Auto-DeepLab: Hierarchical Neural Architecture Search for Semantic Image Segmentation

Chenxi Liu1∗, Liang-Chieh Chen2, Florian Schroff2, Hartwig Adam2, Wei Hua2, Alan Yuille1, Li Fei-Fei3

1Johns Hopkins University 2Google 3Stanford University

Motivation

1. Existing works often focus on searching the repeatable cell structure.
2. *Neural Architecture Search (NAS) has successfully identified neural network architectures that exceed human designed ones on large-scale image classification problems.*

Motivation

1. Existing works often focus on searching the repeatable cell structure.
2. *Neural Architecture Search (NAS) has successfully identified neural network architectures that exceed human designed ones on large-scale image classification problems.*

Model	Auto Search				Task
	Cell	Network	Dataset	Days	
ResNet [25]	✗	✗	-	-	Cls
DenseNet [31]	✗	✗	-	-	Cls
DeepLabv3+ [11]	✗	✗	-	-	Seg
NASNet [92]	✓	✗	CIFAR-10	2000	Cls
AmoebaNet [61]	✓	✗	CIFAR-10	2000	Cls
PNASNet [47]	✓	✗	CIFAR-10	150	Cls
DARTS [49]	✓	✗	CIFAR-10	4	Cls
DPC [6]	✓	✗	Cityscapes	2600	Seg
Auto-DeepLab	✓	✓	Cityscapes	3	Seg

Table 1: Comparing our work against other CNN architectures with two-level hierarchy. The main differences include: (1) we directly search CNN architecture for semantic segmentation, (2) we search the network level architecture as well as the cell level one, and (3) our efficient search only requires 3 P100 GPU days.

contribution

1. One of the first attempts to extend NAS beyond image classification to dense image prediction
2. Propose a network level architecture search space that augments and complements the much-studied cell level one, and consider the more challenging joint search of network level and cell level architectures.
3. Develop a differentiable, continuous formulation that conducts the two-level hierarchical architecture search efficiently in 3 GPU days
4. Without ImageNet pretraining, the model significantly outperforms many popular methods

Related Work

1. Semantic Image Segmentation
    1. Multi-scale context module
    2. Neural network design
2. Neural Architecture Search Method
    1. reinforcement learning
    2. evolutionary algorithms (EA)
3. Neural Architecture Search Space
    1. searching the repeatable cell structure

