动态网络及其在场景分割中的应用

Yanwei LI

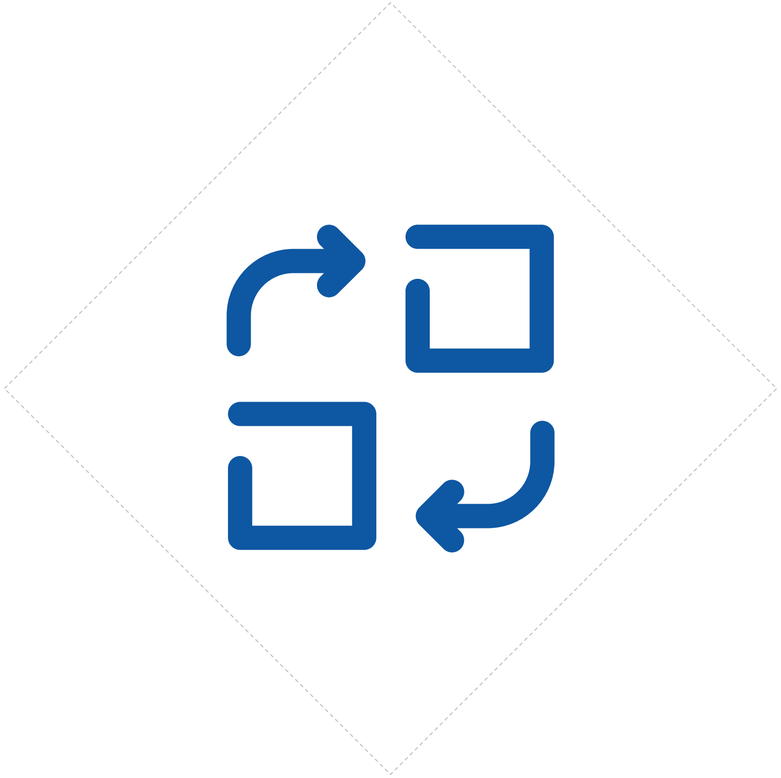
PPT

http://yanwei-li.com

分享提纲

1. 动态网络简介

- 什么是动态网络?
- 为什么需要动态网络?
- 和其他任务的区别?



2. 相关领域研究进展

- 动态网络研究进展
- 场景分割研究进展

3. 动态网络与场景分割

- 用于场景分割的动态网络
- 相关实验及可视化

4. 未来展望

- 还有哪些不足?
- 可以从哪方面改进?

动态网络简介

动态网络是什么?

这里的动态（dynamic）是相对于传统的固定结构而言，指**网络结构或部件（如卷积参数）根据输入数据自适应地进行调整**，以实现data-dependent的前向推断过程。

为什么需要动态网络?

- **更适合：**使网络适应输入图像不同的分布
- **更精确：**根据输入调节以获得更好的性能
- **更高效：**自适应地省去不必要的计算

动态网络简介

动态网络是什么?

这里的动态（dynamic）是相对于传统的固定结构而言，指**网络结构或部件（如卷积参数）根据输入数据自适应地进行调整**，以实现data-dependent的前向推断过程。

和其他任务的区别?

- **手工设计**：根据经验设计出更优的固定连接
- **结构搜索**：自动搜索出给定空间中最优的连接
- **动态网络**：根据输入自适应地调整网络连接

为什么需要动态网络?

- **更适合**：使网络适应输入图像不同的分布
- **更精确**：根据输入调节以获得更好的性能
- **更高效**：自适应地省去不必要的计算

动态网络简介

动态网络是什么?

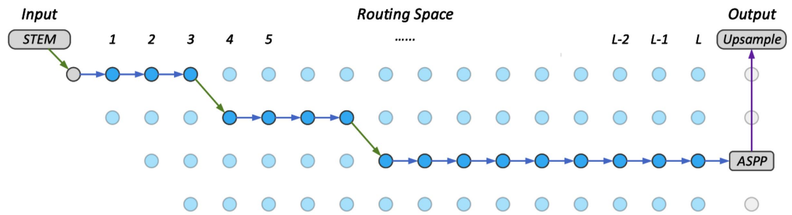
这里的动态（dynamic）是相对于传统的固定结构而言，指**网络结构或部件（如卷积参数）根据输入数据自适应地进行调整**，以实现data-dependent的前向推断过程。

为什么需要动态网络?

- **更适合：**使网络适应输入图像不同的分布
- **更精确：**根据输入调节以获得更好的性能
- **更高效：**自适应地省去不必要的计算

和其他任务的区别?

- **手工设计：**根据经验设计出更优的固定连接 → DeepLab V3
- **结构搜索：**自动搜索出给定空间中最优的连接
- **动态网络：**根据输入自适应地调整网络连接



Rethinking atrous convolution for semantic image segmentation. *arXiv:1706.05587*, 2017.

动态网络简介

动态网络是什么?

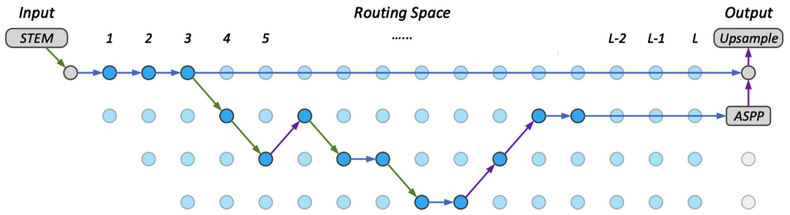
这里的动态（dynamic）是相对于传统的固定结构而言，指**网络结构或部件（如卷积参数）根据输入数据自适应地进行调整**，以实现data-dependent的前向推断过程。

为什么需要动态网络?

- **更适合：**使网络适应输入图像不同的分布
- **更精确：**根据输入调节以获得更好的性能
- **更高效：**自适应地省去不必要的计算

和其他任务的区别?

- **手工设计：**根据经验设计出更优的固定连接 → DeepLab V3
- **结构搜索：**自动搜索出给定空间中最优的连接 → Auto-DeepLab
- **动态网络：**根据输入自适应地调整网络连接



Rethinking atrous convolution for semantic image segmentation. *arXiv:1706.05587*, 2017.
Autodeeplab: Hierarchical neural architecture search for semantic image segmentation. In *CVPR*, 2019.

动态网络简介

动态网络是什么?

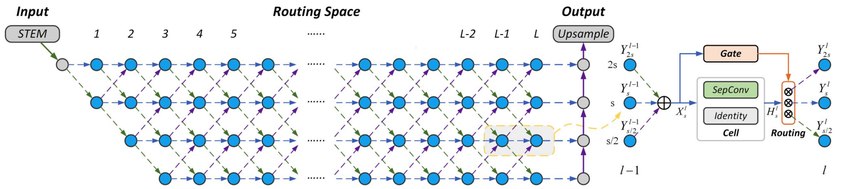
这里的动态（dynamic）是相对于传统的固定结构而言，指**网络结构或部件（如卷积参数）根据输入数据自适应地进行调整**，以实现data-dependent的前向推断过程。

为什么需要动态网络?

- **更适合：**使网络适应输入图像不同的分布
- **更精确：**根据输入调节以获得更好的性能
- **更高效：**自适应地省去不必要的计算

和其他任务的区别?

- **手工设计：**根据经验设计出更优的固定连接 → DeepLab V3
- **结构搜索：**自动搜索出给定空间中最优的连接 → Auto-DeepLab
- **动态网络：**根据输入自适应地调整网络连接 → Dynamic Routing



Rethinking atrous convolution for semantic image segmentation. *arXiv:1706.05587*, 2017.
Autodeeplab: Hierarchical neural architecture search for semantic image segmentation. In *CVPR*, 2019.
Learning Dynamic Routing for Semantic Segmentation. In *CVPR*, 2020.

动态网络简介

动态网络是什么?

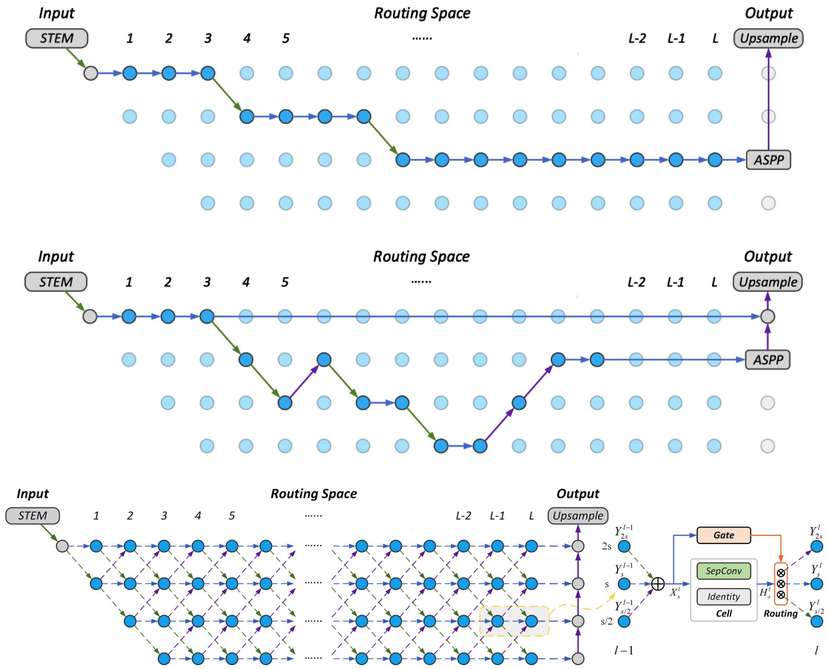
这里的动态（dynamic）是相对于传统的固定结构而言，指**网络结构或部件（如卷积参数）根据输入数据自适应地进行调整**，以实现data-dependent的前向推断过程。

为什么需要动态网络?

- **更适合：**使网络适应输入图像不同的分布
- **更精确：**根据输入调节以获得更好的性能
- **更高效：**自适应地省去不必要的计算

和其他任务的区别?

- **手工设计：**根据经验设计出更优的固定连接 → DeepLab V3
- **结构搜索：**自动搜索出给定空间中最优的连接 → Auto-DeepLab
- **动态网络：**根据输入自适应地调整网络连接 → Dynamic Routing



Rethinking atrous convolution for semantic image segmentation. *arXiv:1706.05587*, 2017.
Autodeeplab: Hierarchical neural architecture search for semantic image segmentation. In *CVPR*, 2019.
Learning Dynamic Routing for Semantic Segmentation. In *CVPR*, 2020.

