论文名

代码地址

论文地址

KeyPoint

- 一个更灵活的网络来**适应每个图像的尺度变化。**
- 动态路由能够生成与数据相关的路径，这意味着特定的网络结构随输入而变化。
- 通过简单的尺度转换模块，本文提出的动态路由实现了与SOTA方法相当的结果，但消耗的计算资源要少得多。
- 与Auto-DeepLab中每个节点只选择一条特定路径不同，在推理阶段，本文的动态路由进一步放宽了路由空间，以支持多路径路由
- 如果基于路由空间构建动态网络，参数量比一般的网络增加了3-6倍不等，但是其通过该方法找到的网络，如果静态化，效果比一般的效果好2%左右。
- **论文中发现网络的最大浮点运算量和最小浮点运算量相差不大，即使采用相差很大的运算开支约束系数来控制开支运算，不同网络间的浮点运算量仍然达不到很好的差距**
- 与人工设计的语义分割网络相比**计算量提高很大**，但是**精度提升较大。**

terms

- **动态路由（dynamic routing）**
- **软条件门（Soft Conditional Gate）**

- **“STEM”块**
- **Cell Operation**
- **Soft Conditional Gate**
- **sepConv** 深度可分离卷积
- NAS：神经网络搜索技术
- Identity mapping：恒等映射

摘要

最近，大量的手工搜索网络被应用于语义分割。然而，以前的工作试图在预定义的静态体系结构中处理不同规模的输入，如 fcn、 u-net 和 deeplab 系列。本文**研究了一种在概念上可以减小语义表示中尺度变化的新方法——动态路由。**该框架根据每幅图像的尺度分布，生成依赖于数据的路由。**为此，提出了一种动态选择缩放变换路径的可微门控函数——软条件门。**另外，通过对**门函数赋予预算约束，可以进一步降低端到端的计算成本。**我们进一步放宽了网络层的路由空间，以支持每个转发中的多径传播和跳跃连接，从而带来了巨大的网络容量。为了说明动态特性的优越性，我们比较了几种静态结构，它们可以在路由空间中建模为特殊情况。在 cityscapes 和 pascal 2012上进行了广泛的实验，以说明动态框架的有效性。

在本文中，作者提出了一个用于语义分割的动态网络框架，称为**动态路由（Dynamic Routing）** 。在推理过程中，动态路由能够生成与数据相关的路径，这意味着特定的网络结构随输入而变化。

通过这种方法，具有不同尺度的实例可以分配到相应的分辨率模块。如上图所示，具有不同尺度分布的输入图像将选择不同的路径进行特征变换。本文研究侧重于语义表征，旨在克服尺度的差异，提高网络效率。

引言

语义分割是计算机视觉领域中最基本、也是最具挑战性的任务之一，其目的是根据语义类别对每个像素进行分配。语义分割的问题之一来自于输入之间巨大的尺度差异，例如，微小的对象实例和图片填充的背景材料。此外，大的分布方差也给特征表征和关系建模带来了困难。传统的方法试图通过设计良好的网络体系结构来解决这个问题。

> **多分辨率融合技术[24,28,1,32,19]用于面向细节的特征映射，长程依赖关系用于全局上下文建模[43,35,44,6,31]。**

随着**神经结构搜索(nas)技术**的发展，自动寻找语义分割的有效体系结构已经成为研究的热点[3,22,25]。

> **然而，传统的人工设计网络和基于 nas 的网络都是在单一的网络体系结构中表示所有的实例，缺乏对现实环境中不同尺度分布的适应性。**

图1中给出了一个例子，其中实例的规模变化很大。实例规模变化的地方很多。为此，需要一个更可定制的网络以适应每个图像的尺度方差。在这篇论文中，我们提出了一个新的框架概念——动态路由*(dynamic routing)*。特别是在推理过程中，动态路由生成依赖于数据的转发路径，这意味着具体的网络结构随输入而变化。