Architecture Search Space

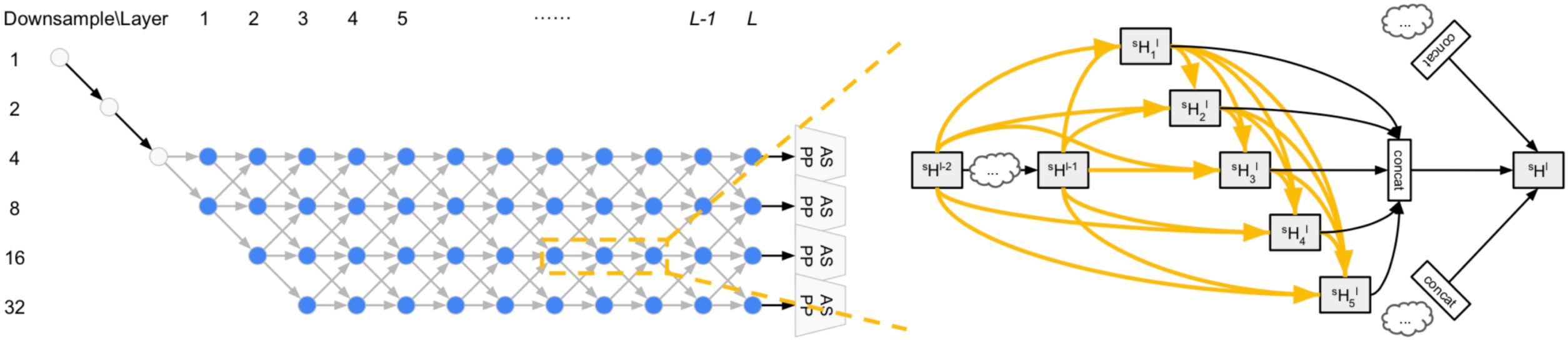
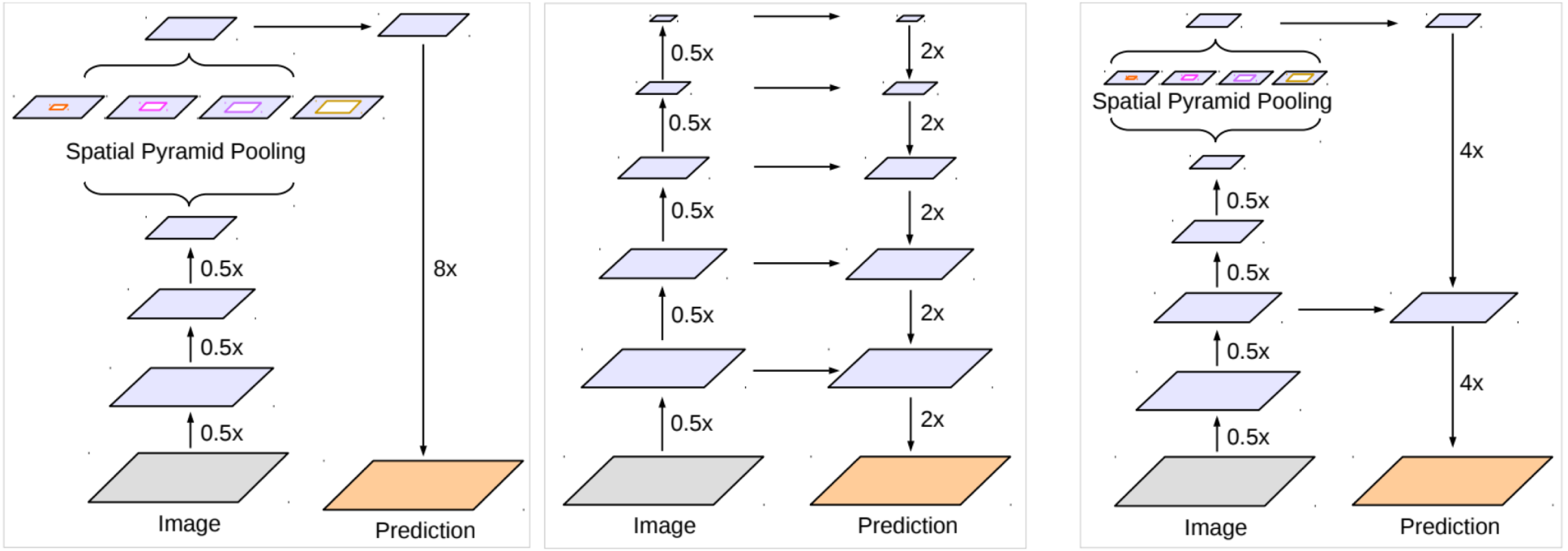
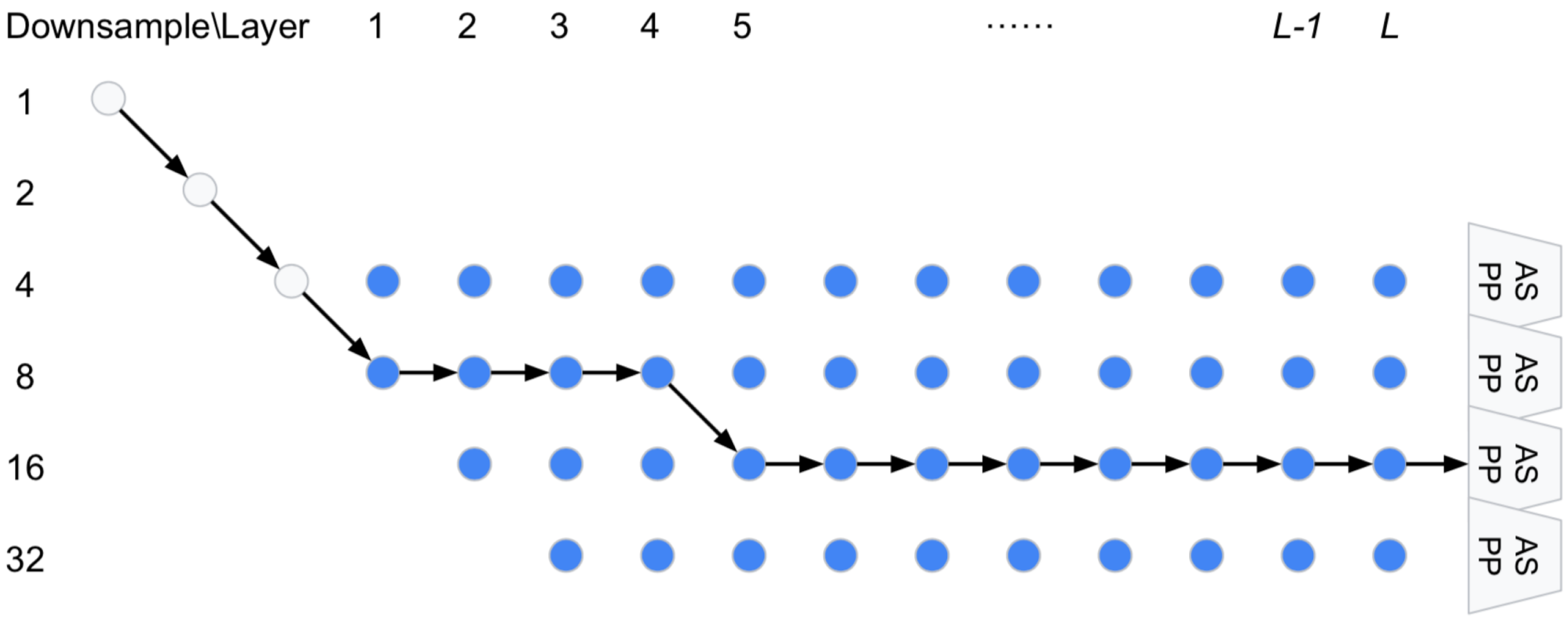