1 动态网络简介

2 相关领域研究进展

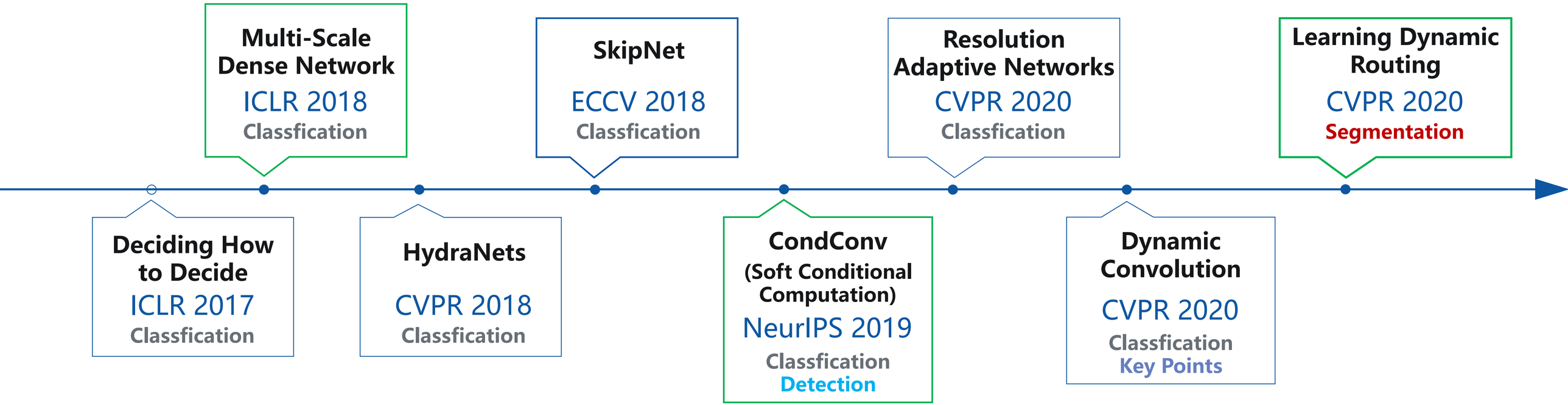
3 动态网络与场景分割

4 未来展望

相关领域研究进展

动态网络研究进展

部分代表性工作：



相关领域研究进展

Multi-Scale Dense Networks

Published as a conference paper at ICLR 2018

MULTI-SCALE DENSE NETWORKS
FOR RESOURCE EFFICIENT IMAGE CLASSIFICATION

Gao Huang Cornell University
Danlu Chen Fudan University
Tianhong Li Tsinghua University
Felix Wu Cornell University
Laurens van der Maaten Facebook AI Research
Kilian Weinberger Cornell University

Motivation:

应使用不同的计算资源来分别相应的图像，对于简单的图像应该使用更少的计算资源从而减少平均计算消耗。

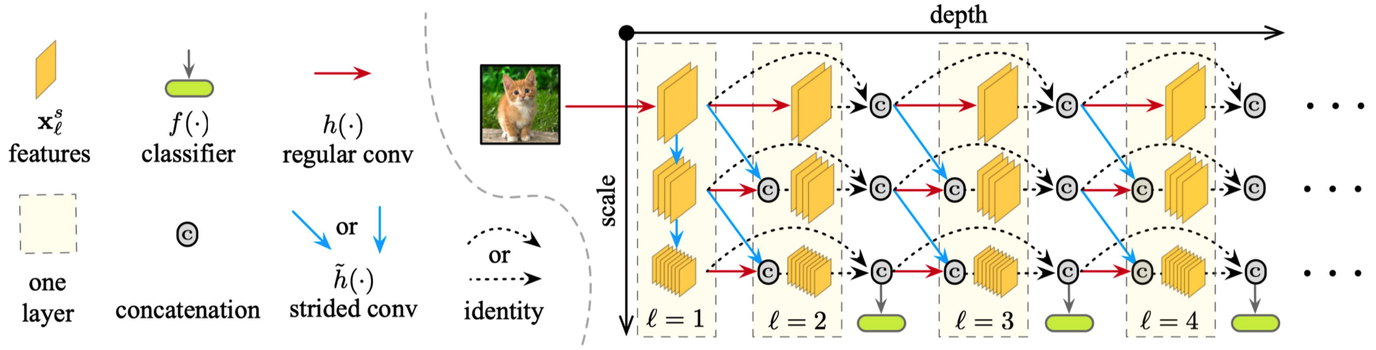
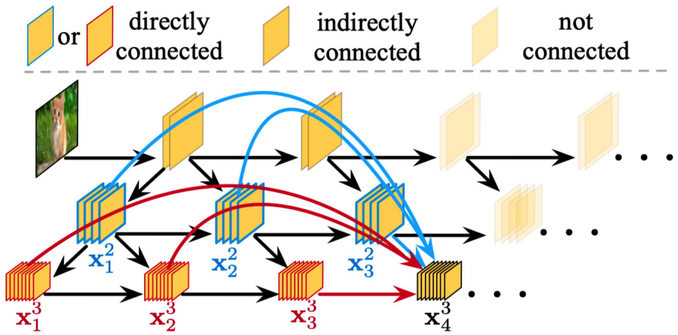


Figure 2: Illustration of the first four layers of an MSDNet with three scales. The horizontal direction corresponds to the layer direction (depth) of the network. The vertical direction corresponds to the scale of the feature maps. Horizontal arrows indicate a regular convolution operation, whereas diagonal and vertical arrows indicate a strided convolution operation. Classifiers only operate on feature maps at the coarsest scale. Connections across more than one layer are not drawn explicitly: they are implicit through recursive concatenations.



x_l^s	$\ell = 1$	$\ell = 2$	$\ell = 3$	$\ell = 4$
$s = 1$	$h_1^1(\mathbf{x}_0^1)$	$h_2^1(\mathbf{x}_1^1)$	$h_3^1(\mathbf{x}_1^1, \mathbf{x}_2^1)$	$h_4^1(\mathbf{x}_1^1, \mathbf{x}_2^1, \mathbf{x}_3^1)$
$s = 2$	$\tilde{h}_1^2(\mathbf{x}_1^1)$	$\begin{bmatrix} \tilde{h}_2^2([\mathbf{x}_1^1]) \\ h_2^2([\mathbf{x}_1^2]) \end{bmatrix}$	$\begin{bmatrix} \tilde{h}_3^2([\mathbf{x}_1^1, \mathbf{x}_2^1]) \\ h_3^2([\mathbf{x}_1^2, \mathbf{x}_2^2]) \end{bmatrix}$	$\begin{bmatrix} \tilde{h}_4^2([\mathbf{x}_1^1, \mathbf{x}_2^1, \mathbf{x}_3^1]) \\ h_4^2([\mathbf{x}_1^2, \mathbf{x}_2^2, \mathbf{x}_3^2]) \end{bmatrix}$
$s = 3$	$\tilde{h}_1^3(\mathbf{x}_1^2)$	$\begin{bmatrix} \tilde{h}_2^3([\mathbf{x}_1^2]) \\ h_2^3([\mathbf{x}_1^3]) \end{bmatrix}$	$\begin{bmatrix} \tilde{h}_3^3([\mathbf{x}_1^2, \mathbf{x}_2^2]) \\ h_3^3([\mathbf{x}_1^3, \mathbf{x}_2^3]) \end{bmatrix}$	$\begin{bmatrix} \tilde{h}_4^3([\mathbf{x}_1^2, \mathbf{x}_2^2, \mathbf{x}_3^2]) \\ h_4^3([\mathbf{x}_1^3, \mathbf{x}_2^3, \mathbf{x}_3^3]) \end{bmatrix}$

Figure 4: The output $\mathbf{x}_\ell^s$ of layer ℓ at the s^{th} scale in a MSDNet. Herein, $[\dots]$ denotes the concatenation operator, $h_\ell^s(\cdot)$ a regular convolution transformation, and $\tilde{h}_\ell^s(\cdot)$ a strided convolutional. Note that the outputs of h_ℓ^s and $\tilde{h}_\ell^s$ have the same feature map size; their outputs are concatenated along the channel dimension.

Multi-Scale Dense Networks for Resource Efficient Image Classification. In *ICLR*, 2018.

相关领域研究进展

Multi-Scale Dense Networks

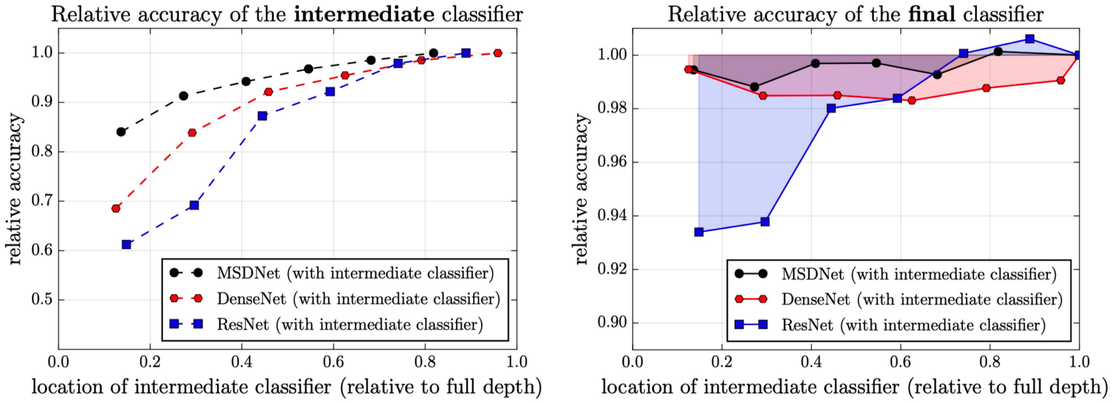


Figure 3: Relative accuracy of the intermediate classifier (*left*) and the final classifier (*right*) when introducing a single intermediate classifier at different layers in a ResNet, DenseNet and MSDNet. All experiments were performed on the CIFAR-100 dataset. Higher is better.

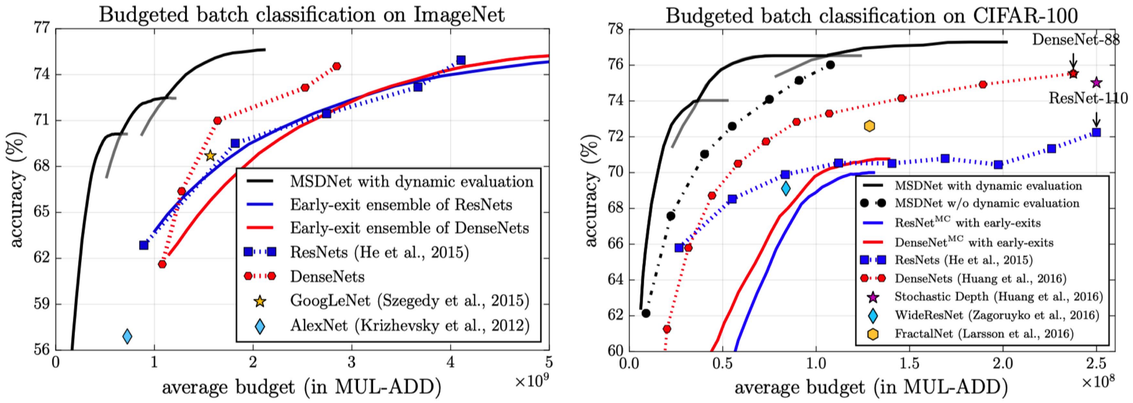


Figure 7: Accuracy (top-1) of *budgeted batch classification* models as a function of average computational budget per image the on ImageNet (left) and CIFAR-100 (right) datasets. Higher is better.

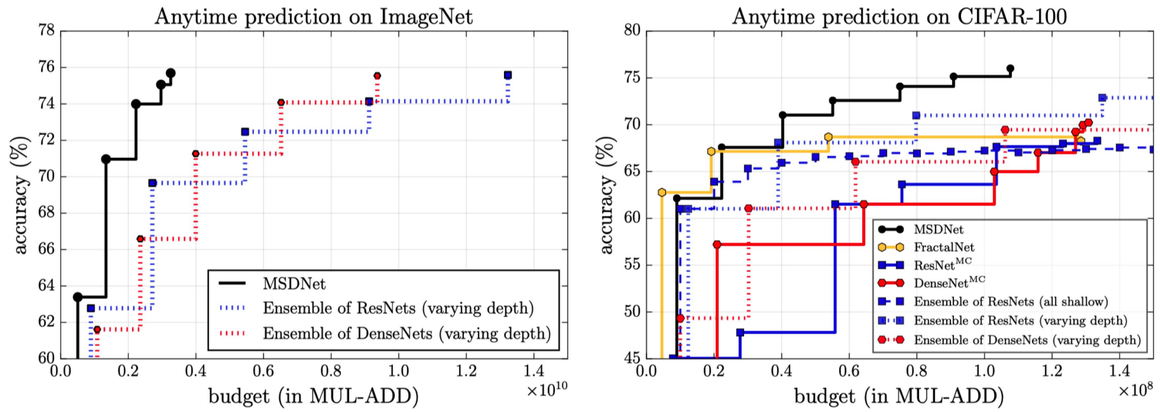


Figure 5: Accuracy (*top-1*) of *anytime prediction* models as a function of computational budget on the ImageNet (left) and CIFAR-100 (right) datasets. Higher is better.