> **使用这种方法，可以将具有不同尺度的实例(或背景)分配到相应的解析阶段，以进行定制的特征转换。**

如图1所示，具有不同尺度分布的输入图像将选择不同的路径进行特征变换。动态网络通过丢弃块[38,17,33,36]或剪枝通道[39,20]实现高效的目标识别已有一些研究。

> 不同于以往的是，**本文着眼于语义表征，旨在缓解规模变化，提高网络效率。**

传统的动态图像分类方法[17,33,36]中的路由空间通常局限于分辨率下降的管线(resolution declining pipeline)，不足以进行语义分割。**我们从 auto-deeplab [22]的搜索空间中得到启发**，开发了一个新的包含多个独立单元的容量更大的路由空间。具体来说，不同于**auto-deeplab**，多路径传播和跳过连接在推理过程中被证明是语义切分的十分重要的基础[28,6]。因此，一些经典的网络结构可以作为特殊情况进行比较(图3)。根据动态路由，设计了软条件门，用于根据输入图像选择路径。使用所提出的路由门，每个基本单元，以及分辨率转换路径，可以单独考虑。此外，提出的路由门可以制定为一个可微模块的端到端优化。因此，给定有限的计算预算(例如，浮点运算)，几乎没有贡献的细胞(cell)将在运行中被丢弃。

整体的方法，命名为动态路由，可以很容易地实例化的语义分割。为了阐明它在性能和效率方面优于固定结构，我们在第4.3节进行了广泛的烧蚀研究和详细的分析。实验结果进一步报告了两个著名的数据集，即 cityscapes [9]和 pascal voc 2012[11]。通过简单的比例变换模块，本文提出的动态路由算法可以达到与现有算法相当的效果，但是资源消耗较少。

相关工作

传统的语义切分研究主要集中在根据人类经验设计精细的网络体系结构[24,28,1,43,6]。随着 nas 的发展，有几种方法试图自动搜索静态网络[3,22,25]。不同于以往的工作，本文提出了动态路径选择的方法，根据输入进行合适的比例变换，这方面的研究很少。在此，我们首先回顾了用于语义分割的手工设计体系结构。然后我们给你介绍一些基本的方法。最后，回顾了动态网络的发展历程。

Handcrafted Architectures

近年来，人们对手工架构进行了深入的研究。在语义分割的网络设计方面已经有了一些研究，如 fcn [24]，unet [28]，conv-deconv [26]，segnet [1]。基于精心设计的 fcn [24]和 u 形结构[28]，人们提出了大量的作品，通过捕捉更大的感受野[43,4,5,6,41]或建立像素级关系[44,18,12,31]来建模全局上下文。由于密集预测的高资源消耗，为了提高预测效率，人们提出了一些轻量级的结构，包括 ICNet [42]和 bisenet [40]。总的来说，手工架构旨在利用静态网络中不同阶段的多尺度特性，而不是适应动态输入。

基于 nas 的方法

近年来，神经网络体系结构搜索(nas)已经广泛应用于自动网络体系结构设计[45,27,23,2,13,7]。当涉及到特定领域时，有几种方法试图寻找更适合于语义分割的有效体系结构。

