基于深度嵌入学习的高效图像分割

刘云 [1], 姜鹏涛 [1], Vahan Petrosyan[2], 李仕杰 [1], 边佳旺 [3], 张乐 [4], 程明明 [1*]

[1] 南开大学 [2] KTH Royal Institute of Technology [3] University of Adelaide [4] ADSC

摘要

图像分割已经被探索了很多年，仍然是一个关键的视觉问题。一些有效或准确的分割算法已被广泛用于许多视觉应用中。但是，很难设计出既高效又准确的图像分割器。在本文中，我们提出了一种称为深度嵌入学习（Deep Embedding Learning, DEL）的新方法，该方法可以有效地将超像素转换为图像分割。从 SLIC 超像素开始，我们训练了一个全卷积神经网络，以学习每个超像素的特征嵌入空间。所学习到的特征嵌入可用于度量两个相邻超像素之间的相似性。凭借基于深度学习的相似性，我们可以将超像素直接合并为大块区域。BSDS500 和 PASCAL Context 数据集上的评测结果表明，我们的方法在效率和有效性之间取得了良好的折衷。具体来说，与 MCG 相比，我们的 DEL 算法可以实现可比的分割结果，但运行速度比它快得多，分别为 11.4fps 与 0.07fps。[1]

1 引言

图像分割旨在将图像划分为较大的感知区域，其中每个区域内的像素通常属于同一物体、部分物体或较大的背景区域，而每个区域内的颜色、渐变、纹理和强度等特征的特征差异很小。图像分割已广泛用于各种中层和高层的视觉任务，例如拟物性采样[2; 3]、跟踪[4]、物体检测/识别[5]、语义分割[6]等。这项技术已经被研究了很多年，但是仍然是计算机视觉的一个主要挑战。

通常，图像分割涉及两个方面，即分割结果的可靠性和应用程序的效率。可以将适当的图像分割用作输入，以显著提高许多视觉任务的性能。此外，计算时间和内存消耗决定了它是否适合许多实际应用，因为图像分割通常在其他视觉应用中用作预处理步骤。但是，现有方法难以平衡分割精度和计算时间。尽管 MCG [2]和 gPb [7] 可以生成高质量的分割，但它们速度太慢了，从而无法应用于对时间敏感的任务。EGB [8]的运行时间几乎与图像像素数成正比，因此非常快。但是它的准确性特别差，尤其是在基于区域的评测指标上，因此不能满足当今的视觉任务的要求。HFS [9]可以进行实时分割，但是，它的分割结果并不令人满意，尤其是在基于区域的评测指标上。很难设计出可以同时满足有效性和效率要求的理想图像分割算法。

与图像分割类似但略有不同，超像素生成通常是指图像过分割。它将输入图像分割成小的、规则的、紧凑的区域，这与通过图像分割技术生成的大的感知区域不同。过度割通常具有很强的边界一致性，并且产生的超像素的数量易于控制。由于超像素方法通常被设计为生成小区域，因此不宜直接使用它们来生成大区域。但是，超像素算法为图像分割提供了一个良好的开端。

在本文中，我们旨在设计一种图像分割算法，该算法可以在效率和效果之间取得良好的平衡。考虑到效率，我们的工作从快速的超像素生成方法开始，即

*程明明（cmm@nankai.edu.cn）是通讯作者。

[1]本文是 IJCAI 2018[1]论文的中译版。

SLIC 的 GPU 版本 [10; 11]。在过去的几年中，卷积神经网络已经提升了许多计算机视觉任务。由于基于深度学习的特征可以表示比手工设计的特征丰富得多的信息，因此我们训练了一个全卷积神经网络来学习深层特征嵌入空间，该空间对每个超像素进行基于深度学习的编码。我们引入了深度嵌入度量，该度量将相邻超像素的特征嵌入向量转换为相似度值。每个相似度值表示两个相邻超像素属于同一区域的概率。通过这种方式，我们可以端到端地训练深度嵌入空间，以学习每对超像素之间的相似性。本文提出了一种用于嵌入学习的新的网络，该网络结合了底部的精细细节和顶部的高级信息的特征。如果学习到的相邻超像素之间的相似度大于阈值，则将其合并为大图像块。由于深度学习特征的强大表示能力，这种简单的合并操作可以比 HFS 的层次合并获得更好的性能。

