

项目编号: 202209045

東 南 大 學

**大学生创新创业训练计划项目**

**结题报告**

项目名称 基于深度学习的超像素分割方法

项目级别 校级重大 校级重点

校级一般 院级

项目负责人 蔡英豪 学号: 58120127

所属学院 人工智能学院

项目成员 牟卓翊 (58120322)

王玟雯 (58120309)

指导教师 贾育衡

# 基于深度学习的超像素分割方法

蔡英豪, 卞卓翊, 王玟雯

东南大学 人工智能系, 南京 210096

**【摘要】** 超像素 (super pixel) 是一组邻近的且像素信息相似的像素。少量超像素就可以替代图片的原始像素完整表示图片信息。超像素的使用使得下游任务计算负担很大程度上减轻, 同时超像素的存在还使得下游任务能够忽略冗余的像素信息, 进而达到更好的效果。值得注意的是, 超像素分割任务不像语义分割 (semantic segmentation) 和实例分割 (instance segmentation) 任务一样需要对分割出的像素标注语义信息。所有分割出的超像素都是不含语义信息的。近年来, 计算机视觉应用越来越依赖超像素<sup>[1]</sup>。基于传统算法的超像素分割算法性能有限, 利用深度神经网络, 比如全卷积神经网络 (FCN) 实现的有监督超像素分割存在鲁棒性不高<sup>[2]</sup>, 数据需要人工标注等缺点。同时, 一些利用深度神经网络进行聚类的工作, 比如 Unsupervised Deep Embedding for Clustering<sup>[3]</sup> 给我们带来颇多启发。超像素本质上可以理解为一种强空间约束性的像素级别聚类, 我们可以使用深度聚类的方法进行超像素分割。

我们的项目旨在提出基于深度聚类的无监督超像素分割算法。我们设想的算法相较于其他用深度网络实现超像素分割的有监督算法, 比如 SSN (Superpixel Sampling Network)<sup>[4]</sup> 等更为精巧、轻量化。更重要的是, 我们的算法不需要人工标注标签, 在大数据时代更具有现实适用性。此外, 我们的算法还可以与后续任务结合, 进行协同训练。具体而言, 就是可以利用后续任务中得到的信息精细化调节深度网络参数, 进而得到更好的聚类效果。

基于上面的宗旨, 我们前后提出了两个分割算法——基于自编码器机制的超像素分割算法、基于自表示机制的超像素分割算法。基于自编码器机制的超像素分割算法是以深度聚类算法 DEC (Deep Embedded Clustering)<sup>[3]</sup> 为基础, 结合超像素分割要求的空间约束性提出的。它在各项指标上表现良好, 但也有着一个缺点——运行速度慢。经过分析, 原因是自编码器网络泛化性能不够好。之后, 在经过大量论文调研, 我们以 SENet (Self-Expressive Network)<sup>[5]</sup> 为基础提出了基于自表示机制的超像素分割算法。这一算法不仅在各项指标上表现良好, 更重要的是由于自表示机制的应用, 网络的泛化性能好, 对不同图片不需要重新训练网络。因此算法的时间性能相较于基于自编码器的超像素分割算法获得了很大提升。