> **特别是 chenet.al。[3]搜索多尺度模块来替换 aspp [5]块。**

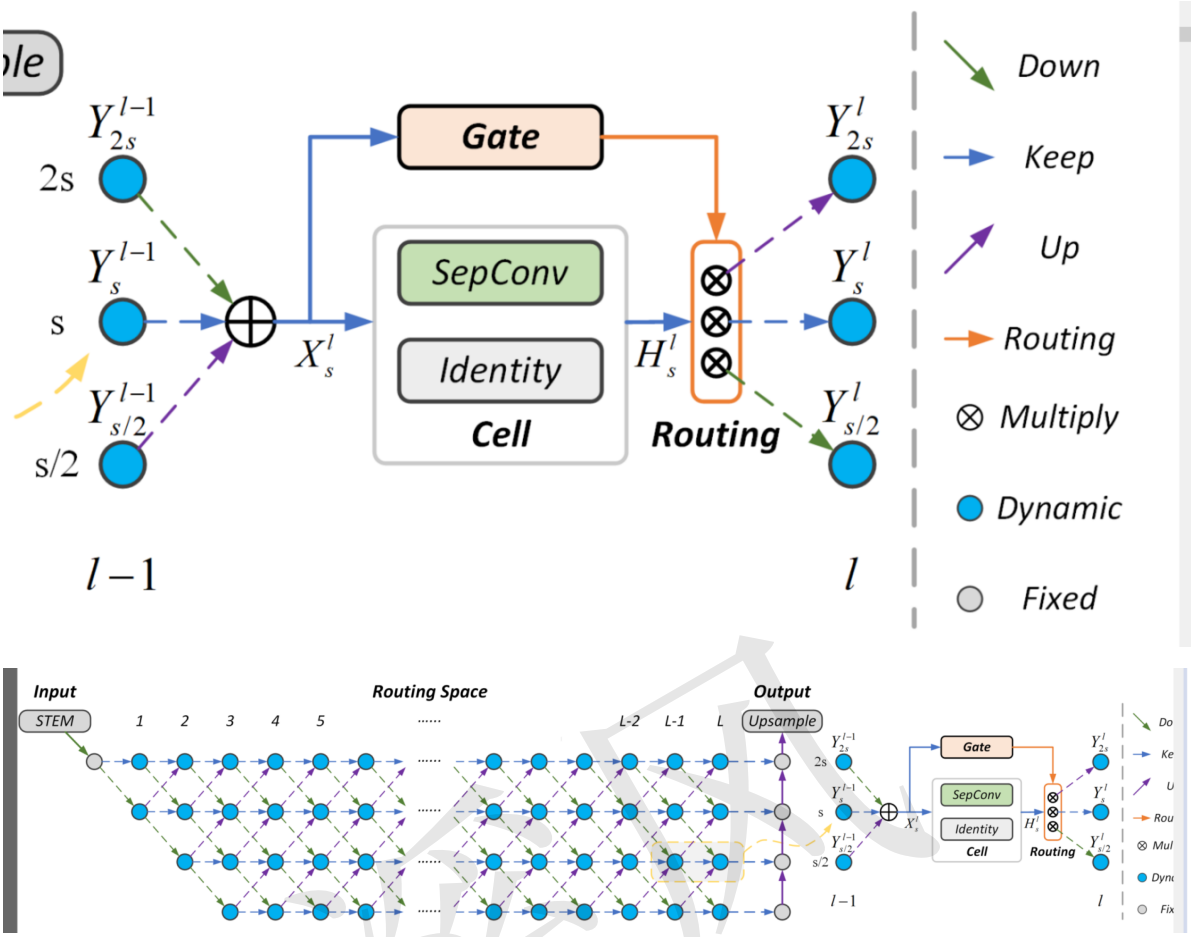
此外 nekrasov 等人 al。[25]利用基于 nas 的方法研究了解码器中辅助cell的路由类型。最近，Auto-DeepLab [22] 被提出用于搜索单一路线来自密集连接的搜索空间。不同于基于 NAS 的方法，搜索单个架构然后重新训练**它，提出的动态路由生成在不搜索的情况下动态地计算前向路径。**

动态网络

动态网络，调整网络结构到相应的输入，已经在计算机视觉领域进行了研究。传统的图像分类方法主要是通过dropping blocks[38,17,33,36]或修剪通道[39,20]进行有效推理。**例如，一个在MSDNet中采用的早期存在的策略是进行资源节约型对象识别，该对象在早期阶段对较容易的输入进行分类并给出输出。**和 skipnet [36]尝试跳过卷积块使用一个基于RL的门控网络。然而，动态路由在规模转换，特别是语义分割方面的研究还很少。为了利用动态特性，本文提出了一种端到端的动态路由框架，以减小输入之间的规模变化。