相关领域研究进展

CondConv

CondConv: Conditionally Parameterized Convolutions for Efficient Inference

Brandon Yang*
Google Brain
bcyang@google.com

Gabriel Bender
Google Brain
gbender@google.com

Quoc V. Le
Google Brain
qvl@google.com

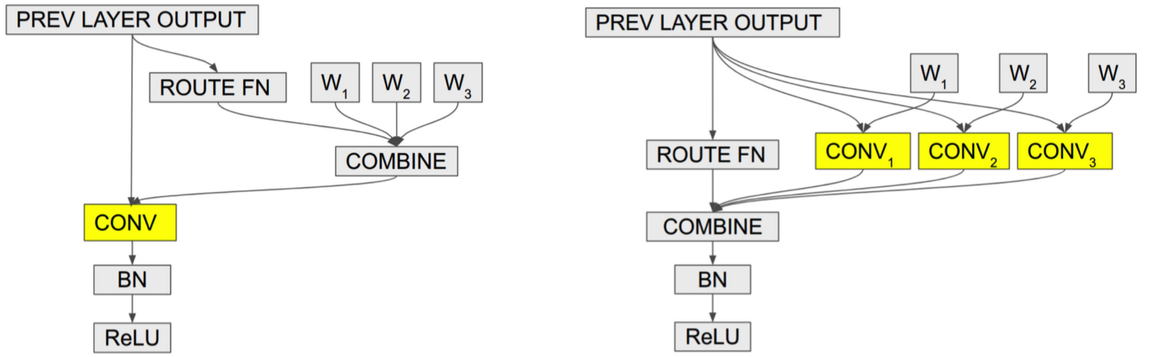
Jiquan Ngiam
Google Brain
jngiam@google.com

Motivation:

针对不同的图片输入学习使用不同的卷积核，从而提升网络的容量和性能。使用参数量的增加来提升网络容量，但计算量保持不变。

$$Output(x) = \sigma((\alpha_1 \cdot W_1 + \ldots + \alpha_n \cdot W_n) * x)$$

$$r(x) = \text{Sigmoid}(\text{GlobalAveragePool}(x)\ R)$$



(a) CondConv: $(\alpha_1 W_1 + \ldots + \alpha_n W_n) * x$ (b) Mixture of Experts: $\alpha_1(W_1 * x) + \ldots + \alpha_n(W_n * x)$

Figure 1: (a) Our CondConv layer architecture with $n = 3$ kernels vs. (b) a mixture of experts approach. By parameterizing the convolutional kernel conditionally on the input, CondConv is mathematically equivalent to the mixture of experts approach, but requires only 1 convolution.

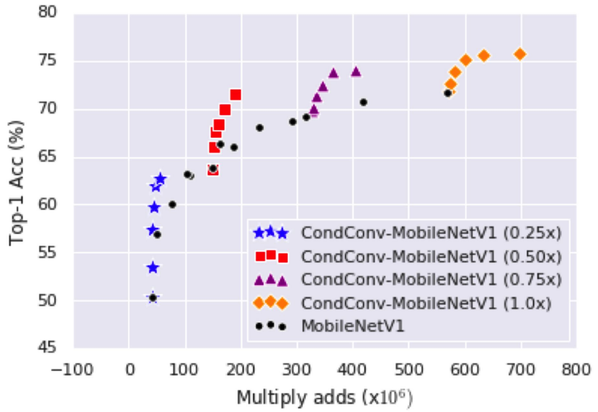


Figure 2: On ImageNet validation, increasing the number of experts per layer of our CondConv-MobileNetV1 models improves performance relative to inference cost compared to the MobileNetV1 frontier [38] across a spectrum of model sizes. Models with more experts per layer achieve monotonically higher accuracy. We train CondConv models with {1, 2, 4, 8, 16, 32} experts at width multipliers {0.25, 0.50, 0.75, 1.0}.

CondConv: Conditionally Parameterized Convolutions for Efficient Inference. In *NeurIPS*, 2019.

相关领域研究进展

CondConv

Table 1: ImageNet validation accuracy and inference cost for our CondConv models on several baseline model architectures. All models use 8 experts per CondConv layer. CondConv improves the accuracy of all baseline architectures with small relative increase in inference cost (<10%).

	Baseline[2]		CondConv	
	MADDs ($\times 10^6$)	Top-1 (%)	MADDs ($\times 10^6$)	Top-1 (%)
MobileNetV1 (1.0x)	567	71.9	600	73.7
MobileNetV2 (1.0x)	301	71.6	329	74.6
MnasNet-A1	312	74.9	325	76.2
ResNet-50	4093	77.7	4213	78.6
EfficientNet-B0	391	77.2	413	78.3

Table 2: COCO object detection minival performance of our CondConv-MobileNetV1 SSD 300 architecture with 8 experts per layer. Mean average precision (mAP) reported with COCO primary challenge metric (AP at IoU=0.50:0.05:0.95). CondConv improves mAP at all model sizes with small relative increase in inference cost (<5%).

	Baseline[3]		CondConv	
	MADDs ($\times 10^6$)	mAP	MADDs ($\times 10^6$)	mAP
MobileNetV1 (0.5x)	352	14.4	363	18.0
MobileNetV1 (0.75x)	730	18.2	755	21.0
MobileNetV1(1.0x)	1230	20.3	1280	22.4

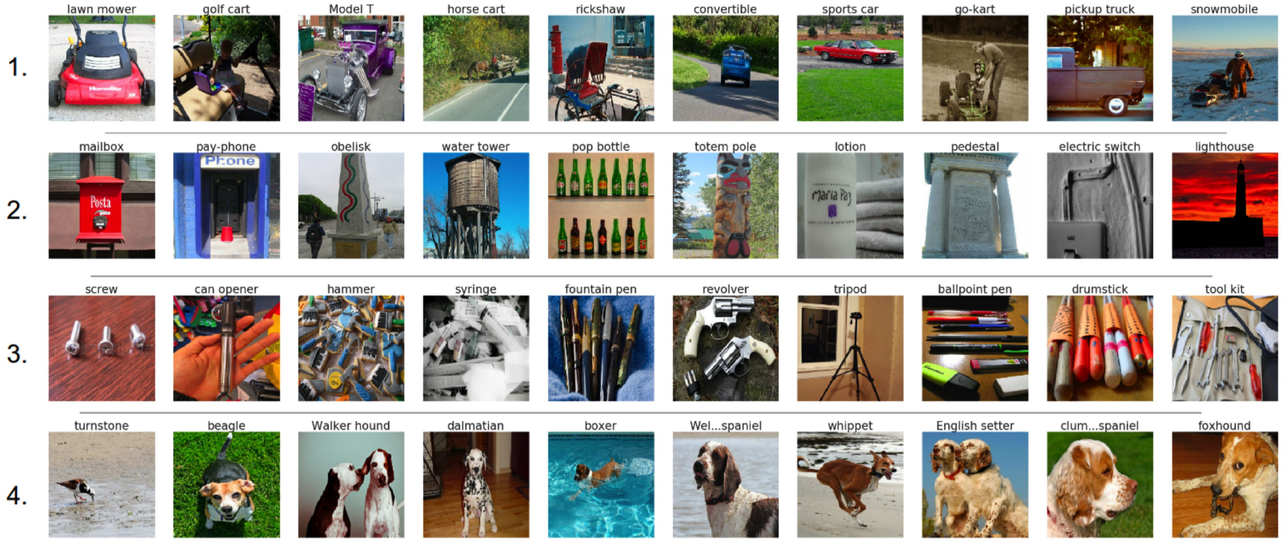
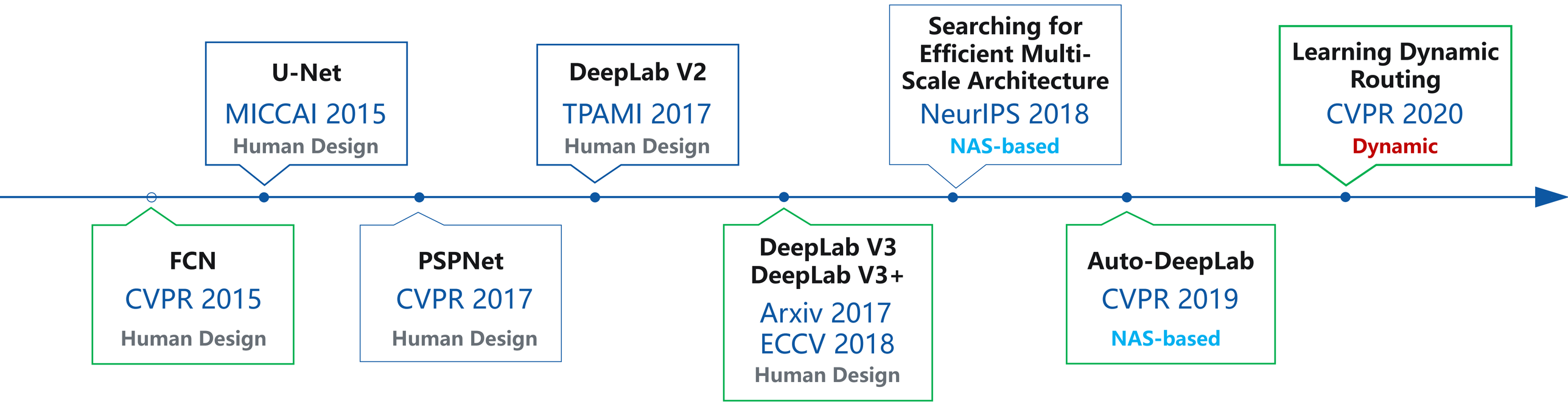


Figure 6: Top 10 classes with highest mean routing weight for 4 different experts in the final CondConv layer in our CondConv-MobileNetV1 (0.5x) model, as measured across the ImageNet validation set. Expert 1 is most activated for wheeled vehicles; expert 2 is most activated for rectangular structures; expert 3 is most activated for cylindrical household objects; expert 4 is most activated for brown and black dog breeds.