Figure 1: *Left:* Our network level search space with $L = 12$. Gray nodes represent the fixed "stem" layers, and a path along the blue nodes represents a candidate network level architecture. *Right:* During the search, each cell is a densely connected structure as described in Sec. 4.1.1. Every yellow arrow is associated with the set of values $\alpha_{j\to i}$. The three arrows after `concat` are associated with $\beta^l_{\frac{s}{2}\to s}, \beta^l_{s\to s}, \beta^l_{2s\to s}$ respectively, as described in Sec. 4.1.2. Best viewed in color.

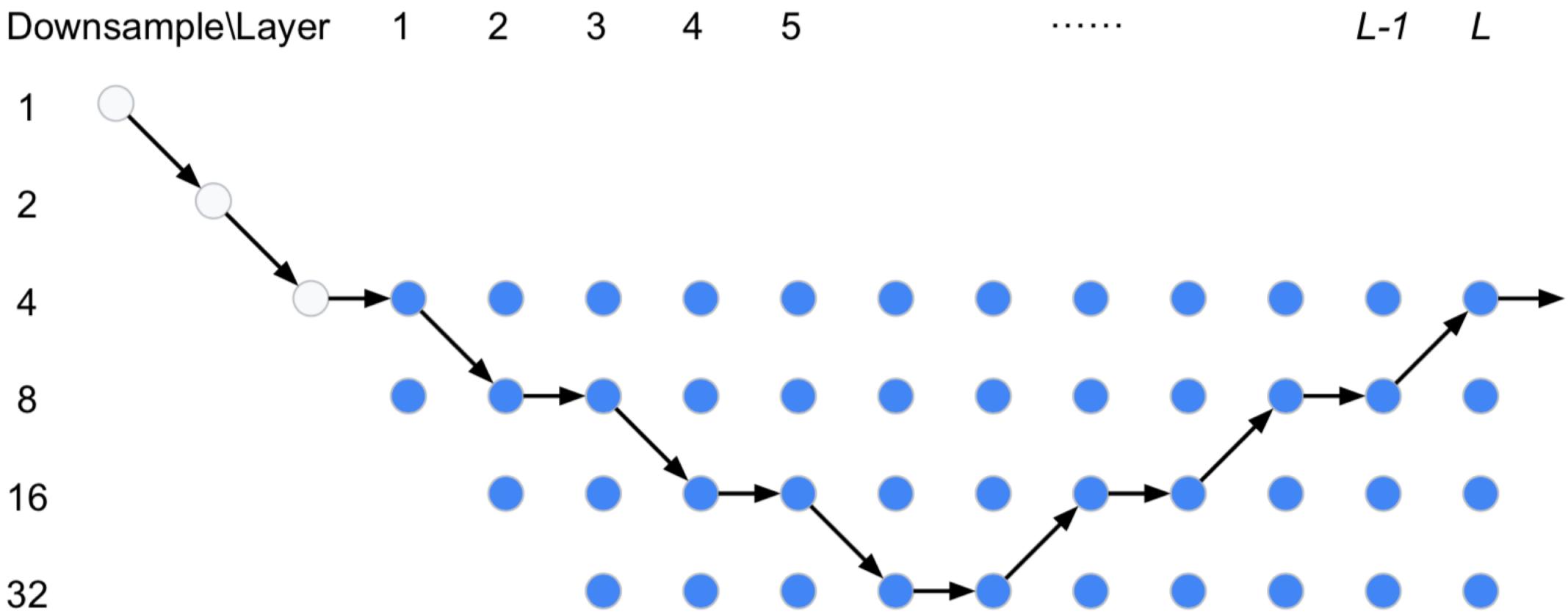


(a) Spatial Pyramid Pooling (b) Encoder-Decoder (c) Encoder-Decoder with Atrous Conv