模型架构

图2。为语义分割提供动态路由框架。为了保持稳定，初始端和最终取样块都是固定的。虚线表示动态路由的替代路径。右: 单元级的动态路由过程。给定前一层的相加输入，首先利用条件门生成激活权值。相应权重高于零的路径被标记为激活，这将被选择用于特征转换。关于网络的更多细节详见第3.4节。最佳彩色效果。



学习动态路由

与静态网络体系结构相比，动态路由在网络容量和性能方面具有优势，并且能够节省预算资源。在这一部分，我们首先介绍设计的路由空间。然后，详细阐述了动态路由框架和约束机制。架构细节将在本节的最后给出。

路由空间(Routing space)

为了释放动态路由的潜力，**我们在相邻层之间提供完全连接的路径，但是有一些先验约束**，例如，单元之间的上采样或下采样跨度，如图2所示。具体来说，根据网络设计中的常见做法，网络的开始是一个固定的3层“STEM”块，这将分辨率降低到1/4比例。在那之后，一个有层的空间被设计为动态路由，称为路由空间。在路由空间中，相邻单元间的比例因子被限制为2，这是基于ResNet的方法中广泛采用的方法。因此，最小比例设置为1/32。在这些约束条件下，每个层次的候选数最多可达4人。

> With these constraints, the number of candidates in each layer is up to 4.

每个候选样本有3种尺度变换路径，即上采样、保持分辨率和下采样。在每个候选者内部，基本的cell用于**特征聚合**，而基本门(gate)用于**路径选择**，如图2所示。层层上采样模块固定在网络的末端以产生预测。关于动态路由过程的更多细节将在第3.2节中解释。不同于 Auto-DeepLab ，在推理阶段，每个节点只选择一条特定的路径，我们进一步放宽了路由空间，以支持多路径路由，并在每个候选节点上进行跳连接。随着更通用的空间，许多流行的架构可以指定为特殊情况，如图3所示。第4.3节进一步进行了定量比较，论证了动态路由的优越性。

路由选择过程(Routing Process)

在给定具有多个节点的路由空间的情况下，分别在每个节点内部采用一个基本单元和一个对应网关来聚合多尺度特征，并选择路由路径。

$$X_s^l = Y_{\frac{s}{2}}^{l-1} + Y_s^l + Y_{2s}^l$$

其中，l表示层数，s表示分辨率。

Cell Operation

对于输入X_s^l，使用卷积和残差连接，其中隐藏状态可表示为

$$H_s^l = \sum_{O^i \in \mathcal{O}} O^i(X_s^l)$$

$\mathcal{O}$表示操作集合，包括 3 × 3 sepConv 和 identity mapping。在这里，每个cell内的操作被用于基本的特征聚合。然后根据激活因子α将生成的特征图转换为不同的尺度。这个过程将在下一节详细阐述。此外，第4.4.1节比较了不同的cell组件。

Soft Conditional Gate

每个路径的概率由网关函数生成，如图2的右图所示。更具体地说，我们在门中采用了轻量级的卷积运算来学习数据依赖向量(the data-dependent vector)G_s^l

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

符号	含义	
G_s^l	输出	
X_s^l	输入	
$\mathcal{F}$	卷积操作	
$\omega_{s,2}^l$, $\omega_{s,1}^l$, β_s^l	卷积权重和偏置	
$\mathcal{G}$	GAP全局平均池化	
σ	ReLU激活函数	
$\mathcal{N}$	BN	