相关领域研究进展

场景分割研究进展

部分代表性工作：



相关领域研究进展

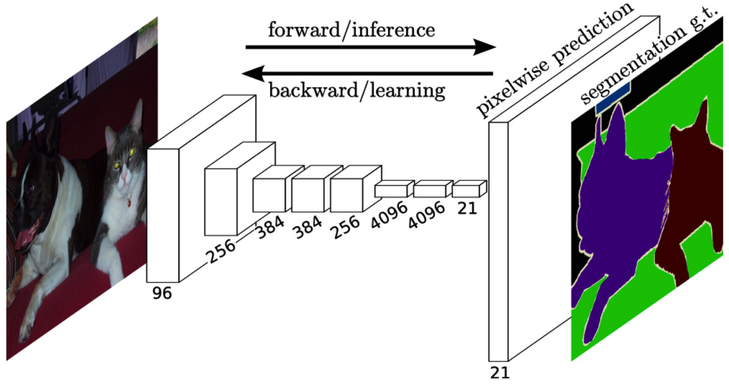
Fully Convolutional Network

Fully Convolutional Networks for Semantic Segmentation

Jonathan Long* Evan Shelhamer* Trevor Darrell
UC Berkeley
{jonlong,shelhamer,trevor}@cs.berkeley.edu

Motivation:

将分类网络结构卷积化，使用全卷积网络来输出和输入相对应的分割预测结果。并提出使用多层级融合的结构来获取多尺度特征。



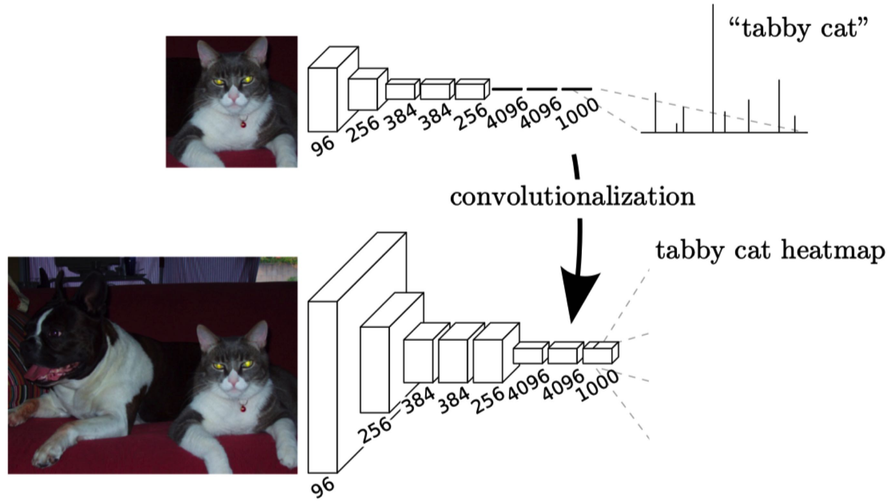


Figure 2. Transforming fully connected layers into convolution layers enables a classification net to output a heatmap. Adding layers and a spatial loss (as in Figure 1) produces an efficient machine for end-to-end dense learning.

Fully Convolutional Network for Semantic Segmentation. In *CVPR*, 2015.

相关领域研究进展

Fully Convolutional Network

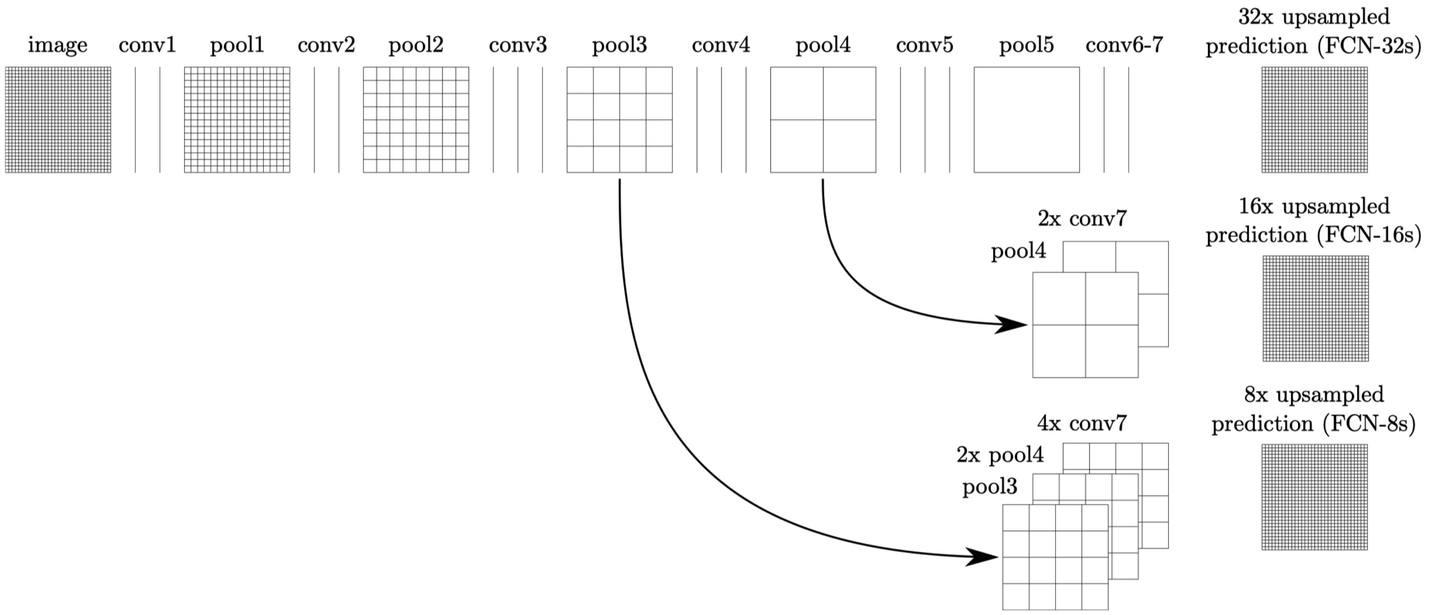


Figure 3. Our DAG nets learn to combine coarse, high layer information with fine, low layer information. Pooling and prediction layers are shown as grids that reveal relative spatial coarseness, while intermediate layers are shown as vertical lines. First row (FCN-32s): Our single-stream net, described in Section 4.1, upsamples stride 32 predictions back to pixels in a single step. Second row (FCN-16s): Combining predictions from both the final layer and the `pool4` layer, at stride 16, lets our net predict finer details, while retaining high-level semantic information. Third row (FCN-8s): Additional predictions from `pool3`, at stride 8, provide further precision.

Table 2. Comparison of skip FCNs on a subset[7] of PASCAL VOC 2011 segval. Learning is end-to-end, except for FCN-32s-fixed, where only the last layer is fine-tuned. Note that FCN-32s is FCN-VGG16, renamed to highlight stride.

	pixel acc.	mean acc.	mean IU	f.w. IU
FCN-32s-fixed	83.0	59.7	45.4	72.0
FCN-32s	89.1	73.3	59.4	81.4
FCN-16s	90.0	75.7	62.4	83.0
FCN-8s	**90.3**	**75.9**	**62.7**	**83.2**

Table 3. Our fully convolutional net gives a 20% relative improvement over the state-of-the-art on the PASCAL VOC 2011 and 2012 test sets and reduces inference time.

	mean IU VOC2011 test	mean IU VOC2012 test	inference time
R-CNN [10]	47.9	-	-
SDS [15]	52.6	51.6	~ 50 s
FCN-8s	**62.7**	**62.2**	~ **175 ms**

相关领域研究进展

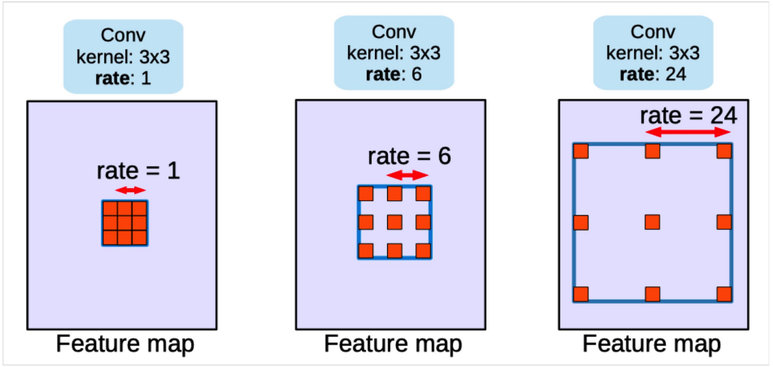
DeepLab V3

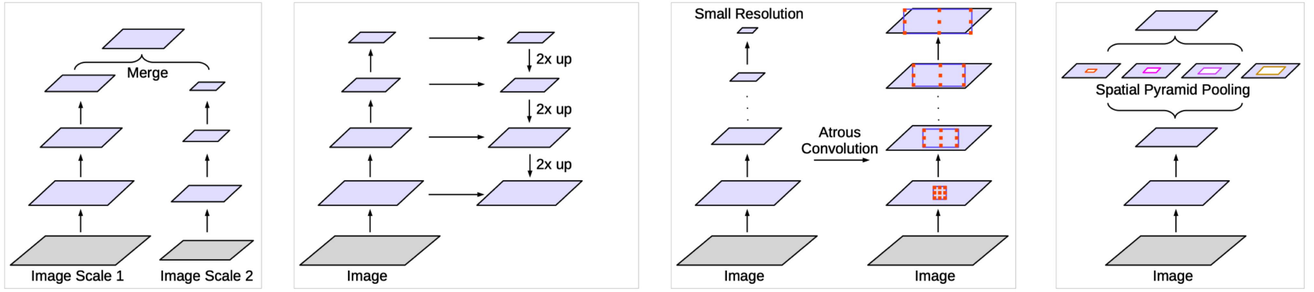
Rethinking Atrous Convolution for Semantic Image Segmentation

Liang-Chieh Chen George Papandreou Florian Schroff Hartwig Adam
Google Inc.
{lcchen, gpapan, fschroff, hadam}@google.com

Motivation:

设计基于空洞卷积的模块并结合DeepLab V2中提出的ASPP来更好地编码全局特征，在不使用DenseCRF的情况下获得更好的结果。





(a) Image Pyramid (b) Encoder-Decoder (c) Deeper w. Atrous Convolution (d) Spatial Pyramid Pooling

Figure 2. Alternative architectures to capture multi-scale context.

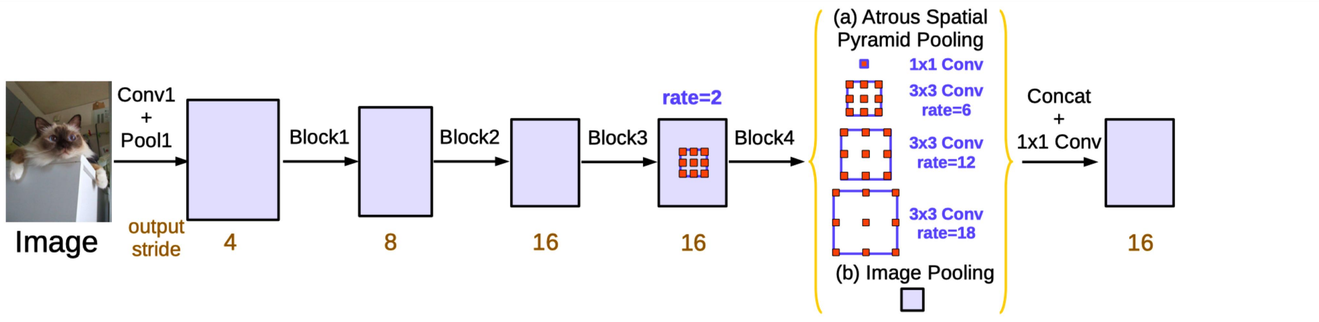


Figure 5. Parallel modules with atrous convolution (ASPP), augmented with image-level features.