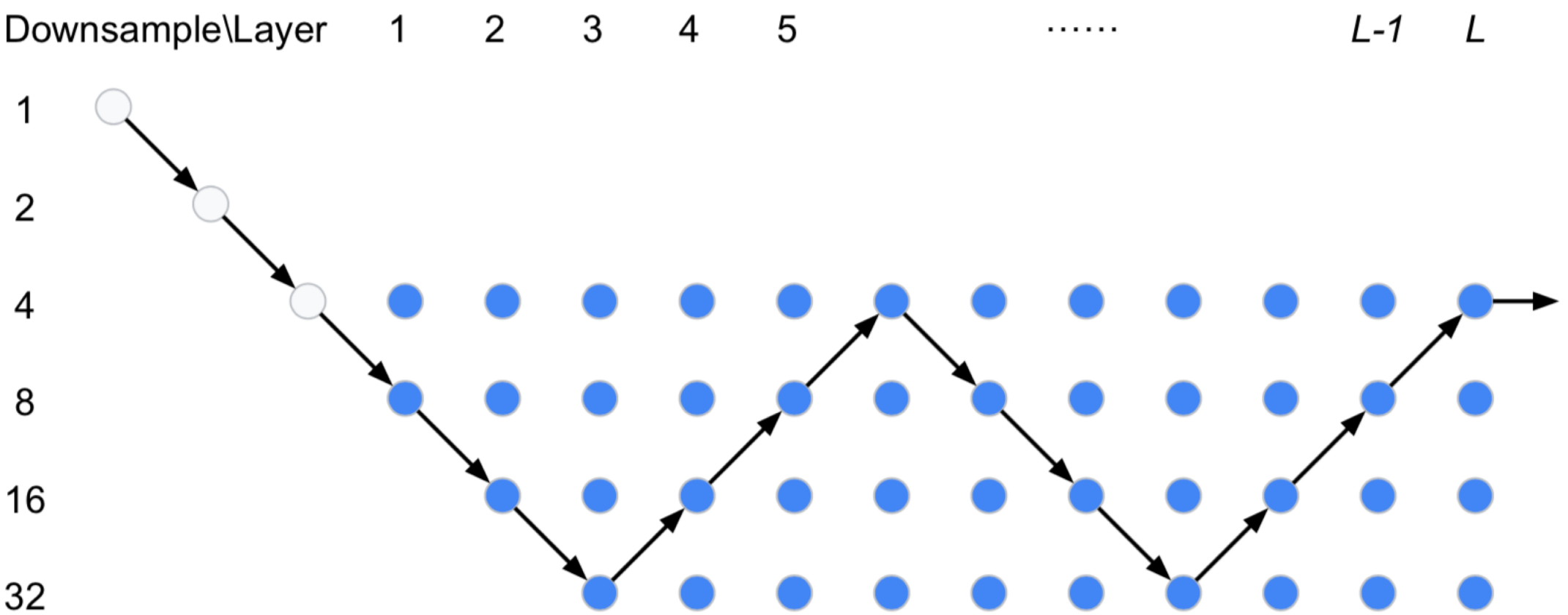
Fig. 1. We improve DeepLabv3, which employs the spatial pyramid pooling module (a), with the encoder-decoder structure (b). The proposed model, DeepLabv3+, contains rich semantic information from the encoder module, while the detailed object boundaries are recovered by the simple yet effective decoder module. The encoder module allows us to extract features at an arbitrary resolution by applying atrous convolution.



(a) Network level architecture used in DeepLabv3 [9].



(b) Network level architecture used in Conv-Deconv [56].