我们在 BSDS500 [7]和 PASCAL Context [12] 数据集上进行了广泛的实验，以评测提出的图像分割算法。为了评测我们的算法在应用程序中的表现，我们将分割结果应用于 PASCAL VOC2007 数据集[13]上的拟物性采样。评测结果表明，我们的算法在效率和有效性之间取得了良好的折衷。具体而言，与最新方法相比，我们提出的 DEL 可以达到可比的分割结果，但比它们快得多，例如 DEL 的 11.4fps 与 MCG 的 0.07fps。这意味着 DEL 有潜力在许多实际应用中使用。本文的代码可从 https://github.com/yun-liu/del 获得。

1.1 相关工作

在过去的几十年中，研究人员为该领域做出了许多有益的贡献。由于篇幅所限，在这里我们只回顾一些典型的算法。Shi 等人[14]将图像分割视为一个图割问题，提出了一种新颖的归一化切割准则，以测量每个区域内的相似度和不同区域之间的相似度。Comaniciu 等人[15]根据传统模式识别中的均值漂移技术提出了均值漂移的图像分割算法。Felzenszwalb 等人[8]提出了一种基于图的高效算法 EGB。基于边缘的方法 gPb [7]将多尺度局部特征和光谱聚类相结合以预测边缘，然后使用定向分水岭变换算法将这些边缘转换为分割。Pont-Tuset 等人[2]组合了多尺度分层区域，以获取准确的分割（MCG）。

随着超像素生成[10]的发展，一些方法尝试从超像素开始，然后将这些超像素合并成感知区域。ISCRA [16]使用 gPb 生成高质量的超像素，用各种特征（包括颜色、纹理、几何背景、SIFT、形状和边界）为相邻的超像素学习相异度得分。程等人[9]使用精心选择的可并行化特征，通过对超像素进行分层合并首次构建了一个实时图像分割系统。在每个合并阶段都会重新训练所选特征的组合权重。我们提出的方法也属于这一类，但是我们的方法使用深度卷积神经网络为该任务提取强大的基于深度学习的特征表示，从而可以获得更好的感知区域。我们将在下一部分中详细介绍我们的方法。

2 方法

我们的方法从 SLIC 超像素[10]开始。我们首先训练一个深度卷积神经网络，以学习相邻超像素之间的相似性，然后使用学习到的相似性将它们直接合并。在本节中，我们将详细描述算法的五个组成部分，依次是超像素生成、特征嵌入学习、网络架构、超像素合并和实现细节。

2.1 超像素生成

图像分割算法将像素划分为较大的感知区域，其中相同区域中的像素比不同区域中的像素具有更大的相似性。但是，当使用相似距离度量对像素进行聚类时，由于算法的运行时间与图像中的像素数高度相关，因此算法通常会花费过多时间。此外，算法在直接合并像素时缺乏鲁棒性。考虑到这两个方面，我们的算法从快速超像素生成方法 SLIC [10] 开始，该方法基于 k-means 聚类算法。超像素的数量远少于原始像素，因此可以提高效率。一个超像素是一个很小的区域，因此比单个像素更鲁棒。

通常，超像素算法不能直接应用于图像分割，因为与超像素不同，大的感知区域通常不规则且与图像中的全局信息有关。受 HFS 启发，HFS 从超像素开始，并使用经过精心设计的功能将它们分层地合并，我们的算法学习了相邻超像素之间的相似性度量。由于其简单性和效率，SLIC 是许多最新算法中一种广泛使用的超像素算法。我们选择 SLIC 的 GPU 版本 gSLIC [11]作为我们方法的开始。为了平衡运行时间和生成

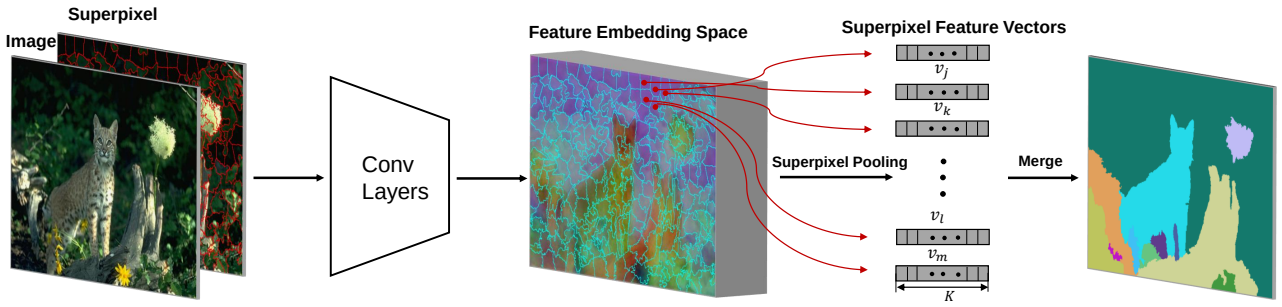

图 1: DEL 图像分割算法的流程。

的超像素的边界一致性，我们控制每个超像素包含约 64 个像素。假设我们现在有 M 个超像素用于图像 I。生成的超像素集表示为 $\mathcal{S} = \{S_1, S_2, \cdots, S_M\}, S_i = \{1, 2, \cdots, |I|\}^{|S_i|}$。