相关领域研究进展

Auto-DeepLab

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

Chenxi Liu[1]*, Liang-Chieh Chen[2], Florian Schroff[2], Hartwig Adam[2], Wei Hua[2],
Alan Yuille[1], Li Fei-Fei[3]
[1]Johns Hopkins University [2]Google [3]Stanford University

Motivation:

针对场景分割，将NAS应用到整个网络结构层面的搜索，使用Viterbi算法来找到空间中激活权重最大的网络连接，并取得了不错的效果。

Model	Auto Search				Task
	Cell	Network	Dataset	Days	
ResNet [25]	✗	✗	-	-	Cls
DenseNet [31]	✗	✗	-	-	Cls
DeepLabv3+ [11]	✗	✗	-	-	Seg
NASNet [93]	✓	✗	CIFAR-10	2000	Cls
AmoebaNet [62]	✓	✗	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

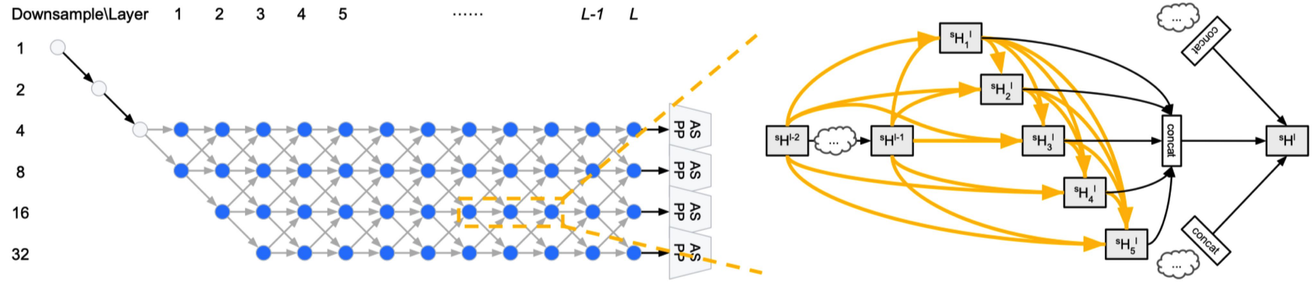


Figure 1: *Left:* Our network level search space with $L = 12$. Gray nodes represent the fixed "stem" layers, and a path along the blue nodes represents a candidate network level architecture. *Right:* During the search, each cell is a densely connected structure as described in Sec. 4.1.1. Every yellow 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.

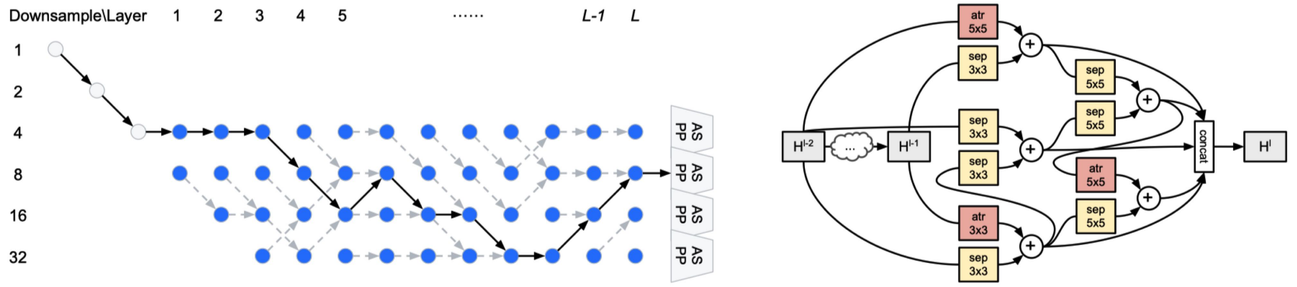


Figure 3: The Auto-DeepLab architecture found by our Hierarchical Neural Architecture Search on Cityscapes. Gray dashed arrows show the connection with maximum β at each node. **atr:** atrous convolution. **sep:** depthwise-separable convolution.

1 动态网络简介

2 相关领域研究进展

3 动态网络与场景分割

4 未来展望

动态网络与场景分割

Learning Dynamic Routing for Semantic Segmentation

Learning Dynamic Routing for Semantic Segmentation

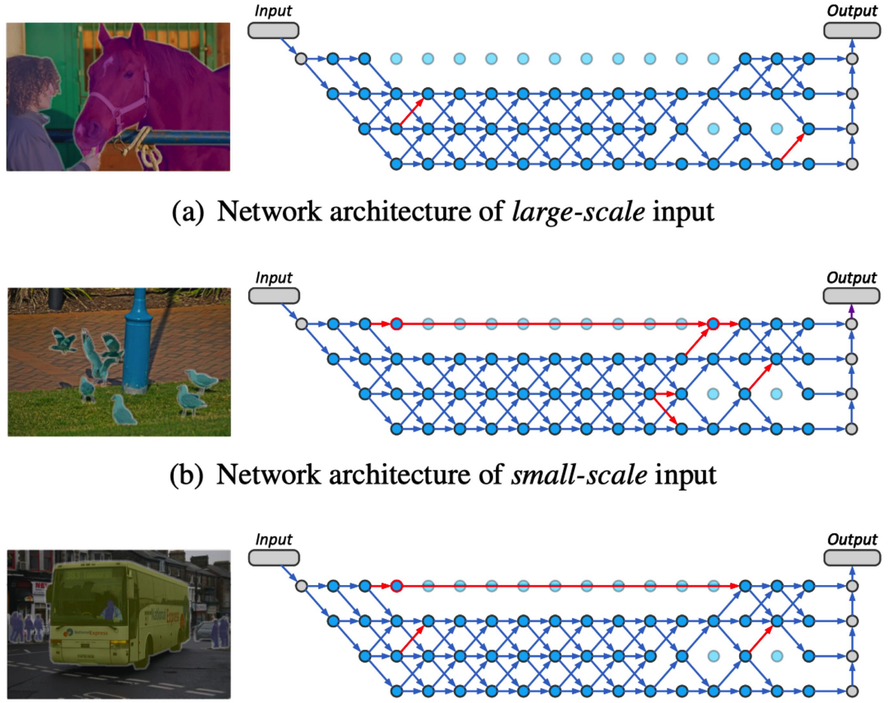
Yanwei Li[1,2], Lin Song[3], Yukang Chen[1,2], Zeming Li[4], Xiangyu Zhang[4],
Xingang Wang[1], Jian Sun[4]
[1]Institute of Automation, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences
[3]Xi'an Jiaotong University [4]Megvii Technology

Motivation:

实际场景中物体尺寸分布往往存在着巨大的差异。针对不同输入图片的尺寸分布，选择不同的前向推断路径进行处理，并针对给定的计算资源限制进行调整。

Paper
https://arxiv.org/abs/2003.10401

Code
https://github.com/yanwei-li/DynamicRouting



(a) Network architecture of *large-scale* input

(b) Network architecture of *small-scale* input

(c) Network architecture of *mixed-scale* input

Figure 1. Given inputs with different scale distributions, the proposed dynamic routing will choose corresponding forward paths. For example, the architecture of *large-scale* instances 1(a) could ignore low-level features. The *small-scale* objects 1(b) may depend on low-level details as well as higher resolution. And the *mixed-scale* things 1(c) would enjoy both connection patterns. Red lines in diagrams denote the difference among them.

Learning Dynamic Routing for Semantic Segmentation. In *CVPR*, 2020.

动态网络与场景分割

整体网络结构

针对场景分割所设计的动态网络结构，在空间中的每一个节点中均可动态地选择上采样、保持分辨率、或进行下采样。所设计的动态网络能够支持每次前向推断时的**跨层连接**和**多路径推断**。

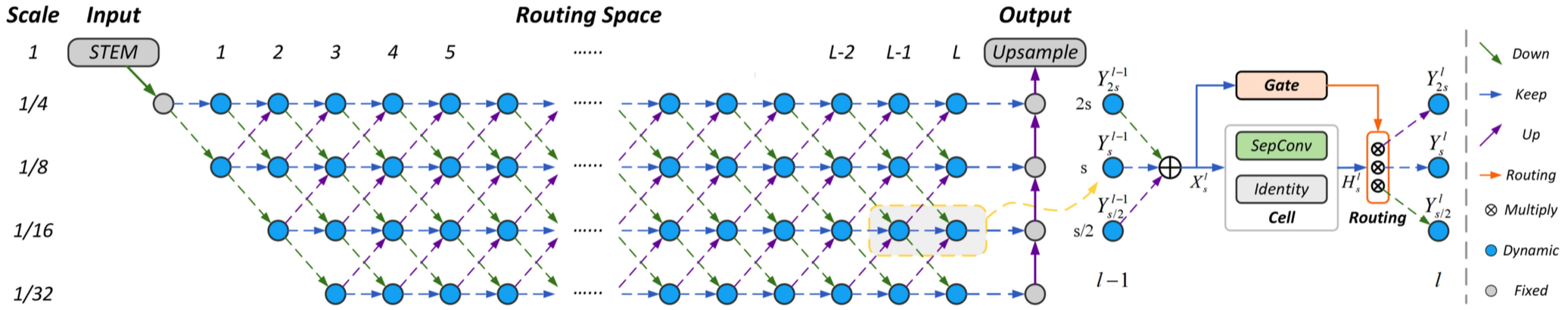
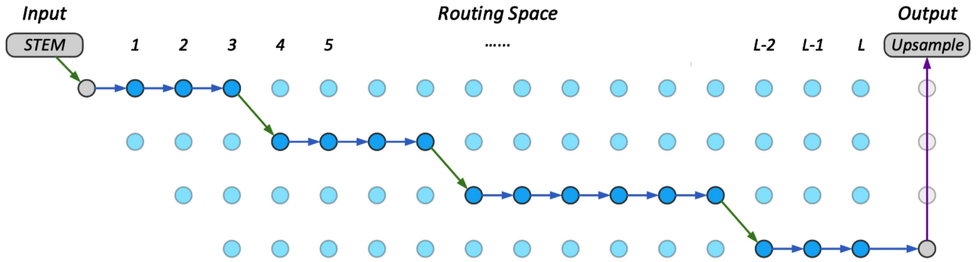


Figure 2. The proposed *dynamic routing* framework for semantic segmentation. *Left:* The routing space with layer L and max down-sampling rate 32. The beginning *STEM* and the final *Upsample* block are fixed for stability. Dashed lines denote alternative paths for dynamic routing. *Right:* Dynamic routing process at the cell level. Given the summed input from the former layer, we first generate activating weights using the *Soft Conditional Gate*. Paths with corresponding weights above zero are marked as activated, which would be selected for feature transformation. More details about the network are elaborated in Sec. 3.4. Best viewed in color.