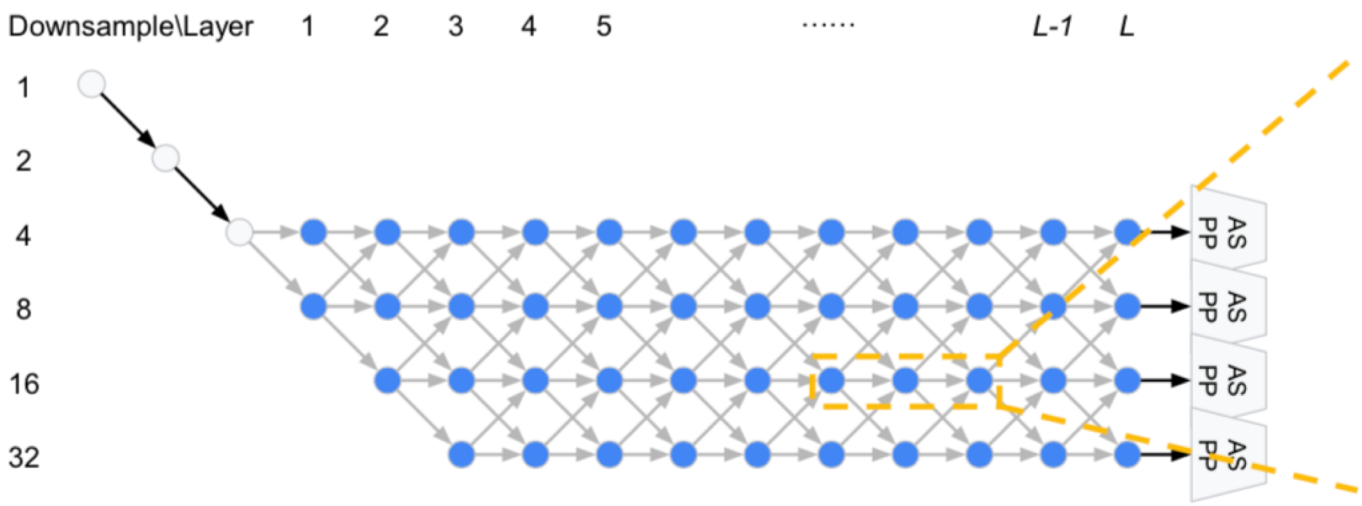


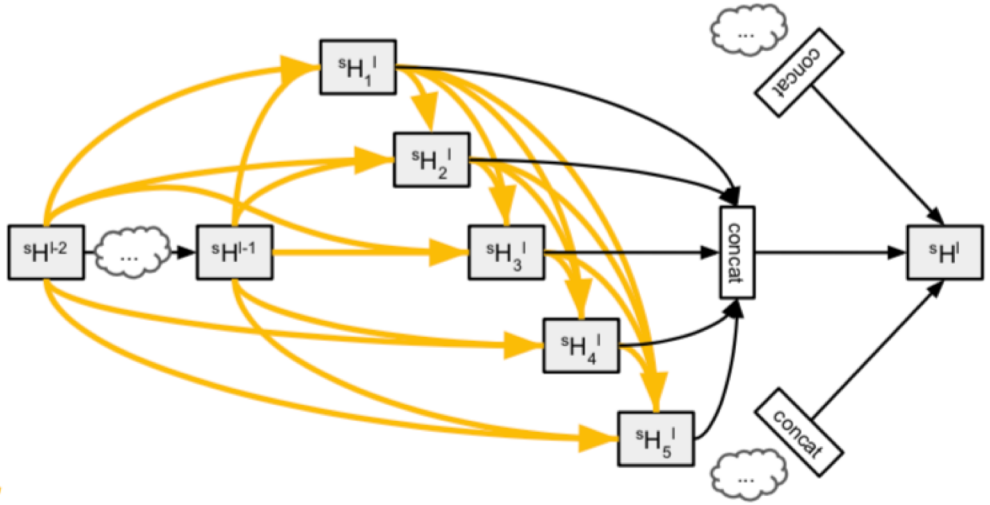
(c) Network level architecture used in Stacked Hourglass [55].

Methods

Two principles that are consistent:

- The spatial resolution of the next layer is either twice as large, or twice as small, or remains the same.
- The smallest spatial resolution is downsampled by 32.





$$H_i^l = \sum_{H_j^l \in \mathcal{I}_i^l} O_{j\to i}(H_j^l) \quad (1)$$

$$\bar{O}_{j\to i}(H_j^l) = \sum_{O^k \in \mathcal{O}} \alpha_{j\to i}^k O^k(H_j^l) \quad (2)$$