2.2 特征嵌入学习

生成超像素后，我们开始训练一个深度卷积神经网络以学习特征嵌入空间。如图 1中所示，我们在特征嵌入空间上执行池化操作以为每个超像素提取特征向量 $\vec{v} = \{\vec{v}_1, \vec{v}_2, \vec{v}_3, \cdots, \vec{v}_M\}$。每个特征向量是在超像素的相应区域中学习到的深度特征图的平均值。它可以表述为：

$$\vec{v}_i = \frac{1}{|S_i|} \sum_{k \in S_i} \vec{x}_k, \tag{1}$$

其中 $\vec{x}_k$ 表示第 i 个超像素对应区域内的特征向量。我们称这种池化操作为*超像素池化*。在我们的设计中，每个特征嵌入向量 $\vec{v}_i$ 是 64 维的。超像素池化层相对于输入 $\vec{x}_k$ 的反向传播函数可以写成

$$\frac{\partial L}{\partial \vec{x}_k} = \sum_{S_i \in \mathcal{S}} 1_{\{k \in S_i\}} \cdot \frac{1}{|S_i|} \cdot \frac{\partial L}{\partial \vec{v}_i}, \tag{2}$$

其中 $1_{\{k \in S_i\}}$ 是一个指示函数。

我们设计了一个距离度量来测量相邻超像素之间的相似性。所提出的距离度量可以表述为

$$d_{ij} = \frac{2}{1 + exp(\|\vec{v}_i - \vec{v}_j\|_1)}. \tag{3}$$

相似度 $d_{i,j}$ 的取值范围为 $[0, 1]$。当 v_i 和 v_j 相似时，其值接近 1；而当 v_i 和 v_j 极其不同时，其值接近 0。由于建立了距离度量，因此我们考虑相似度损失函数，如下所示：

$$\begin{aligned} L = -\sum_{S_i \in \mathcal{S}} \sum_{S_j \in \mathcal{R}} [(1-\alpha) \cdot l_{ij} \cdot log(d_{ij}) \\ + \alpha \cdot (1 - l_{ij}) \cdot log(1 - d_{ij})], \end{aligned} \tag{4}$$

其中 $l_{ij} = 1$ 表示 v_i 和 v_j 属于同一区域，而 $l_{ij} = 0$ 表示 v_i 和 v_j 属于不同的区域。$\mathcal{R}$ 是超像素 S_i 的相邻超像素集。$\alpha = |Y_+|/|Y|$，表示真值中属于相同区域的超像素对的比例。我们使用此参数来平衡正负样本比例。

利用这种相似度损失函数，我们可以以一种端到端的方式学习特征嵌入空间。在相同的真值区域中的超像素对之间的相似性将被期望大于属于不同区域的超像素对之间的相似性。在下一步中，我们将使用学习到的相似性距离度量来合并这些超像素。

2.3 网络架构

在本节中，我们介绍用于学习特征嵌入空间的网络架构。我们的网络基于 VGG16 网络[17]构建，并受到最近作品 [18; 19]的启发。根据池化层，VGG16 中的卷积层可以分为五个卷积阶段。如图 2中所示，我们剪切了 VGG16 网络中的 *pool5* 层和全连接层。由于 *conv5* 阶段的侧输出分辨率较低，因此我们将 *pool4* 的步幅从 2 修改为 1，并使用孔洞算法[20]来保持第五阶段的卷积的响应野大小与原始 VGG16 网络相同。我们认为，随着网络的加深，学习到的特征变得越来越粗糙。精细特征包含更多的详细信息，而粗糙特征则表示全局信息。五个阶段的特征被拼接起来，将粗糙的全局信息与精细的局部信息相结合。