不同于传统的基于 RL方法[36,38,34] ，采用策略梯度更新代理进行离散路径选择，我们提出了软条件网关可微化。为此，利用向量矩阵的特征，设计了一个激活函数

$$\delta(\cdot) = max(0, \mathrm{Tanh}(\cdot))$$

$$\alpha_s^l = \delta(\mathrm{G}_\mathrm{s}^l)$$

这里激活因子$\alpha_s^l \in [0, 1)$。所有$\alpha_s^l > 0$的路径将被保留，从而实现多路径传播。

这意味着路由路径随输入而变化，或者说是依赖于数据。这样，每条路径都可以单独考虑，而不是只选择相对重要的路径进行传播[23,37,22]。此外，在第4.4.2节中研究了不同的激活函数。

$$Y_j^l = \alpha_{s\to j}\mathcal{T}_{s\to j}(\mathrm{H}_s^l)$$

符号	含义	
$\mathcal{T}_{s\to j}$	s到j的分辨率变换，变为1/2, 1, 2	

在推理中，如果所有的路径都被标记为关闭，操作将被删除，以保存计算足迹。回想 eq. 1，这个过程被总结为

$$\mathbf{H}_s^l = \begin{cases} \mathbf{X}_s^l & \sum_j \alpha_{s\to j}^l = 0 \\ \sum_{O^i\in\mathcal{O}} O^i(\mathbf{X}_s^l) & \sum_j \alpha_{s\to j}^l > 0 \end{cases} \tag{5}$$

$$\mathbf{Y}_j^l = \begin{cases} 0 & \sum_j \alpha_{s\to j}^l = 0,\ j \neq s \\ \mathbf{H}_s^l & \sum_j \alpha_{s\to j}^l = 0,\ j = s \\ \alpha_{s\to j}^l \mathcal{T}_{s\to j}(\mathbf{H}_s^l) & \sum_j \alpha_{s\to j}^l > 0 \end{cases} \tag{6}$$

考虑到现实世界中有限的计算资源，我们考虑了有效的动态路由预算约束。让我们重新考虑与预定义的操作相关的计算开销，例如，flops。回想一下 eq. 1,2，和4，我们公式的预期成本内的节点的第二阶段和第一层作为.

通过定义预算约束损失函数来降低参数量。

$$\begin{aligned} \mathcal{C}(\mathrm{Node}_s^l) &= \mathcal{C}(\mathrm{Cell}_s^l) + \mathcal{C}(\mathrm{Gate}_s^l) + \mathcal{C}(\mathrm{Trans}_s^l) \\ &= \max(\alpha_s^l)\textstyle\sum_{O^i\in\mathcal{O}} \mathcal{C}(O^i) + \mathcal{C}(\mathrm{Gate}_s^l) \\ &+ \textstyle\sum_j \alpha_{s\to j}^l \mathcal{C}(\mathcal{T}_{s\to j}) \end{aligned} \tag{7}$$

where Cell_s^l, Gate_s^l, and Trans_s^l indicates the functional operation inside *Cell*, *Gate*, and *Scale Transform*, respectively. Going one step further, the expected cost of the whole routing space could be calculated by

$$\mathcal{C}(\mathrm{Space}) = \sum_{l\le L}\sum_{s\le 1/4} \mathcal{C}(\mathrm{Node}_s^l) \tag{8}$$

Then we formulate the expected resource cost $\mathcal{C}(\mathrm{Space})$ into loss function $\mathcal{L}_\mathrm{C}$ for the end-to-end optimization:

$$\mathcal{L}_\mathrm{C} = \left(\mathcal{C}(\mathrm{Space})/\mathrm{C} - \mu\right)^2 \tag{9}$$

在整个路由空间的实际资源消耗中，μ ∈[0,1]表示设计的衰减因子。根据不同的 μ，每次传播中选择的路径将自适应地限制在相应的预算内。不同预算约束下的网络性能将在第4.4.3节中讨论。**总的来说，网络权值和软条件门都可以在一个统一的框架下通过联合损耗函数进行优化。**

$$\mathcal{L} = \lambda_1\mathcal{L}_N + \lambda_2\mathcal{L}_C$$

> 利用 λ1和 λ2分别平衡网络预测和资源成本期望的优化过程。$\mathcal{L}_N$表示网络想要优化的损失函数。

架构细节

> 从宏观的角度来看，我**们将路由空间的深度设置为16或33，这与广泛使用的 resnet-50和 resnet-101[16]的深度相同，即图2中的总层数 l = 16或33。**