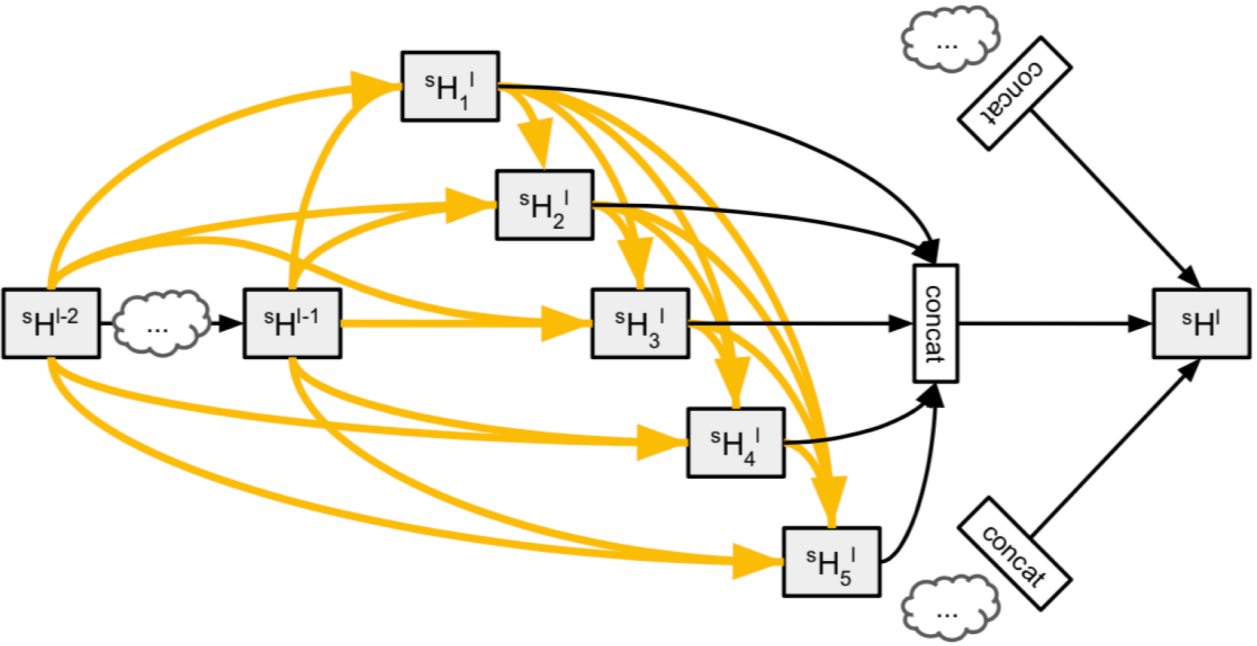
$$\sum_{k=1}^{|\mathcal{O}|} \alpha_{j\to i}^k = 1 \qquad \forall i, j \quad (3)$$

$$\alpha_{j\to i}^k \geq 0 \qquad \forall i, j, k \quad (4)$$

$$H^l = \text{Cell}(H^{l-1}, H^{l-2}; \alpha) \quad (5)$$

Implementation details

4 hidden states {4Hl, 8Hl, 16Hl, 32Hl}



$$
\begin{aligned}
{}^{s}H^{l} = & \beta^{l}_{\frac{s}{2}\to s}\,\mathrm{Cell}({}^{\frac{s}{2}}H^{l-1}, {}^{s}H^{l-2}; \alpha) \\
& + \beta^{l}_{s\to s}\,\mathrm{Cell}({}^{s}H^{l-1}, {}^{s}H^{l-2}; \alpha) \\
& + \beta^{l}_{2s\to s}\,\mathrm{Cell}({}^{2s}H^{l-1}, {}^{s}H^{l-2}; \alpha) \qquad (6)
\end{aligned}
$$

$$
\beta^{l}_{s\to\frac{s}{2}} + \beta^{l}_{s\to s} + \beta^{l}_{s\to 2s} = 1 \qquad \forall s, l \qquad (7)
$$

$$
\beta^{l}_{s\to\frac{s}{2}} \geq 0 \quad \beta^{l}_{s\to s} \geq 0 \quad \beta^{l}_{s\to 2s} \geq 0 \qquad \forall s, l \qquad (8)
$$

The optimization alternates between:

1. Update network weights w by $\nabla_w L_{trainA}(w, \alpha, \beta)$
2. Update architecture α,β by $\nabla_{\alpha,\beta} L_{trainB}(w,\alpha,\beta)$

The optimization alternates between:

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

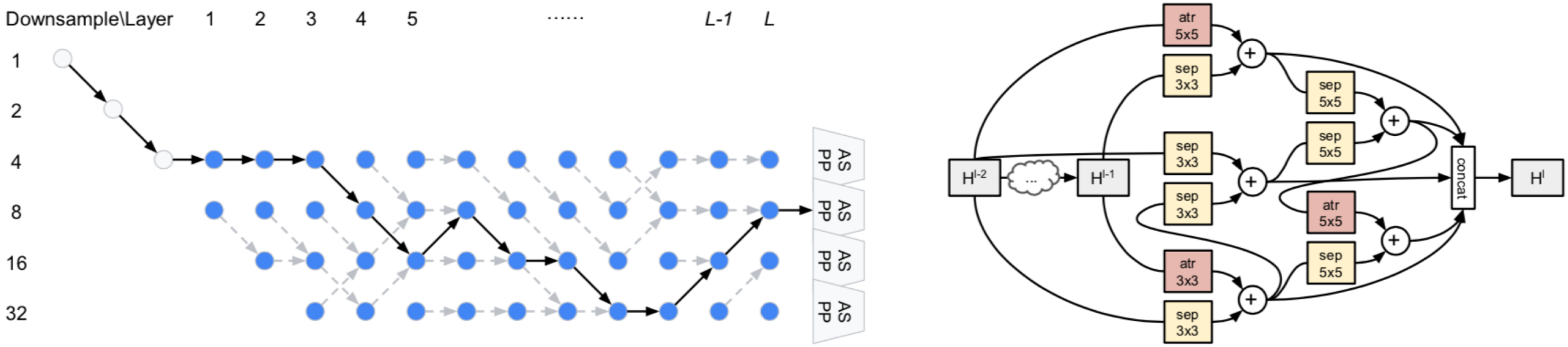


Figure 3: Best network and cell architecture found by our Hierarchical Neural Architecture Search. Gray dashed arrows show the connection with maximum β at each node. **atr:** atrous convolution. **sep:** depthwise-separable convolution.

Implementation details

1. Every blue node in Fig. 1 with downsample rate s has B × F × s output filters, where F is the filter multiplier controlling the model capacity, F = 8.
2. L = 12 layers.
3. B = 5 blocks in a cell.
4. batch size is 2.
5. SGD optimizer with momentum 0.9, cosine learning rate that decays from 0.025 to 0.001, and weight decay 0.0003.
6. Epochs = 40, optimize α, β after 20 epochs.

Semantic Segmentation Results

Method	ImageNet	F	Multi-Adds	Params	mIOU (%)
Auto-DeepLab-S		20	333.25B	10.15M	79.74
Auto-DeepLab-M		32	460.93B	21.62M	80.04
Auto-DeepLab-L		48	695.03B	44.42M	80.33
FRRN-A [60]		-	-	17.76M	65.7
FRRN-B [60]		-	-	24.78M	-
DeepLabv3+ [11]	✓	-	1551.05B	43.48M	79.55

Table 2: Cityscapes validation set results with different Auto-DeepLab model variants. F: the filter multiplier controlling the model capacity. All our models are trained from *scratch* and with *single-scale* input during inference.

Method	ImageNet	Coarse	mIOU (%)
FRRN-A [60]			63.0
GridNet [17]			69.5
FRRN-B [60]			71.8
Auto-DeepLab-S			79.9
Auto-DeepLab-L			80.4
Auto-DeepLab-S		✓	80.9
Auto-DeepLab-L		✓	82.1
ResNet-38 [81]	✓	✓	80.6
PSPNet [87]	✓	✓	81.2
Mapillary [4]	✓	✓	82.0
DeepLabv3+ [11]	✓	✓	82.1
DPC [6]	✓	✓	82.7
DRN_CRL_Coarse [90]	✓	✓	82.8

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

Method	ImageNet	COCO	mIOU (%)
Auto-DeepLab-S		✓	82.5
Auto-DeepLab-M		✓	84.1
Auto-DeepLab-L		✓	85.6
RefineNet [44]	✓	✓	84.2
ResNet-38 [81]	✓	✓	84.9
PSPNet [87]	✓	✓	85.4
DeepLabv3+ [11]	✓	✓	87.8
MSCI [43]	✓	✓	88.0

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.

Method	**ImageNet**	mIOU (%)	Pixel-Acc (%)	Avg (%)
Auto-DeepLab-S		40.69	80.60	60.65
Auto-DeepLab-M		42.19	81.09	61.64
Auto-DeepLab-L		43.98	81.72	62.85
CascadeNet (VGG-16) [89]	✓	34.90	74.52	54.71
RefineNet (ResNet-152) [44]	✓	40.70	-	-
UPerNet (ResNet-101) [82] †	✓	42.66	81.01	61.84
PSPNet (ResNet-152) [87]	✓	43.51	81.38	62.45
PSPNet (ResNet-269) [87]	✓	44.94	81.69	63.32
DeepLabv3+ (Xception-65) [11] †	✓	45.65	82.52	64.09

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.