具体来说，我们在第 1-5 阶段分别连接一个具有 32、64、128、256、256 个输出通道的 3×3 卷积层。

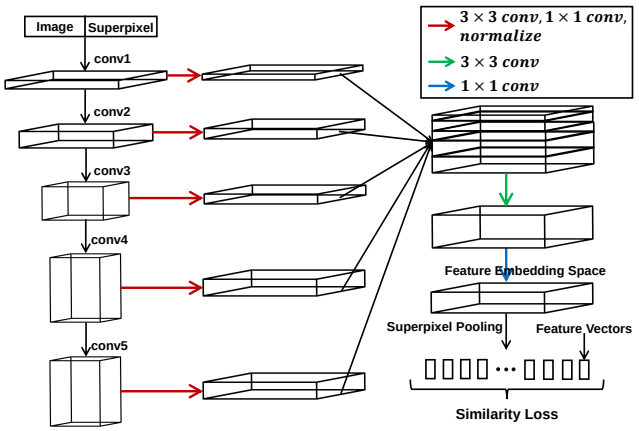

图 2: 特征嵌入学习的网络架构。

在这个 3×3 的卷积层之后，第 1-5 阶段再分别连接一个具有 32、64、64、128、128 个输出通道的 1×1 卷积层。由于不同卷积阶段的特征（数值）尺度不同，因此将多个阶段的特征直接拼接起来将使某些阶段的特征变得毫无意义。因此，我们使用[21]中引入的 L2 归一化技术对不同阶段的响应进行归一化。归一化后，我们拼接各个阶段的特征图，然后跟随一个具有 256 个输出通道的 3×3 卷积层。最后，我们使用一个 1×1 卷积层来获得 64 维特征嵌入空间。如第 2.2节所述，我们将特征嵌入合并到与超像素对应的特征向量，然后使用所提出的相似度损失函数来训练网络。

2.4 超像素合并

深度神经网络学习到的相邻超像素之间的差异度被用于将超像素合并为感知区域。我们设置一个阈值以确定是否应该合并两个相邻的超像素。超像素合并的算法伪代码显示算法 1中。为了提高合并效率，我们在 EGB [8]中提出的的数据结构 *universe*。与 HFS [9]中的分层合并策略不同，我们仅执行一次合并操作。HFS 使用一些底层特征的线性组合，并在每个合并阶段重新训练组合权重。DEL 的单阶段合并也可以大大超过 HFS。我们将在实验部分中展示详细信息。

2.5 实现细节

我们的网络基于 Caffe 实现，Caffe 是一种广泛使用的深度学习框架。一般来说，分割区域通常是指物体、部分物体或部分背景。因此，我们首先在 SBD 数据集[22]上对网络进行语义分割任务的预训练，以获取网络的语义信息。预训练通过用语义分割任务的分类层替换特征嵌入空间来调整网络。

算法 1 DEL 的超像素合并算法

输入： 图像 I, 差异度 $f = (1 - d)$, 阈值 T, 超像素 $\mathcal{S} = \{S_1, S_2, \cdots, S_M\}$

构造 $\mathcal{R} = \{R_1, R2, \cdots, R_M\}$, 其中 R_i 是 S_i 的相邻超像素的集合

for each $S_i \in \mathcal{S}$ **do**

  for each $S_j \in R_i$ **do**

    if $f_{i,j} < T$: **then**

      $S_i \leftarrow S_i \cup S_j, \mathcal{S} \leftarrow \mathcal{S} \backslash S_j$

      更新 $\mathcal{R}$

    end if

  end for

end for

输出： 分割 $\mathcal{S}$

然后，我们微调用于特征嵌入空间的预训练模型。我们使用随机梯度下降（SGD）技术来优化神经网络。基本学习率设置为 1e-5。我们使用 0.0002 的权重衰减和 5 的批处理大小。使用 *step* 的学习率策略，并且针对 *step size* 为 8000 的 SGD 总共运行 10000 次迭代。如深度度量学习中建议的那样，特征嵌入层的学习率将被设置为大于基本的卷积层。

事实证明，数据增强对于深度学习非常重要。在由 300 张 *trainval* 图像和 200 张 *test* 图像组成的 BSDS500 数据集 [7]上训练特征嵌入模型时，我们了增强 *trainval* 集。图像被旋转到 16 个角度，并且在每个角度水平翻转。然后，我们从转换后的图像中裁剪出最大的矩形，从而生成 9600 张训练图像。在分为 7605 张 *trainval* 图像和 2498 张 *test* 图像的 PASCAL Context 数据集[12]上进行训练时，我们仅水平翻转 *trainval* 图像进行训练，因为该数据集中的图像数量已经足够多。

3 实验

在本节中，我们首先在 BSDS500 数据集[7]和 PASCAL Context 数据集[12]上评测 DEL 方法以进行图像分割。为了评测应用程序中的分割质量，我们