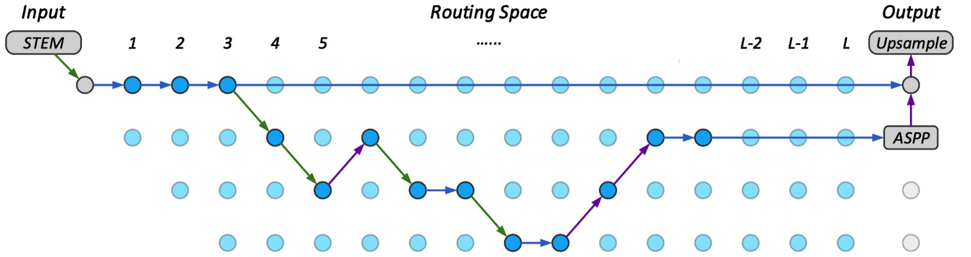
动态网络与场景分割

动态网络空间

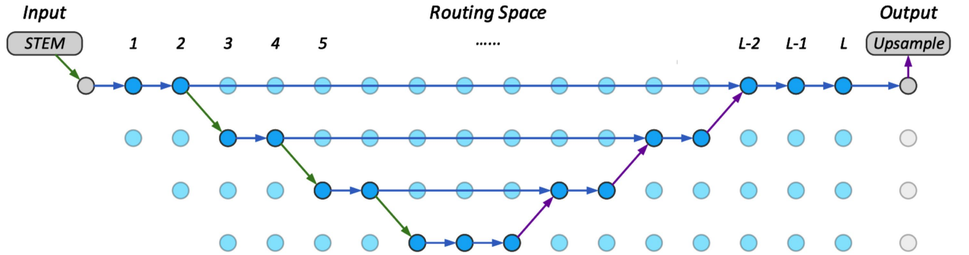
在所设计的动态网络空间中对经典网络进行建模，可以将经典网络结构动态网络当做单次前向推断的子集。



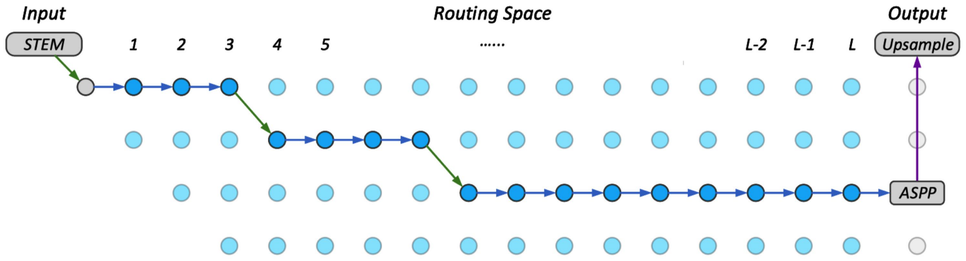
(a) Network architecture modeled from FCN-32s [24]



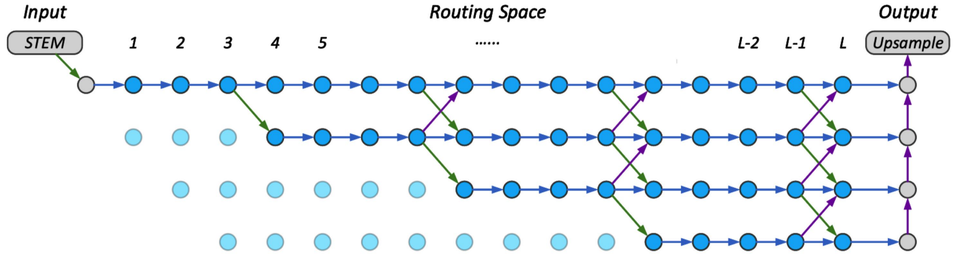
(e) Network architecture modeled from Auto-DeepLab [22]



(b) Network architecture modeled from U-Net [28]



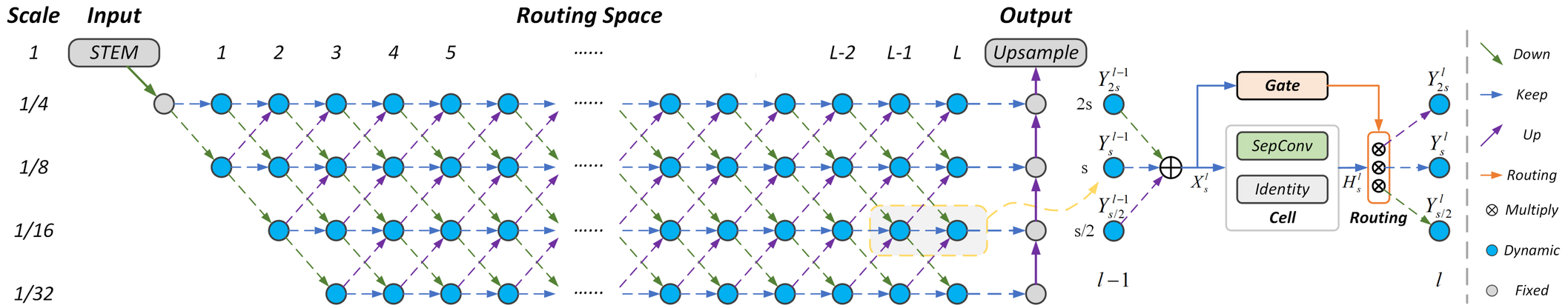
(c) Network architecture modeled from DeepLabV3 [5]



(d) Network architecture modeled from HRNetV2 [32]

动态网络与场景分割

动态路径选择过程



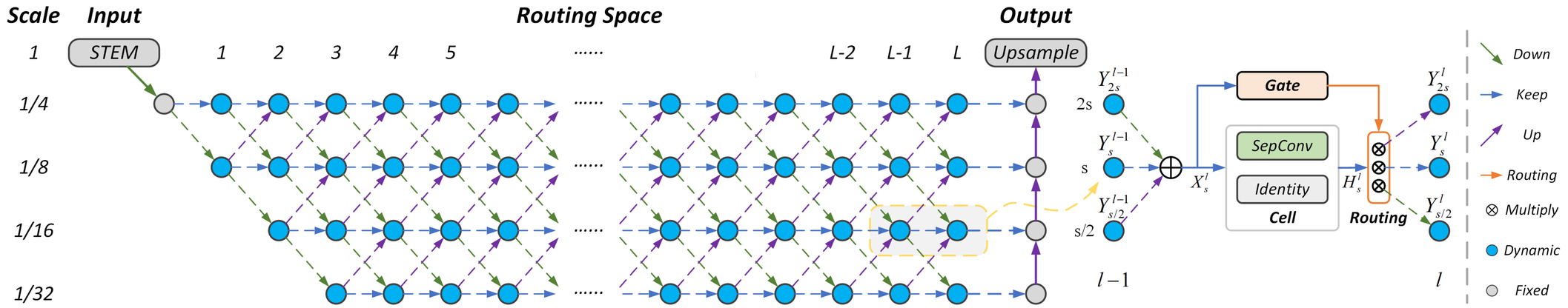
Cell操作

进行多尺度特征聚合

$$\mathbf{H}_s^l = \sum_{O^i \in \mathcal{O}} O^i(\mathbf{X}_s^l)$$

动态网络与场景分割

动态路径选择过程



Cell操作

进行多尺度特征聚合

$$\mathbf{H}_s^l = \sum_{O^i \in \mathcal{O}} O^i(\mathbf{X}_s^l)$$

软条件门控

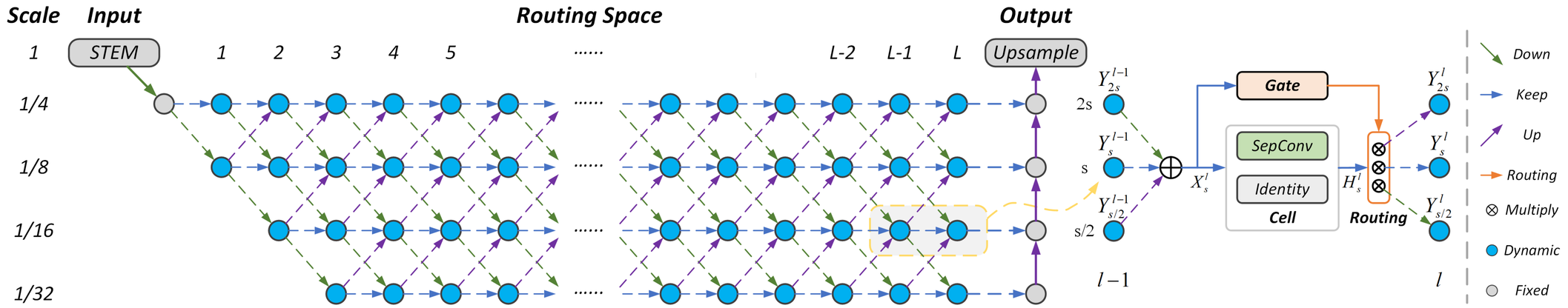
根据输入特征选择该节点的前向推断路径

$$\mathbf{G}_s^l = \mathcal{F}(\omega_{s,2}^l, \mathcal{G}(\sigma(\mathcal{N}(\mathcal{F}(w_{s,1}^l, \mathbf{X}_s^l))))) + \beta_s^l$$

$$\delta(\cdot) = \max(0, \mathrm{Tanh}(\cdot)) \qquad \mathbf{Y}_j^l = \alpha_{s\to j}^l \mathcal{T}_{s\to j}(\mathbf{H}_s^l)$$

动态网络与场景分割

动态路径选择过程



Cell操作

进行多尺度特征聚合

$$\mathbf{H}_s^l = \sum_{O^i \in \mathcal{O}} O^i(\mathbf{X}_s^l)$$

软条件门控

根据输入特征选择该节点的前向推断路径

$$\mathbf{G}_s^l = \mathcal{F}(\omega_{s,2}^l, \mathcal{G}(\sigma(\mathcal{N}(\mathcal{F}(w_{s,1}^l, \mathbf{X}_s^l))))) + \beta_s^l$$

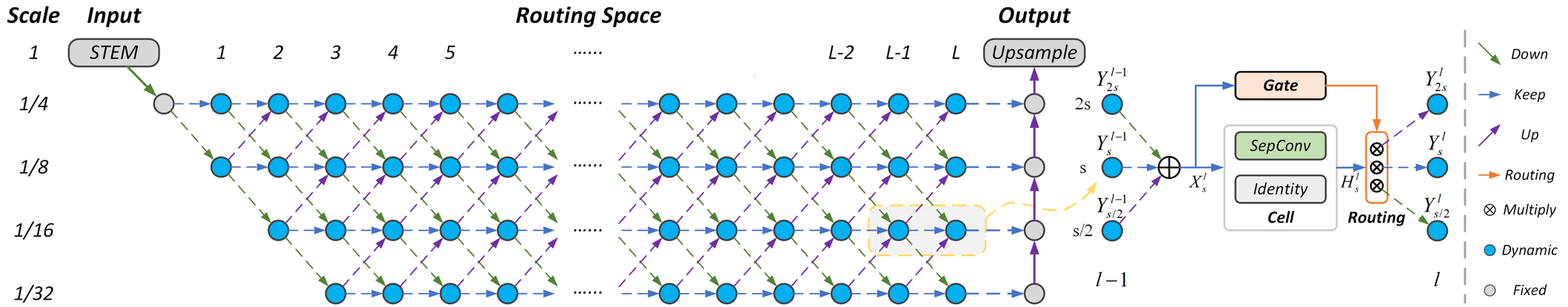
$$\delta(\cdot) = \max(0, \mathrm{Tanh}(\cdot)) \qquad \mathbf{Y}_j^l = \alpha_{s\to j}^l \mathcal{T}_{s\to j}(\mathbf{H}_s^l)$$

$$\mathbf{H}_s^l = \begin{cases} \mathbf{X}_s^l & \sum_j \alpha_{s\to j}^l = 0 \\ \sum_{O^i \in \mathcal{O}} O^i(\mathbf{X}_s^l) & \sum_j \alpha_{s\to j}^l > 0 \end{cases}$$

$$\mathbf{Y}_j^l = \begin{cases} 0 & \sum_j \alpha_{s\to j}^l = 0,\ j \neq s \\ \mathbf{H}_s^l & \sum_j \alpha_{s\to j}^l = 0,\ j = s \\ \alpha_{s\to j}^l \mathcal{T}_{s\to j}(\mathbf{H}_s^l) & \sum_j \alpha_{s\to j}^l > 0 \end{cases}$$