这种设置方便了与基于资源网络的网络相比，后者可以直接使用所提出的路由空间来表示。对于网络中的微节点，我们在"STEM"块中采用三个SepConv 3 × 3，其中卷积核数为64个。**一个步长为的conv1 × 1用于所有的检测s→ s/2路径，既降低了特征分辨率，又加倍了卷积核的数量。**对于所有的 s →2s连接，均采用了随后的双线性上采样，以提高空间分辨率和减少卷积核数目。

此外，一个朴素解码器***(naive decoder)***被设计用于融合最终预测的特征，这些特征在图2中表示为网络末端的灰色节点。具体来说，该算法采用了Conv 1 × 1和双线性上采样相结合的方法来融合解码器中不同尺度的特征。将尺度1/4中的预测向上取样4，生成最终结果。卷积权值按正态分布[15]初始化，偏置初始化为常数1.5。当给定一个预算约束时，我们将输入$\mathrm{X}_{\mathrm{s}}^{l}$下降4倍，以减少网关函数的资源消耗。否则，输入量的分辨率保持不变。

实验结果

在这一部分，我们首先介绍数据集和提议的动态路由的实现细节。然后我们在城市景观数据集上进行大量的消融研究[9]。并对各组分的作用进行详细的分析。最后，与 cityscapes [9]和 pascal voc 2012数据集[11]上的几个基准进行比较，以说明建议方法的有效性和效率。

数据集

城市景观: 城市景观[9]是一个广泛应用的城市景观理解数据集，其中包含19个评价类。数据集包括5000个使用 size1024 × 2048的 fineannotation，它可以分为2975,500和1525个图像，分别用于训练，验证和测试。它还有另外20公里的训练路线，这在我们的实验中是没有用到的。Pascal voc: 我们在2012 pascal voc 数据集[11]上进行实验，该数据集包括20个对象类别和一个背景类。最初的数据集包含1464、1449和1456张图像，分别用于训练、验证和测试。在这里，我们使用[14]提供的增强数据，产生了10582张图像用于训练。

实现细节

在此，优化细节报告方便的实现。为了获得更好的性能，factorλ1in eq. 10被设置为1.0。根据第4.4.3节中不同的预算约束设置 λ2。采用权值为 decay1e-4，动量为0.9的 sgd 进行网络优化。类似于[5,40,31]，我们采用' poly'调度，在每次迭代中，初始速率乘以(1-iter 最大迭代)幂，功率为0.9。在训练阶段，我们随机翻转和缩放每个图像0.5到2.0 × 。根据实验环境，采用不同的初始速率。具体来说，我们设置初始速度为0.05和0.02时，从头开始训练和使用 imagenet [10]预训练，分别。对于城市景观[9]，我们从8个随机768 × 768图像作物中构建每个小批量用于培训。对于 pascal voc 2012[11]，每次迭代采用16个随机512 × 512图像作物进行优化。

动态路由

为了验证动态路由的优越性，我们比较了动态网络与已有的几种结构和从路由空间采集的静态路由。特别是，在具有类似连接模式的路由空间中建立了传统的人工设计的网络以及搜索结构，包括 fcn-32s [24]、 u-net [28]、 deeplabv3[5]、 hrnetv2[32]和 autodeeplab [22]，如图3所示。为了公平比较，我们通过给 eq. 9中的损失函数赋予不同的预算约束，将计算开销与这些方法对齐。因此，可以生成三种类型的动态网络(详情请参阅第4.4.3节)，在表格1中表示为 dynamic-a、 b 和 c。与手工搜索的体系结构相比，动态路由算法在相似的成本下获得了更好的性能。例如，假设预算约束分别为45g、55g 和65g，动态-a、 b 和 c 分别为5.8% 、2.2% 和2.1% over the modeled DeepLabV3, U-Net, and HRNetV2, 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