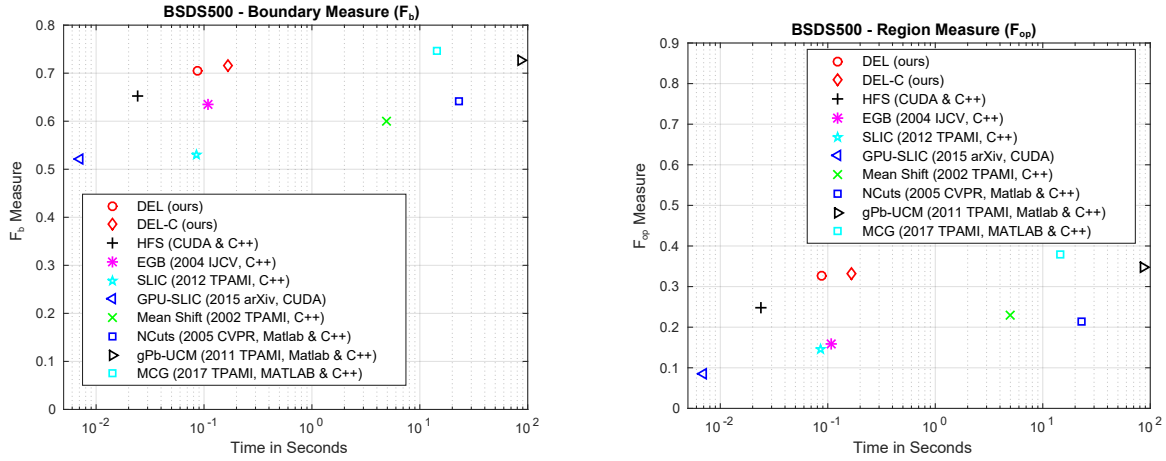

图 3: BSDS500 数据集上的评测结果。**左边**：边缘度量；**右边**：区域度量。

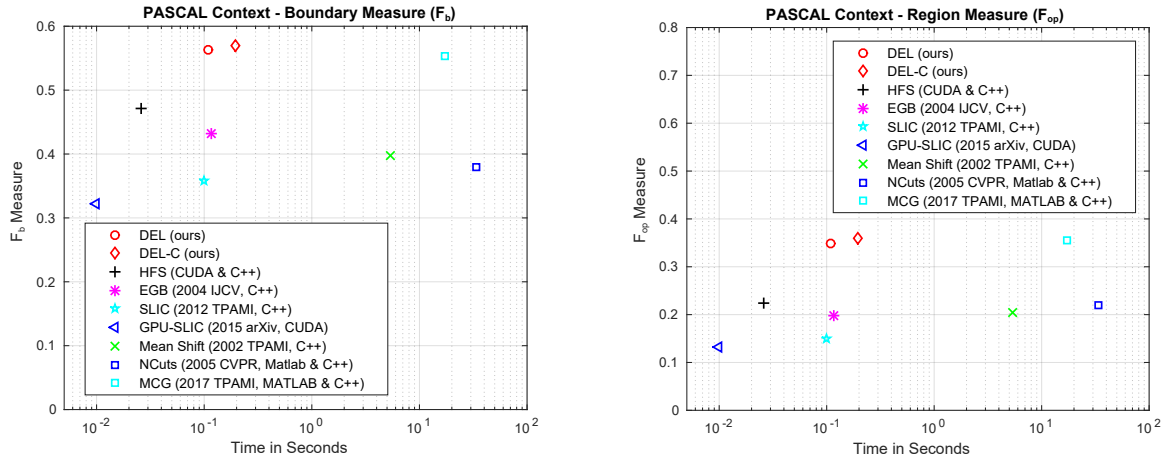

图 4: PASCAL Context 数据集上的评测结果。**左边**：边缘度量；**右边**：区域度量。

使用分割的区域在 PASCAL VOC2007 数据集[13]上进行拟物性采样。为了评测图像分割，我们使用了公开可用的基准 SEISM [23]。最佳数据集尺度（Optimal Dataset Scale, ODS）通常是指为整个数据集选择最佳参数使得整体效果最好，而最佳图像尺度（Optimal Image Scale, OIS）是指为每张图像选择最佳参数使每张图像的性能最佳。我们在 ODS 和 OIS 上报告了边界 F-measure（F_b）和区域 F-measure（F_{op}）。对于拟物性采样的评测，我们报告不同采样数量下的检测召回率（Detection Recall, DR）。我们将 DEL 与一些最新的分割算法进行了比较，包括 EGB [8]、Mean Shift [15]、NCuts [24]、gPb-UCM [7]、MCG [2]、SLIC [10]、GPU-SLIC [11]和 HFS [9]。除了 SLIC 的 GPU 版本之外，我们还使用 SLIC 的 CPU 版本为 DEL 生成超像素，我们将其称为 DEL-C。

