k) / \sigma_k
\]

**A^2Net**

\[
q_{ij}^k = \frac{\exp \left( x_{ij}^T w_k \right)}{\sum_k \exp \left( x_{ij}^T w_k \right)}
\]

\[
z_k = \frac{1}{\sum_{ij} q_{ij}^k} \sum_{ij} q_{ij}^k x_{ij}
\]

\[
\phi(X; W_\phi) \text{softmax} \left( \theta(X; W_\theta)^T \right) \text{softmax} \left( \rho(X; W_\rho) \right)
\]

Graph Convolution

\[ \tilde{Z} = f(\mathcal{A}Z^T W_g) \]
\[ \mathcal{A} = Z^T Z \]

Different definitions of A:
- SGR - pre-defined according to priors
- GloRe - learnable parameters
- GCU – sample-independent

\[ Z = GVW_g = ((I - A_g)V)W_g \]

Graph Reprojection

\[ \tilde{X} = Q\tilde{Z}^T \]
\[ \left[ \phi(X; W_\phi) \text{softmax} (\theta(X; W_\theta))^T \right] \text{softmax} (\rho(X; W_\rho)) \]
Graph Convolution

\[ \tilde{Z} = f(\mathcal{A}Z^T W_g) \]
\[ \mathcal{A} = Z^T Z \]

\[ Z = G V W_g = ((I - A_g)V)W_g \]

Different definitions of A:
- SGR - pre-defined according to priors
- GloRe - learnable parameters
- GCU – sample-independent

Graph Reprojection

\[ \tilde{X} = Q\tilde{Z}^T \]

\[ \hat{X} = X \oplus \text{GCU}_{k_1}(X) \oplus ... \oplus \text{GCU}_{k_n}(X) \]
Graph Convolution

\[ \tilde{Z} = f(\mathcal{A}Z^T W_g) \]
\[ \mathcal{A} = Z^T Z \]

\[ A^{2}\text{Net} \]
\[ Z = G V W_g = ((I - A_g)V)W_g \]

Different definitions of A:
- SGR - pre-defined according to priors
- GloRe - learnable parameters
- GCU – sample-independent

Graph Reprojection

\[ \tilde{X} = Q \tilde{Z}^T \]

\[ A^{2}\text{Net} \]
\[ \left[ \phi(X; W_\phi) \text{softmax} \left( \theta(X; W_\theta) \right)^T \right] \text{softmax} \left( \rho(X; W_\rho) \right) \]

\[ \hat{X} = X \oplus \text{GCU}_{k_1}(X) \oplus \ldots \oplus \text{GCU}_{k_2}(X) \]

Figure 4: Visualization of the assignment matrix in GCU for semantic segmentation (with ResNet 50). From left to right: input image, pixel-to-vertex assignments with 2, 4, 8 and 32 vertices. Pixels with the same color are assigned to the same vertex. Vertices are colored consistently across images.
1.4 GCU

Figure 3: Visualization of segmentation results on ADE20K (with ResNet 50). Our method produces “smoother” maps—regions that are similar are likely to be labeled as the same category.

<table>
<thead>
<tr>
<th>Backbone</th>
<th>Method</th>
<th>PixAcc%</th>
<th>mIoU%</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG16 [42]</td>
<td>FCN-8s [12]</td>
<td>71.32</td>
<td>29.39</td>
</tr>
<tr>
<td></td>
<td>SegNet [41]</td>
<td>71.00</td>
<td>21.64</td>
</tr>
<tr>
<td></td>
<td>DilatedNet [17]</td>
<td>73.55</td>
<td>32.31</td>
</tr>
<tr>
<td></td>
<td>CascadeNet [37]</td>
<td>74.52</td>
<td>34.90</td>
</tr>
<tr>
<td>Res50 [38]</td>
<td>Dilated FCN</td>
<td>76.51</td>
<td>35.60</td>
</tr>
<tr>
<td></td>
<td>PSPNet [13]</td>
<td>80.76</td>
<td>42.78</td>
</tr>
<tr>
<td></td>
<td>EncNet [14]</td>
<td>79.73</td>
<td>41.11</td>
</tr>
<tr>
<td></td>
<td>GCU (ours)</td>
<td>79.51</td>
<td>42.60</td>
</tr>
<tr>
<td>Res101 [38]</td>
<td>RefineNet [19]</td>
<td>-</td>
<td>40.20</td>
</tr>
<tr>
<td></td>
<td>PSPNet [13]</td>
<td>81.39</td>
<td>43.29</td>
</tr>
<tr>
<td></td>
<td>EncNet [14]</td>
<td>81.69</td>
<td>44.65</td>
</tr>
<tr>
<td></td>
<td>GCU (ours)</td>
<td>81.19</td>
<td>44.81</td>
</tr>
</tbody>
</table>