在此基础上，**提取保留了95% 以上正向推理的动态网络基本路由**，构造出对应的公共网络。一般网络的连接模式如图4所示。在表1中，我们进一步比较了动态网络与通用体系结构(common-a、 b 和 c)。具体来说，使用动态路由框架，动态网络在每个预算约束下都会比静态通用网络有更好的性能。

我们观察到图4中公共网络的连接路由与几种已知的结构，如人工设计的 u-net [28]和基于 nas 的 auto-deeplab [22]相似。特别是前端采用下采样操作，后端采用上采样操作。此外，对于物体细节(如图1所示)，需要低级阶段的高分辨率特征，这可能导致更好的性能。

成分分析

为了揭示所提方法中每个组件的影响，我们将在本节中逐步分解我们的方法。首先，将详细讨论细胞内的组织结构。然后我们研究了所提出的软条件门的激活函数。不同资源预算的影响将在最后进一步说明。

为了揭示所提方法中每个组件的影响，我们将在本节中逐步分解我们的方法。首先，将详细讨论细胞内的组织结构。然后我们研究了所提出的软条件门的激活函数。不同资源预算的影响将在最后进一步说明。

cell component

为了和先前的架构进行公平的比较，只有基本的卷积操作和身份映射(identity mapping)。几个典型操作的实验结果，包括瓶颈[16]，mbconv [29]，sepconv [8]在表2中。我们发现叠加**两个 sep 3 × 3进行特征变换时**，动态网络的性能最好，**重运算对增益的贡献不大。**我们猜测这可能是因为路由体系结构比较繁重的操作起着更重要的作用。实际上，当分辨率为1/4时，我们也进行了较大内核(例如，sep conv5 × 5)的实验，但只发现了0.2% 的绝对增益。因此，为了简便起见，本文只使用了 sep3 × 3。

Activation Function

在第3.2.2节中，我们进一步比较了几种常用的软条件门激活函数。

> Firstly, all of the paths in the routing space are fixed with no difference to formulate our baseline, namely the 'Fix' in Tab.3.

首先，路由空间中的所有路径都是固定的，没有任何差异来制定我们的基线，即表格中的"固定"。然后，激活函数 δ直接替换为表3中的候选函数。我们发现max(0，tanh)比其他的表现更好。此外，将每个单元中的三条路由路径放在一起考虑的SoftMax激活函数的性能低于单独考虑的路由路径，例如，sigmoid和max (0，tanh)。这意味着每条路径都应该在软条件门中解耦。然后，激活 factorα > 0的路径将在这个正向推理过程中被保留，如第3.2.2节所阐述的。

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

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

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.