Methods	Boundary		Region		Time (s)
	ODS	OIS	ODS	OIS	
DEL-Max	0.703	0.738	0.323	0.389	0.088
DEL-*conv5*	0.667	0.695	0.278	0.343	0.070
DEL-EGB	0.662	0.686	0.305	0.325	0.091
DEL	0.704	0.738	0.326	0.397	0.088
DEL-C	0.715	0.745	0.333	0.402	0.165

表 1: BSDS500 数据集上的消融实验。

3.1 消融实验

BSDS500 [7]是图像分割、过分割和边缘检测的标准数据集。我们使用此数据集来评测每个 DEL 组件的不同选择。第一个 DEL 的变体表示为 DEL-Max，用最大值操作替换超像素池化中的平均操作，其他

DEL 的组件保持不变。第二个变体 DEL-*conv5* 仅使用 VGG16 网络的最后一个卷积层（*conv5*）。第三种变体 DEL-EGB 通过将每个超像素视为图割问题中的一个结点来应用 EGB 的合并策略。

评测结果汇总在表 1中。与原始 DEL 相比，这些变体的性能更差。它表明 DEL 组件的原始的选择是合理的。例如，我们在 DEL 中提出的网络架构既可以捕获细节信息，也可以捕获粗糙信息，而 DEL-*conv5* 的简单设计仅使用了粗糙信息。因此，DEL 比 DEL-*conv5* 好得多。此外，EGB 对于超像素合并似乎毫无用处。最大值池化比平均池化稍差。这符合我们的直觉，即最大值池化和平均池化通常具有相似的效果。

3.2 BSDS500 数据集上的评测

由于 ODS F-measure 是最重要的分割指标，因此我们在图 3 中显示了 ODS F-measure 与运行时间的关系。尽管我们提出的 DEL 并没有达到最佳性能，但是它在效率和有效性之间取得了很好的权衡。在这些方法中，最快的方法是可以实时运行的 HFS [9]。但是，它的性能很差，尤其是对于区域评测指标而言。因此，尽管速度很快，它仍无法满足当今的视觉任务。SLIC [10]和 GPU-SLIC [11]似乎在图像分割方面遇到了困难。这符合我们的直觉，即过分割方法不适用于图像分割。从 GPU-SLIC/SLIC 到 DEL/DEL-C 的改进证明了我们的深度嵌入特征学习方式的有效性。有趣的是，GPU-SLIC 的性能略低于 SLIC，而 DEL 的性能也略低于 DEL-C。由于 GPU-SLIC 的效率高（尽管效果略差），我们选择 DEL 作为默认设置。用更准确的超像素生成方法替换 SLIC 可能会产生更好的性能。DEL 提供了从超像素到图像分割的转换器。新的超像素技术将以这种方式有益于图像分割。尽管 MCG [2]获得了准确的结果，但它的低速限制了它在许多视觉任务中的应用。请注意，由于 MCG 不是可并行化的算法，因此不能直接实现 GPU 版本的 MCG。

数值比较总结在表 2中。DEL 的 ODS F_b 和 F_{op} 分别比 HFS 高 5.2% 和 7.7%。在速度方面，HFS 达到 41.7fps，而 DEL 达到 11.4fps。从 HFS 到 DEL 的精度提高对于许多应用而言都很重要。与 EGB 相比，DEL 在准确性和速度上均具有更好的性能。DEL 可以产生与最新的性能相当的结果，但速度要快得多。

Methods	Boundary		Region		Time (s)
	ODS	OIS	ODS	OIS	
HFS	0.652	0.686	0.249	0.272	0.024
EGB	0.636	0.674	0.158	0.240	0.108
SLIC	0.529	0.565	0.146	0.182	0.085
GPU-SLIC	0.522	0.547	0.085	0.132	0.007
MShift	0.601	0.644	0.229	0.292	4.95
NCuts	0.641	0.674	0.213	0.270	23.2
gPb-UCM	0.726	0.760	0.348	0.385	86.4
MCG	0.747	0.779	0.380	0.433	14.5
DEL	0.704	0.738	0.326	0.397	0.088
DEL-C	0.715	0.745	0.333	0.402	0.165