Table 1: Results of semantic segmentation on ADE20K. mIoU scores within 0.5% of the best result are marked. With ResNet 50, our method improves Dilated FCN by 7%. With ResNet 101, our method outperforms PSPNet by 1.5%.
2. NAS in Semantic Segmentation

Key components of Network Architecture Search (NAS)

1. Search space
   1. Block level
   2. Cell level
2. Proxy task
   1. Low-resolution image
3. Search strategy
   1. Reinforcement learning
   2. Evolutionary algorithm
   3. Bayesian optimization
   4. Differentiable methods
2.1 DPC

1. Search space
   - Head
   - Cell level
2. Search strategy
   - Random search

Cell definition: \((X_i, OP_i, Y_i)\)

\[
X_i = \{F, Y_1, \ldots, Y_{i-1}\} \\
Y = \text{concat}(Y_1, Y_2, \ldots, Y_B)
\]

- Convolution with a \(1 \times 1\) kernel.

\(OP_i\)
- \(3 \times 3\) atrous separable convolution with rate \(r_h \times r_w\), where \(r_h\) and \(r_w\) \(\in\) \{1, 3, 6, 9, \ldots, 21\}.
- Average spatial pyramid pooling with grid size \(g_h \times g_w\), where \(g_h\) and \(g_w\) \(\in\) \{1, 2, 4, 8\}.

In total 81 operators

\[
\mathcal{B}! \times 81^{\mathcal{B}} \approx 4.2 \times 10^{11}
\]

2.1 DPC

3. Proxy task
   - Small backbone
   - Fix backbone
   - Early stopping

From 1 week to 90 minutes

Using 370 GPUs over one week
Explore 28k DPC architectures

Spearman’s rank correlation coefficient

\[ \rho = 0.46 \]
2.3 Auto-DeepLab

1. Search space
   1. Cell level \((I_1, I_2, O_1, O_2, C)\)
      For the \(l\)-th cell in the \(i\)-th block
      \(I_i \in \{H^{i-2}, H^{i-1}, \ldots, H_{i-1}^i\}\)
      
      \(O\) : element-wise addition
      2. Block level

2.3 Auto-DeepLab

3. Continuous relaxation
   1. Cell level
      \[ H_i^l = \sum_{H_j^l \in \mathcal{H}_l} O_{j \rightarrow i}(H_j^l) \]
      \[ \bar{O}_{j \rightarrow i}(H_j^l) = \sum_{o^k \in \mathcal{O}} \alpha_{j \rightarrow i}^k O^k(H_j^l) \]
      \[ \sum_{k=1}^{[\mathcal{O}]} \alpha_{j \rightarrow i}^k = 1 \quad \forall i, j \]
      \[ \alpha_{j \rightarrow i}^k \geq 0 \quad \forall i, j, k \]
   2. Block level
      \[ sH^l = \beta_{s \rightarrow \frac{s}{2}} H^{l-1} + sH^{l-2}; \alpha \]
      \[ + \beta_{s \rightarrow s} Cell(sH^{l-1}, sH^{l-2}; \alpha) \]
      \[ + \beta_{2s \rightarrow s} Cell(2sH^{l-1}, sH^{l-2}; \alpha) \]
      \[ \beta_{s \rightarrow \frac{s}{2}} + \beta_{s \rightarrow s} + \beta_{2s \rightarrow s} = 1 \quad \forall s, l \]
      \[ \beta_{s \rightarrow \frac{s}{2}} \geq 0 \quad \beta_{s \rightarrow s} \geq 0 \quad \beta_{2s \rightarrow s} \geq 0 \quad \forall s, l \]

4. Search strategy
   1. Update network weights \( w \) by \( \nabla_w \mathcal{L}_{trainA}(w, \alpha, \beta) \)
   2. Update architecture \( \alpha, \beta \) by \( \nabla_{\alpha, \beta} \mathcal{L}_{trainB}(w, \alpha, \beta) \)

5. Decoding discrete architecture
   1. Cell architecture
      - Argmax
   2. Block architecture
      - Viterbi algorithm
2.3 Auto-DeepLab

![Diagram](image)