动态网络与场景分割

动态路径选择过程



Cell操作

进行多尺度特征聚合

$$\mathbf{H}_s^l = \sum_{O^i \in \mathcal{O}} O^i(\mathbf{X}_s^l)$$

软条件门控

根据输入特征选择该节点的前向推断路径

$$\mathbf{G}_s^l = \mathcal{F}(\omega_{s,2}^l, \mathcal{G}(\sigma(\mathcal{N}(\mathcal{F}(w_{s,1}^l, \mathbf{X}_s^l))))) + \beta_s^l$$

$$\delta(\cdot) = \max(0, \mathrm{Tanh}(\cdot)) \qquad \mathbf{Y}_j^l = \alpha_{s\to j}^l \mathcal{T}_{s\to j}(\mathbf{H}_s^l)$$

$$\mathbf{H}_s^l = \begin{cases} \mathbf{X}_s^l & \sum_j \alpha_{s\to j}^l = 0 \\ \sum_{O^i \in \mathcal{O}} O^i(\mathbf{X}_s^l) & \sum_j \alpha_{s\to j}^l > 0 \end{cases}$$

$$\mathbf{Y}_j^l = \begin{cases} 0 & \sum_j \alpha_{s\to j}^l = 0,\ j \neq s \\ \mathbf{H}_s^l & \sum_j \alpha_{s\to j}^l = 0,\ j = s \\ \alpha_{s\to j}^l \mathcal{T}_{s\to j}(\mathbf{H}_s^l) & \sum_j \alpha_{s\to j}^l > 0 \end{cases}$$

给定计算约束

在给定计算资源约束的情况下进行端到端的训练

$$\begin{aligned} \mathcal{C}(\mathrm{Node}_s^l) &= \mathcal{C}(\mathrm{Cell}_s^l) + \mathcal{C}(\mathrm{Gate}_s^l) + \mathcal{C}(\mathrm{Trans}_s^l) \\ &= \max(\alpha_s^l) \textstyle\sum_{O^i \in \mathcal{O}} \mathcal{C}(O^i) + \mathcal{C}(\mathrm{Gate}_s^l) \\ &+ \textstyle\sum_j \alpha_{s\to j}^l \mathcal{C}(\mathcal{T}_{s\to j}) \end{aligned}$$

$$\mathcal{C}(\mathrm{Space}) = \sum_{l \le L} \sum_{s \le 1/4} \mathcal{C}(\mathrm{Node}_s^l)$$

$$\mathcal{L}_\mathrm{C} = (\mathcal{C}(\mathrm{Space})/\mathrm{C} - \mu)^2 \qquad \mathcal{L} = \lambda_1 \mathcal{L}_\mathrm{N} + \lambda_2 \mathcal{L}_\mathrm{C}$$

动态网络与场景分割

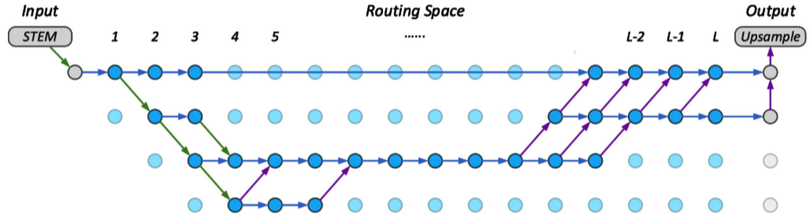
动态路径选择实验

相似资源消耗的实验对比

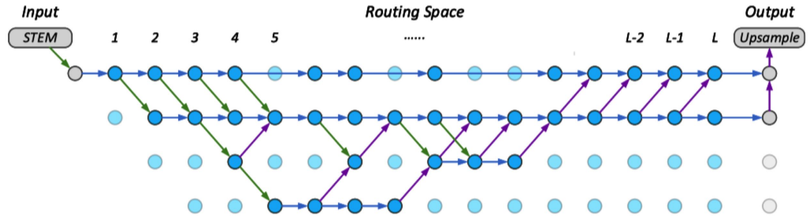
- 在动态网络空间中对经典的网络结构进行建模，并对动态网络结构进行约束，在**相似的计算量**下进行比较。
- 从动态网络中抽取出95%的前向推断均会选择的路径作为静态的**基础网络**加入比较实验。

Table 1. Comparisons with classic architectures on the Cityscapes *val* set. 'Dynamic' denotes the proposed *dynamic routing*. 'A', 'B', and 'C' represent different computational budgets in Sec. 4.4.3. 'Common' indicates the common connection pattern of corresponding dynamic network. FLOPs_{Avg}, FLOPs_{Max}, and FLOPs_{Min} represent the *Average*, *Maximum*, and *Minimum* FLOPs of the network, respectively. All of the architectures are sampled from the designed routing space and evaluated by ourself under the same setting.

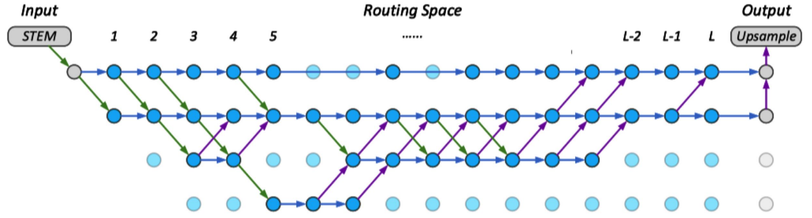
Method	Dynamic	Modeled from	mIoU(%)	FLOPs_{Avg}(G)	FLOPs_{Max}(G)	FLOPs_{Min}(G)	Params(M)
Handcrafted	✗	FCN-32s [24]	66.9	35.1	35.1	35.1	2.9
	✗	DeepLabV3 [5]	67.0	42.5	42.5	42.5	3.7
	✗	U-Net [28]	71.6	53.9	53.9	53.9	6.1
	✗	HRNetV2 [32]	72.5	62.5	62.5	62.5	5.4
Searched	✗	Auto-DeepLab [22]	67.2	33.1	33.1	33.1	2.5
Common-A	✗	Dynamic-A	71.6	41.6	41.6	41.6	4.1
Common-B	✗	Dynamic-B	73.0	53.7	53.7	53.7	4.3
Common-C	✗	Dynamic-C	73.2	57.1	57.1	57.1	4.5
Dynamic-A	✓	Routing-Space	72.8	44.9	48.2	43.5	17.8
Dynamic-B	✓	Routing-Space	73.8	58.7	63.5	56.8	17.8
Dynamic-C	✓	Routing-Space	74.6	66.6	71.6	64.3	17.8



(a) Network architecture of Common-A



(b) Network architecture of Common-B



(c) Network architecture of Common-C

Figure 4. Network architectures of Common-A, B, C, which are extracted from Dynamic models with different budget constraints in Tab. 1, are visualized in 4(a), 4(b), and 4(c), respectively.

动态网络与场景分割

对比实验

Cell内部不同操作的实验

Table 2. Comparisons among different cell components on the Cityscapes *val* set. '×2' and '×3' mean stacking 2 and 3 SepConv3×3, respectively. Due to the data-dependent property of the dynamic routing, we report the *average* FLOPs here.

Cell Operation	mIoU(%)	FLOPs(G)	Params(M)
BottleNeck [16]	73.7	1134.8	203.9
MBConv [29]	75.0	323.8	48.2
SepConv3×3	71.2	81.4	12.6
SepConv3×3 ×2	76.1	119.5	17.8
SepConv3×3 ×3	75.2	153.8	22.9

门控网络中不同激活函数的实验

Table 3. Comparisons among different activation functions on the Cityscapes *val* set. Due to the data-dependent property of the dynamic routing, we report the *average* FLOPs here.

Activation	mIoU(%)	FLOPs(G)	Params(M)
Fix	74.5	103.1	15.3
Softmax	74.1	120.0	17.8
Sigmoid	75.9	120.0	17.8
max(0, Tanh)	76.1	119.5	17.8

不同资源约束系数的实验

Table 4. Comparisons among different resource budgets on the Cityscapes *val* set. λ_2 and μ denote the coefficients for budget constraint in Sec. 3.3. Due to the data-dependent property of the dynamic routing, we report the *average* FLOPs here.

Method	λ_2/μ	mIoU(%)	FLOPs(G)	Params(M)
Network-Fix	-	74.5	103.1	15.3
Dynamic-A	0.8/0.1	72.8	44.9	17.8
Dynamic-B	0.5/0.1	73.8	58.7	17.8
Dynamic-C	0.5/0.2	74.6	66.6	17.8
Dynamic-Raw	0.0/0.0	76.1	119.5	17.8

不同训练步长的实验

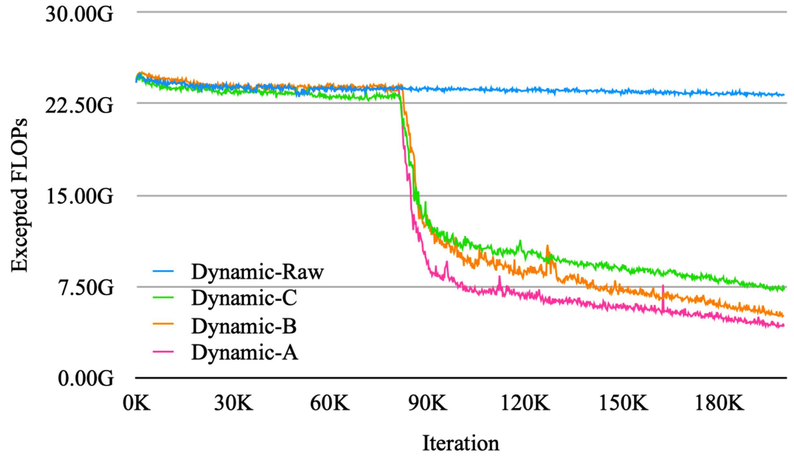
Table 5. Experiments with different settings on Cityscapes *val* set with a single scale and no flipping. 'ImageNet' denotes ImageNet pre-training. 'SDP' indicates Scheduled Drop Path.

Method	Iter(K)	ImageNet	SDP	mIoU(%)
Network-Fix	186	✗	✗	74.5
Dynamic	186	✗	✗	76.1
Network-Fix	372	✗	✗	76.3
Dynamic	372	✗	✗	77.4
Network-Fix	558	✗	✓	76.7
Dynamic	558	✗	✓	78.3
Network-Fix	186	✓	✗	75.8
Dynamic	186	✓	✗	**78.6**