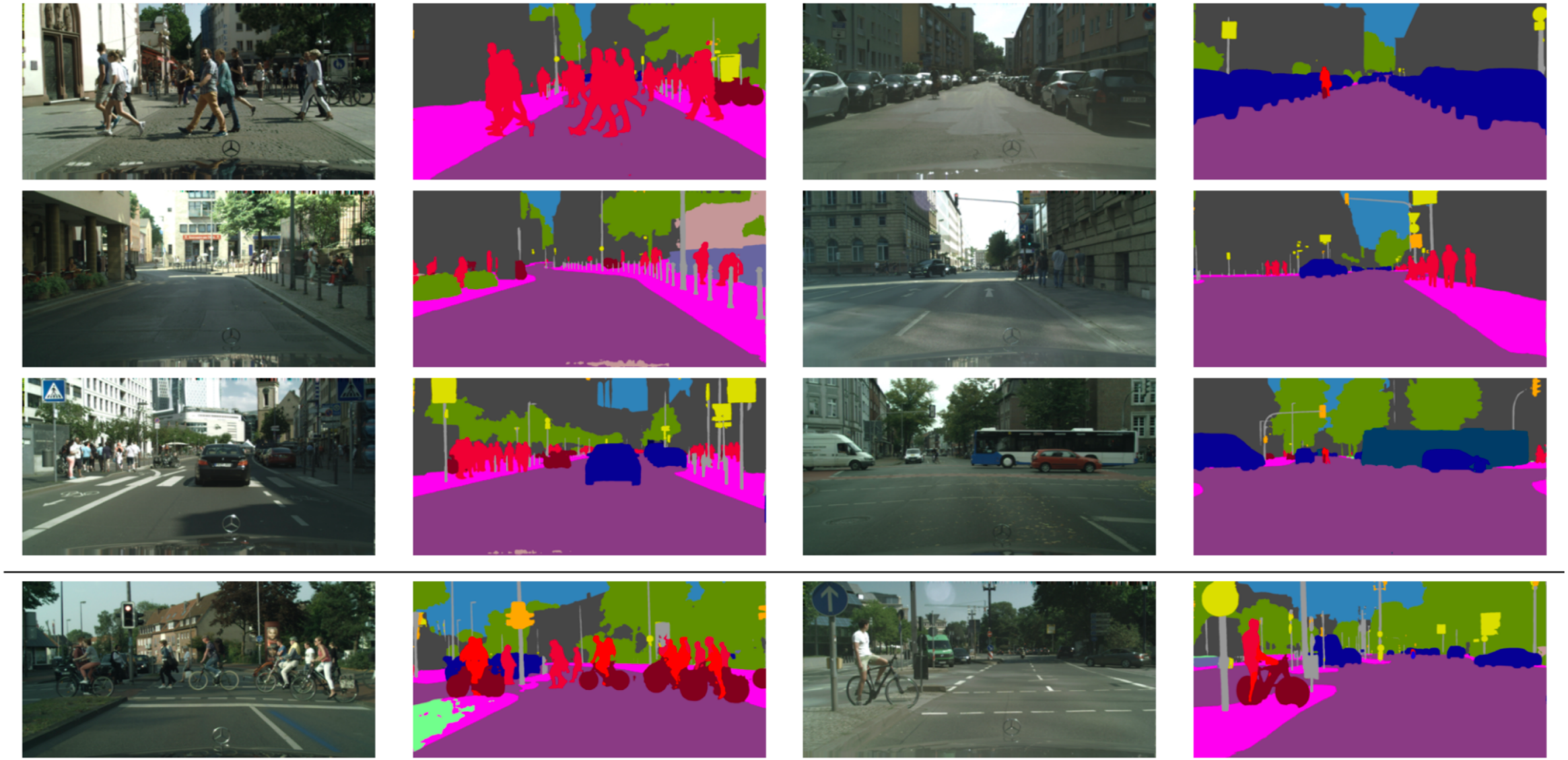


Figure 5: Visualization results on Cityscapes *val* set. Our failure mode is shown in the last row where our model confuses with some difficult semantic classes such as person and rider.

Figure 6: Visualization results on ADE20K *validation* set. Our failure mode is shown in the last row where our model could not segment very fine-grained objects (*e.g.*, chair legs) and confuse with some difficult semantic classes (*e.g.*, floor and rug).

Review

1. Review this work
    1. Classification → Segmentaion
    2. Enlarge the exploring space
    3. Long range context information

2. Application in my work
    1. general tendency to downsample in the first 3/4 layers and upsample in the last 1/4 layers
    2. search best scale for different level feature

Thanks for your attention!