<table>
<thead>
<tr>
<th>Method</th>
<th>ImageNet</th>
<th>COCO</th>
<th>mIoU (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRRN-A [60]</td>
<td></td>
<td></td>
<td>63.0</td>
</tr>
<tr>
<td>GridNet [77]</td>
<td></td>
<td></td>
<td>69.5</td>
</tr>
<tr>
<td>FRRN-B [60]</td>
<td></td>
<td></td>
<td>71.8</td>
</tr>
<tr>
<td>Auto-DeepLab-S</td>
<td></td>
<td></td>
<td>79.9</td>
</tr>
<tr>
<td>Auto-DeepLab-L</td>
<td></td>
<td></td>
<td>80.4</td>
</tr>
<tr>
<td>Auto-DeepLab-S</td>
<td>✓</td>
<td>✓</td>
<td>80.9</td>
</tr>
<tr>
<td>Auto-DeepLab-L</td>
<td>✓</td>
<td>✓</td>
<td>82.1</td>
</tr>
<tr>
<td>ResNet-38 [81]</td>
<td>✓</td>
<td>✓</td>
<td>80.6</td>
</tr>
<tr>
<td>PSPNet [85]</td>
<td>✓</td>
<td>✓</td>
<td>81.2</td>
</tr>
<tr>
<td>Mapillary [4]</td>
<td>✓</td>
<td>✓</td>
<td>82.0</td>
</tr>
<tr>
<td>DeepLab3+ [11]</td>
<td>✓</td>
<td>✓</td>
<td>82.1</td>
</tr>
<tr>
<td>DPC [60]</td>
<td>✓</td>
<td>✓</td>
<td>82.7</td>
</tr>
<tr>
<td>DRN_CRL, Course [90]</td>
<td>✓</td>
<td>✓</td>
<td>82.8</td>
</tr>
</tbody>
</table>

Table 4: Cityscapes test set results with multi-scale inputs during inference. **ImageNet**: Models pretrained on ImageNet. **Coarse**: Models exploit coarse annotations.

<table>
<thead>
<tr>
<th>Method</th>
<th>ImageNet</th>
<th>COCO</th>
<th>mIoU (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto-DeepLab-S</td>
<td>✓</td>
<td>✓</td>
<td>82.5</td>
</tr>
<tr>
<td>Auto-DeepLab-M</td>
<td>✓</td>
<td>✓</td>
<td>84.1</td>
</tr>
<tr>
<td>Auto-DeepLab-L</td>
<td>✓</td>
<td>✓</td>
<td>85.6</td>
</tr>
<tr>
<td>RefineNet [44]</td>
<td>✓</td>
<td>✓</td>
<td>84.2</td>
</tr>
<tr>
<td>ResNet-38 [81]</td>
<td>✓</td>
<td>✓</td>
<td>84.9</td>
</tr>
<tr>
<td>PSPNet [87]</td>
<td>✓</td>
<td>✓</td>
<td>85.4</td>
</tr>
<tr>
<td>DeepLab3+ [11]</td>
<td>✓</td>
<td>✓</td>
<td>87.8</td>
</tr>
<tr>
<td>MSCT [43]</td>
<td>✓</td>
<td>✓</td>
<td>88.0</td>
</tr>
</tbody>
</table>

Table 6: PASCAL VOC 2012 test set results. Our Auto-DeepLab-L attains comparable performance with many state-of-the-art models which are pretrained on both ImageNet and COCO datasets. We refer readers to the official leader-board for other state-of-the-art models.

<table>
<thead>
<tr>
<th>Method</th>
<th>ImageNet</th>
<th>COCO</th>
<th>mIoU (%)</th>
<th>Pixel-Acc (%)</th>
<th>Avg (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto-DeepLab-S</td>
<td>40.69</td>
<td></td>
<td>74.82</td>
<td>54.71</td>
<td></td>
</tr>
<tr>
<td>Auto-DeepLab-M</td>
<td>42.19</td>
<td></td>
<td>81.09</td>
<td>61.64</td>
<td></td>
</tr>
<tr>
<td>Auto-DeepLab-L</td>
<td>43.98</td>
<td></td>
<td>81.72</td>
<td>62.85</td>
<td></td>
</tr>
<tr>
<td>CascadeNet (VGG-16) [48]</td>
<td>34.90</td>
<td>✓</td>
<td>74.82</td>
<td>54.71</td>
<td></td>
</tr>
<tr>
<td>RefineNet (ResNet-152) [44]</td>
<td>40.70</td>
<td></td>
<td>81.09</td>
<td>61.64</td>
<td></td>
</tr>
<tr>
<td>UPerNet (ResNet-101) [52]</td>
<td>42.66</td>
<td></td>
<td>81.01</td>
<td>61.84</td>
<td></td>
</tr>
<tr>
<td>PSPNet (ResNet-152) [87]</td>
<td>43.51</td>
<td></td>
<td>81.38</td>
<td>62.45</td>
<td></td>
</tr>
<tr>
<td>PSPNet (ResNet-269) [87]</td>
<td>44.94</td>
<td></td>
<td>81.69</td>
<td>63.32</td>
<td></td>
</tr>
<tr>
<td>DeepLab3+ (Xception-65) [11]</td>
<td>45.65</td>
<td></td>
<td>82.52</td>
<td>64.09</td>
<td></td>
</tr>
</tbody>
</table>

Table 7: ADE20K validation set results. We employ multi-scale inputs during inference. ✓: Results are obtained from their up-to-date model zoo websites respectively. **ImageNet**: Models pretrained on ImageNet. Avg: Average of mIoU and Pixel-Accuracy.
6. References


Thanks

Speaker: Xia Li