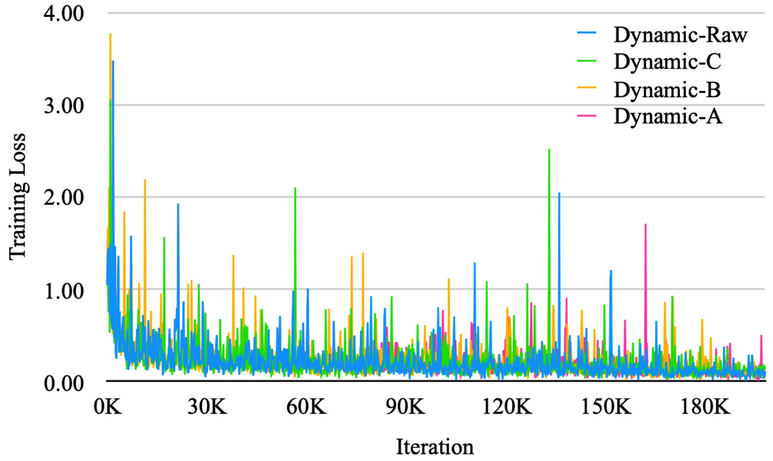
动态网络与场景分割

可视化分析

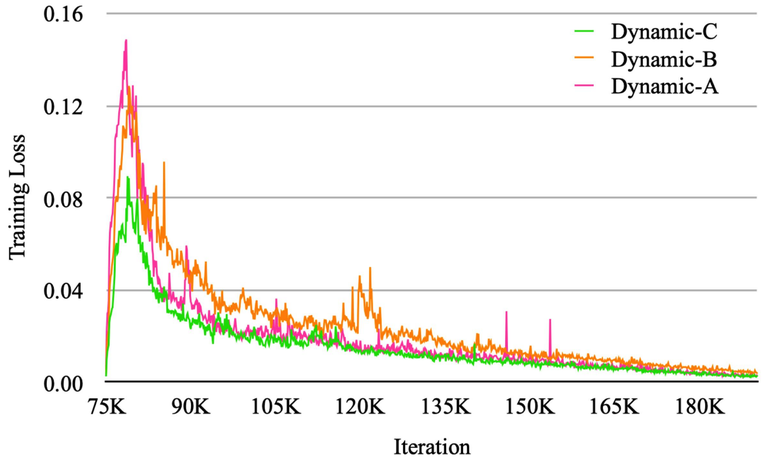
不同资源约束下的FLOPs期望



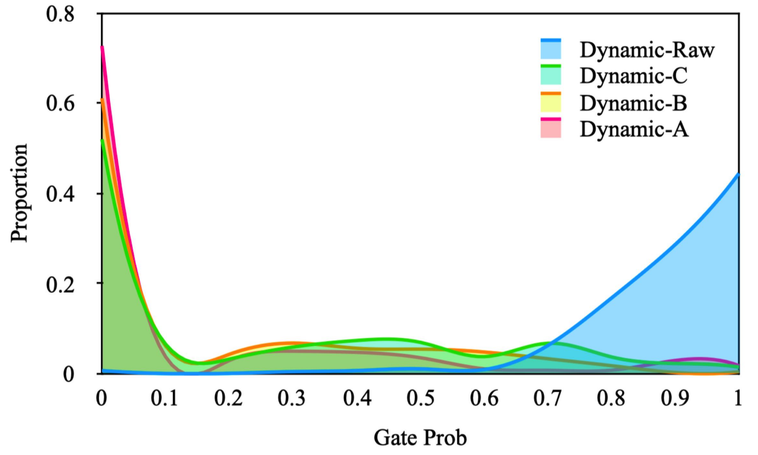
不同资源约束下的网络整体损失



不同资源约束下的资源损失



不同资源约束下的门控网络激活值分布

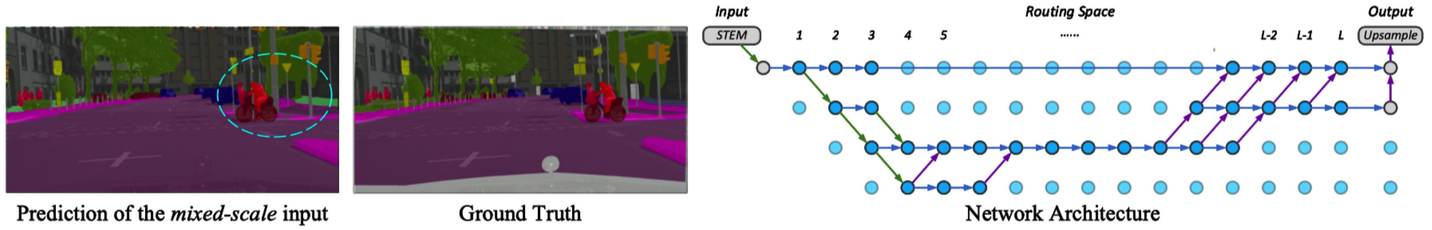


动态网络与场景分割

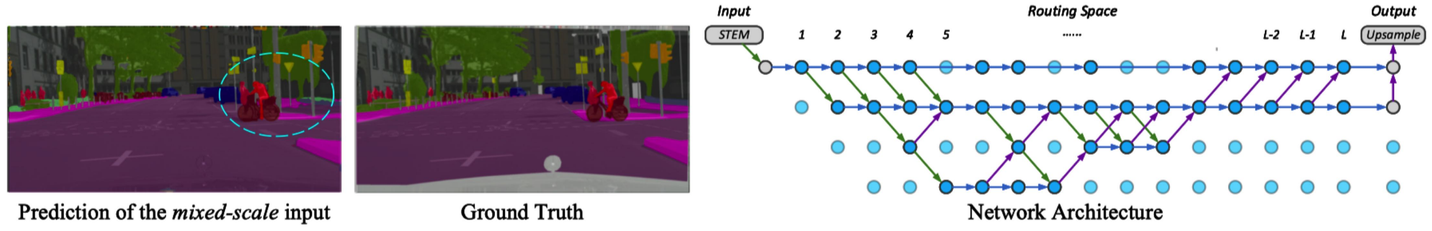
可视化分析

给定相同图片在不同计算约束下的前向推断网络结构

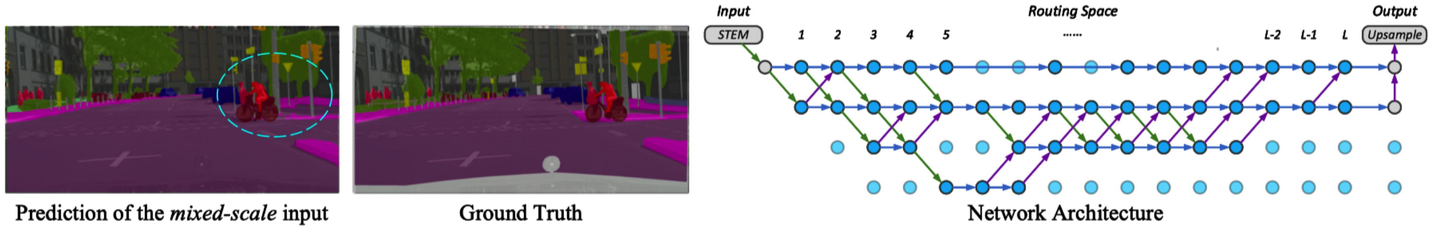
可以看出在资源预算较小的时候会使用更少的计算节点，并产生次优但仍可接受的预测结果。



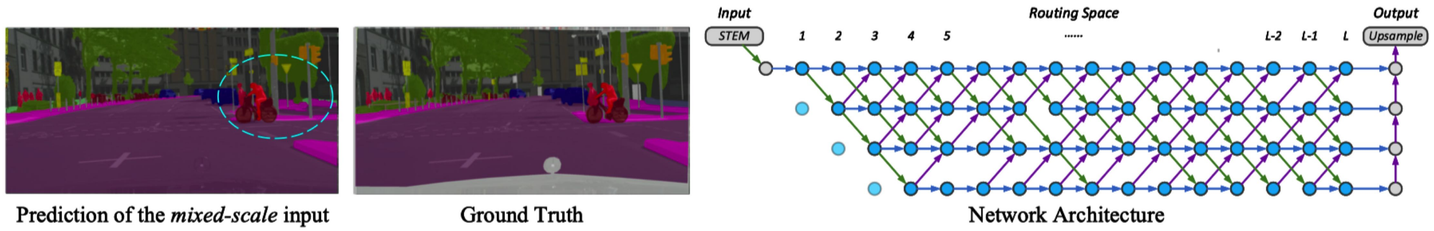
(a) Network architecture of Dynamic-A



(b) Network architecture of Dynamic-B



(c) Network architecture of Dynamic-C



(d) Network architecture of Dynamic-Raw

动态网络与场景分割

结果分析

在Cityscapes和VOC 2012数据集上的结果

动态网络在使用小得多的计算量的情况下取得了和SOTA相似的结果。并且使用上下文建立模块（如PSP模块）会有进一步提升。

Table 6. Comparisons with previous works on the Cityscapes. mIoU_{test} and mIoU_{val} denote performance on *test* set and *val* set respectively. Multi-scale and flipping strategy are used in *test* set but dropped in *val* set. We report FLOPs with input size 1024×2048.

Method	Backbone	$\text{mIoU}_{test}(\%)$	$\text{mIoU}_{val}(\%)$	FLOPs(G)
BiSenet [40]	ResNet-18	77.7	74.8	98.3†
DeepLabV3 [5]	ResNet-101-ASPP	-	78.5	1778.7
Semantic FPN [19]	ResNet-101-FPN	-	77.7	500.0
DeepLabV3+ [6]	Xception-71-ASPP	-	79.6	1551.1
PSPNet [43]	ResNet-101-PSP	78.4	79.7	2017.6
Auto-DeepLab* [22]	Searched-F20-ASPP	79.9	79.7	333.3
Auto-DeepLab* [22]	Searched-F48-ASPP	80.4	80.3	695.0
Dynamic*	Layer16	79.1	78.3	111.7
Dynamic	Layer16	79.7	78.6	119.4
Dynamic	Layer33	80.0	79.2	242.3
Dynamic	Layer33-PSP	**80.7**	**79.7**	**270.0**

† estimated from corresponding settings
* training from scratch

Table 7. Comparisons with previous works on the PASCAL VOC 2012. mIoU_{test} and mIoU_{val} denote performance on *test* set and *val* set respectively. Multi-scale and flipping strategy are used in *test* set but dropped in *val* set. We report FLOPs with input size 512×512.

Method	Backbone	$\text{mIoU}_{test}(\%)$	$\text{mIoU}_{val}(\%)$	FLOPs(G)
DeepLabV3 [5]	MobileNet-ASPP	-	75.3	14.3
DeepLabV3 [5]	MobileNetV2-ASPP	-	75.7	5.8
Auto-DeepLab [22]	Searched-F20-ASPP	82.5	78.3	41.7†
Dynamic	Layer16	82.8	78.6	14.9
Dynamic	Layer33	**84.0**	**79.0**	**30.8**

† estimated from corresponding settings

1 动态网络简介

2 相关领域研究进展

3 动态网络与场景分割

4 未来展望

未来展望

更多任务

将动态网络应用至更多任务

动态网络在场景分割方面取得了不错的结果，这给我们在其他Dense Prediction的任务留下了充足的想象空间。实际上这种多尺度融合的预测结构对于其他任务同样有着重要在作用。

未来展望

更多任务

将动态网络应用至更多任务

动态网络在场景分割方面取得了不错的结果，这给我们在其他Dense Prediction的任务留下了充足的想象空间。实际上这种多尺度融合的预测结构对于其他任务同样有着重要在作用。

更加动态化

更充分利用动态模型容量

当前的动态路径选择方式仍存在改进的空间。由于资源损失函数的约束，同一动态网络中最大的模型FLOPs和最小模型FLOPs差距在10%以内。可以通过新的方式更加充分地利用动态模型容量。

未来展望

更多任务

将动态网络应用至更多任务

动态网络在场景分割方面取得了不错的结果，这给我们在其他Dense Prediction的任务留下了充足的想象空间。实际上这种多尺度融合的预测结构对于其他任务同样有着重要在作用。

更加动态化

更充分利用动态模型容量

当前的动态路径选择方式仍存在改进的空间。由于资源损失函数的约束，同一动态网络中最大的模型FLOPs和最小模型FLOPs差距在10%以内。可以通过新的方式更加充分地利用动态模型容量。

更快的速度

提升动态网络的推断速度

由于当前所设计的动态网络需要逐层进行特征的聚合和运算，实际的推断速度仍存在提升的空间。可以设计新的动态网络结构或通过工程优化来加速前向推断，使其具有更广阔的应用场景。

PPT
http://yanwei-li.com/

Paper
https://arxiv.org/abs/2003.10401

Code
https://github.com/yanwei-li/DynamicRouting

AD Time!

旷视研究院
Base-Detection组招聘实习生和正式员工
欢迎砸简历给Team Leader
lizeming@megvii.com

Thanks!