表 2: BSDS500 数据集上的评测结果。

Methods	Boundary		Region		Time (s)
	ODS	OIS	ODS	OIS	
HFS	0.472	0.495	0.223	0.231	0.026
EGB	0.432	0.454	0.198	0.203	0.116
SLIC	0.359	0.409	0.149	0.160	0.099
GPU-SLIC	0.322	0.340	0.133	0.157	0.010
MShift	0.397	0.406	0.204	0.214	5.32
NCuts	0.380	0.429	0.219	0.285	33.4
MCG	0.554	0.609	0.356	0.419	17.05
DEL	0.563	0.623	0.349	0.420	0.108
DEL-C	0.570	0.631	0.359	0.429	0.193

表 3: PASCAL Context 数据集上的评测结果。

因此，DEL 实现了有效性与效率之间的良好折衷。这使得 DEL 适用于许多高级视觉任务。我们在图 6中展示了一些定性比较。我们可以看到，DEL 可以适应复杂的场景并产生更准确和规则的分割区域。

3.3 PASCAL Context 数据集上的评测

PASCAL Context 数据集[12]包含 540 个用于语义分割的类别。由于整个图像都按像素进行标记，因此可以用于评测图像分割方法。通过连通域标记将语义分割标注转换为图像分割区域。我们在 *trainval* 集上训练模型，并在 *test* 集上进行测试。由于测试图像更多，因此此数据集比 BSDS500 更具挑战性。

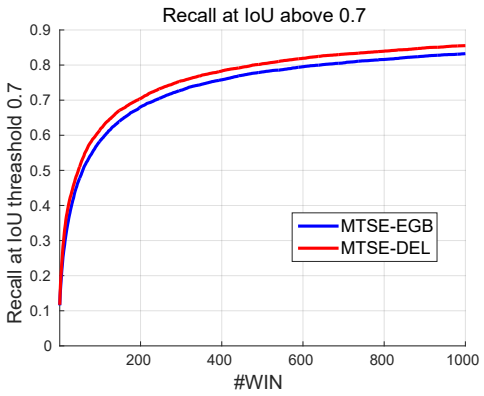

图 5: PASCAL VOC2007 数据集上的关于拟物性采样的评测。

图 6: 一些定性比较。第一列展示了 BSDS500 数据集中的原始图像，后四列分别展示了 EGB, HFS, MCG 和我们的 DEL 生成的结果。

我们将评测结果汇总在图 4中。DEL 和 DEL-C 比 MCG 具有更好的性能。而且，DEL 比 MCG 快 160 倍。可以看到，DEL 在精度和运行时间之间具有良好的权衡。我们在表 3中列出数值结果。在边界度量和区域度量方面，DEL 分别比 HFS 高 9.1% 和 12.6%。这表明我们学习到的深度特征比 HFS 中使用的手工设计特征更有效。因此，这项工作是采用基于深度学习的特征进行通用图像分割的良好开端。

3.4 拟物性采样

拟物性采样对于一系列中高层视觉任务是必需的，例如物体检测[25] 和语义实例分割[26]。已经有许多拟物性采样算法被提出，并且这些方法的相当一部分是基于图像分割的。为了在实际应用中评估我们提出的 DEL，我们将其应用于拟物性采样。MTSE [27]使用 EGB 的分割区域来改善通过其他拟物性采样方法生成的物体边界框的位置。如[27]中所示，当将 MTSE 用作后处理步骤时，BING 算法 [28]在性能上有最显著的提高。因此，我们用 DEL 代替了 MTSE 中的 EGB，以改善 BING 产生的物体边界框。我们在图 5中展示了 IoU 重叠率为 0.7 时的检测召回率与不同物体边界框数量的关系。可以看到，通过我们的 DEL 分割，MTSE 有了显著改善。更多的应用实验不在本文讨论范围之内，但是关于拟物性采样的评测证明了 DEL 在实际应用中的有效性。

4 总结

在本文中，我们提出了一种基于深度学习的图像分割算法。具体来说，我们首先使用快速 SLIC 算法生成输入图像的超像素。然后，学习对每个超像素的高层和低层表示进行编码的深度嵌入特征空间。我们提出一种相似性度量，将学习到的嵌入向量转换为相似性值。根据相似度值执行简单的超像素合并，以获得感知区域。我们提出的 DEL 方法在效率和有效性之间取得了很好的权衡。这使得 DEL 有潜力应用于许多视觉任务。将 DEL 应用于拟物性采样，可以显著地提高所生成物体边界框的质量。将来，我们计划在其他应用中探索 DEL，例如 [2; 4; 5]。

致谢

这项研究得到了国家自然科学基金委员会（项目编号 61620106008、61572264）、华为创新研究计划和中央大学基础研究基金的支持。

参考文献

[1] Y. Liu, P.-T. Jiang, V. Petrosyan, S.-J. Li, J. Bian, L. Zhang, and M.-M. Cheng, “Del: Deep embedding learning for efficient image segmentation,” in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2018.