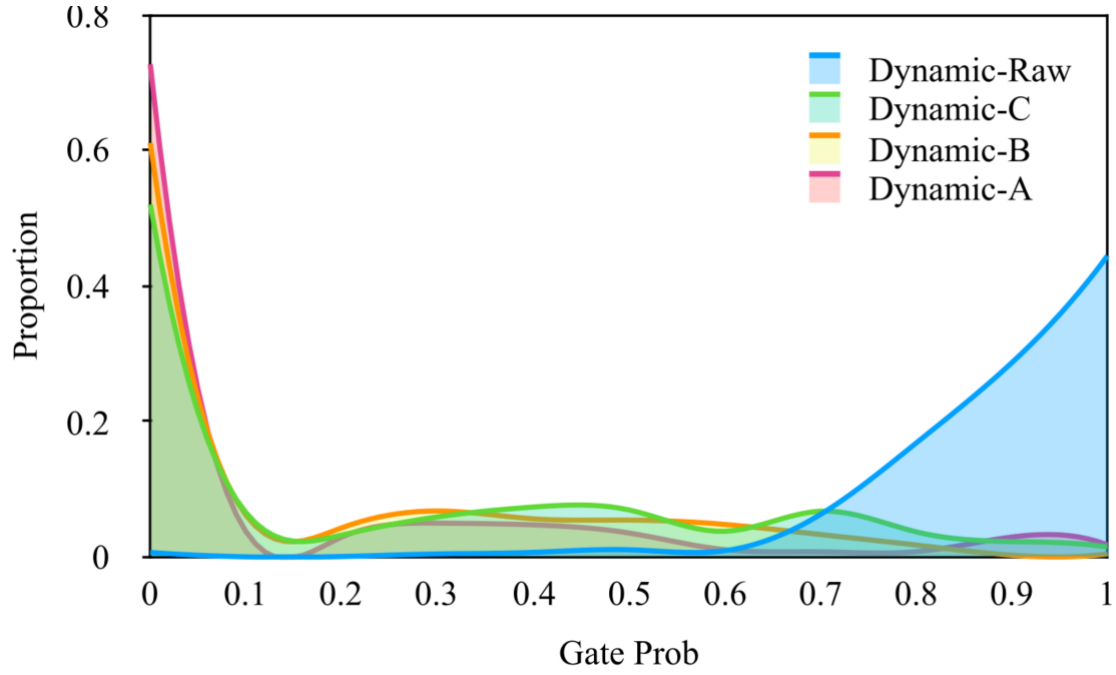
Method	λ_2/μ	mIoU(%)	FLOPs(G)	Params(M)
Network-Fix	-	74.5	103.1	15.3
Dynamic-A	0.8/0.1	72.8	44.9	17.8
Dynamic-B	0.5/0.1	73.8	58.7	17.8
Dynamic-C	0.5/0.2	74.6	66.6	17.8
Dynamic-Raw	0.0/0.0	76.1	119.5	17.8

Resource Budgets

利用设计的门函数，通过调整系数 λ2和 μ，给出了不同的资源预算。如表4所示，如果给定不同的预算约束，路由框架将生成几种类型的动态网络(dynamic-a、 b 和 c)。与无资源预算的原始动态网络(dynamicraw)相比， Dynamic-C 的成本降低到55.7% ，性能下降很小。同时， Dynamic-C 在有效性和效率方面仍然优于全连网(fully-connected Network-Fix)。在更强的约束下。资源成本可以进一步降低到37.6% (Dynamic-A) 。

此外，我们提出了路由激活概率的分布图5。很明显，大部分路径都是动态原始的。如果给定资源预算，路线的不同比例将被降低，这可以从图5中的分布中了解到。

图5。动态网络中路由激活概率的分布。在 dynamic-raw 中，大多数路径倾向于保留，不受预算限制。在给定资源预算的情况下，不同比例的线路将在动态 a、 b 和 c 中被占用。



因此，在推理过程中，动态路由将cut out 无用的路径和cell 。我们发现 $FLOP_{max}$和$FLOP_{min}$之间的差距相对较小(10%)，这可以归因于预算约束的作用。实际上，我们也尝试了不同类型的变异系数，以扩大差距，但发现效果不佳。

在数据集上的实验结果

cale and flipping strategy are used in *test* set but dropped in *val* set. We report FLOPs with input size 1024 × 2048.

Method	Backbone	mIoU$_{test}$(%)	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

parisons with previous works on the PASCAL VOC 2012. mIoU$_{test}$ and mIoU$_{val}$ denote performance on *test* set and *val* set 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	mIoU$_{test}$(%)	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

结论

在这项工作中，我们提出了动态路由的语义分割。与以往工作的关键区别在于，我们根据每幅图像的尺度分布生成了依赖于数据的前向路径。为此，提出了软条件门技术，以端到端的方式选择规模转换路径，在给定资源预算的情况下，学会放弃无用的操作以提高效率。通过大量的消融实验，证明了动态网络相对于静态网络结构的优越性，可以在设计的路由空间中进行建模。在 cityscapes 和 pascal voc

2012上的实验证明了该方法的有效性，该方法的性能可与现有技术相媲美，但消耗的计算资源要少得多。

参考文献

Dynamic Routing-中科院&西交&旷视（孙剑团队）提出用于语义分割的动态路由网络，精确感知多尺度目标，代码已开源！