[2] J. Pont-Tuset, P. Arbeláez, J. T. Barron, F. Marques, and J. Malik, “Multiscale combinatorial grouping for image segmentation and object proposal generation,” *IEEE TPAMI*, vol. 39, no. 1, pp. 128–140, 2017.

[3] Z. Zhang, Y. Liu, X. Chen, Y. Zhu, M.-M. Cheng, V. Saligrama, and P. H. Torr, “Sequential optimization for efficient high-quality object proposal generation,” *IEEE TPAMI*, 2017.

[4] S. Wang, H. Lu, F. Yang, and M.-H. Yang, “Superpixel tracking,” in *IEEE ICCV*. IEEE, 2011, pp. 1323–1330.

[5] M. Juneja, A. Vedaldi, C. Jawahar, and A. Zisserman, "Blocks that shout: Distinctive parts for scene classification," in *CVPR*, 2013, pp. 923–930.

[6] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *IEEE TPAMI*, vol. 35, no. 8, pp. 1915–1929, 2013.

[7] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE TPAMI*, vol. 33, no. 5, pp. 898–916, 2011.

[8] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *IJCV*, vol. 59, no. 2, pp. 167–181, 2004.

[9] M.-M. Cheng, Y. Liu, Q. Hou, J. Bian, P. Torr, S.-M. Hu, and Z. Tu, "HFS: Hierarchical feature selection for efficient image segmentation," in *ECCV*. Springer, 2016, pp. 867–882.

[10] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE TPAMI*, vol. 34, no. 11, pp. 2274–2282, 2012.

[11] C. Y. Ren, V. A. Prisacariu, and I. D. Reid, "gSLICr: SLIC superpixels at over 250hz," *arXiv preprint arXiv:1509.04232*, 2015.

[12] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, and A. Yuille, "The role of context for object detection and semantic segmentation in the wild," in *IEEE CVPR*, 2014, pp. 891–898.

[13] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results," 2007.

[14] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE TPAMI*, vol. 22, no. 8, pp. 888–905, 2000.

[15] D. Comaniciu and P. Meer, "Mean Shift: A robust approach toward feature space analysis," *IEEE TPAMI*, vol. 24, no. 5, pp. 603–619, 2002.

[16] Z. Ren and G. Shakhnarovich, "Image segmentation by cascaded region agglomeration," in *IEEE CVPR*, 2013, pp. 2011–2018.

[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[18] Y. Liu, M.-M. Cheng, X. Hu, K. Wang, and X. Bai, "Richer convolutional features for edge detection," in *IEEE CVPR*. IEEE, 2017, pp. 3000–3009.

[19] Y. Liu, M.-M. Cheng, J. Bian, L. Zhang, P.-T. Jiang, and Y. Cao, "Semantic edge detection with diverse deep supervision," *arXiv preprint arXiv:1804.02864*, 2018.

[20] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," in *ICLR*, 2015.

[21] W. Liu, A. Rabinovich, and A. C. Berg, "ParseNet: Looking wider to see better," in *ICLR*, 2016.

[22] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik, "Semantic contours from inverse detectors," in *IEEE ICCV*. IEEE, 2011, pp. 991–998.

[23] J. Pont-Tuset and F. Marques, "Supervised evaluation of image segmentation and object proposal techniques," *IEEE TPAMI*, vol. 38, no. 7, pp. 1465–1478, 2016.

[24] T. Cour, F. Benezit, and J. Shi, "Spectral segmentation with multiscale graph decomposition," in *IEEE CVPR*, vol. 2. IEEE, 2005, pp. 1124–1131.

[25] R. Girshick, "Fast R-CNN," in *IEEE ICCV*, 2015, pp. 1440–1448.

[26] A. Arnab and P. H. Torr, "Pixelwise instance segmentation with a dynamically instantiated network," in *IEEE CVPR*, 2017, pp. 441–450.

[27] X. Chen, H. Ma, X. Wang, and Z. Zhao, "Improving object proposals with multi-thresholding straddling expansion," in *IEEE CVPR*, 2015, pp. 2587–2595.

[28] M.-M. Cheng, Z. Zhang, W.-Y. Lin, and P. Torr, "BING: Binarized normed gradients for objectness estimation at 300fps," in *IEEE CVPR*, 2014, pp. 3286–3293.