**【关键词】** 超像素, 深度学习, 聚类, 自编码器, 自注意力

## 目录

<b>1 引言</b>	<b>3</b>
<b>2 文献综述</b>	<b>3</b>
2.1 超像素分割 . . . . .	3
2.1.1 超像素 . . . . .	3
2.1.2 评价指标 . . . . .	3
2.1.3 分割算法 . . . . .	5
2.2 深度聚类自编码器 . . . . .	7
2.2.1 深度聚类部分 . . . . .	7
2.2.2 无监督聚类部分 . . . . .	7
2.3 自表达模型在子空间聚类中的应用 . . . . .	8
2.3.1 Self-Expressive Model 自表达模型 . . . . .	8
2.3.2 SENet . . . . .	8
2.4 特征提取 . . . . .	9
2.4.1 纹理特征 . . . . .	9
<b>3 研究报告</b>	<b>9</b>
3.1 项目进程概览 . . . . .	9
3.2 项目研究过程 . . . . .	9
3.2.1 基础环境配置 . . . . .	9
3.2.2 基础知识储备 . . . . .	11
3.2.3 Pytorch 学习 & 实践 . . . . .	12
3.2.4 阅读超像素算法论文 & 分析源码 . . . . .	13
3.3 基于自编码器的超像素分割算法 . . . . .	14
3.3.1 原理 . . . . .	14
3.3.2 算法描述 . . . . .	16
3.3.3 实验 . . . . .	18
3.4 基于自表示机制的超像素分割算法 . . . . .	19
3.4.1 原理 . . . . .	19
3.4.2 算法描述 . . . . .	20
3.4.3 实验 . . . . .	21
<b>4 研究设计方案</b>	<b>22</b>
4.1 基于自表示机制的超像素分割算法 . . . . .	23
<b>5 项目成果简介</b>	<b>24</b>
<b>6 个人感想</b>	<b>24</b>
6.1 蔡英豪 . . . . .	24
6.2 牟卓翊 . . . . .	25
6.3 王玟雯 . . . . .	25
<b>7 展板</b>	<b>26</b>

# 1 引言

超像素 (super pixel) 算法将像素分组为具有感知意义的原子区域，这些原子区域可用于替换像素网格的刚性结构。它们捕获图像冗余，提供一个方便的基元来计算图像特征，并大大降低后续图像处理任务的复杂性。以上的优点使得超像素已成为许多计算机视觉算法的关键构建块。

超像素应用于许多现有的算法和模型中：PASCAL VOC Challenge, object localization, depth estimation, segmentation, body model estimation 等。

近年来，计算机视觉应用越来越依赖超像素。基于传统算法的超像素分割算法性能有限，利用深度神经网络，比如全卷积神经网络 (FCN) 实现的有监督超像素分割存在鲁棒性不高，数据需要人工标注等缺点。同时，一些利用深度神经网络进行聚类的工作，比如 Unsupervised Deep Embedding for Clustering 给我们带来颇多启发。超像素本质上可以理解为一种强空间约束性的像素级别聚类，我们可以使用深度聚类的方法进行超像素分割。

我们前后提出了两个分割算法——基于自编码器机制的超像素分割算法、基于自表示机制的超像素分割算法。基于自编码器机制的超像素分割算它在各项指标上表现良好，但也有着一个缺点——运行速度慢。之后我们提出了基于自表示机制的超像素分割算法。由于自表示机制的应用，网络的泛化性能好，对不同图片不需要重新训练网络。算法的时间性能获得了很大提升。

# 2 文献综述

## 2.1 超像素分割

### 2.1.1 超像素

超像素是指具有相似纹理、颜色、亮度等特征的相邻像素构成的有一定视觉意义的不规则像素块。具体见图1 它利用像素之间特征的相似性将像素分组，用少量的超像素代替大量的像素来表达图片特征，很大程度上降低了图像后处理的复杂度，所以通常作为分割算法的预处理步骤。<sup>[6]</sup>

### 2.1.2 评价指标

一般来说，大多数超像素相关论文的作者都赞同以下的定性指标<sup>[7]</sup>：

- 部分性 (Partition): 每一个像素都应该属于某一个超像素，图像由超像素组成。
- 连通性 (Connectivity): 超像素中的像素应该是连通的。
- 边界附着 (Boundary Adherence): 超像素的边界应该能保持图像本身的边界。
- 紧凑、规则和平滑 (Compactness, Regularity and Smoothness): 在没有图像边界的情况下，超像素应该是致密的、规则的，并且显示出平滑的边界。

但在实际评价时，我们往往需要更加量化的评价指标，上述仅作为一种定性的描述而存在。

超像素是图像分割的一种变种，超像素分割属于图像分割 (image segmentation) 之中的过分割 (over segmentation)<sup>[8]</sup>。

因此很多超像素的评价指标都是基于于图像分割中的评价标准。

**Boundary Recall (Rec)** 边界召回率 是给定 Ground Truth 情况下最常用的评价指标。反应了超像素的边界附着情况，Martin, D.R and Fowlkes, C.C.and Malik, J.<sup>[9]</sup> 提出使用 precision-recall curves 来度量图像分割

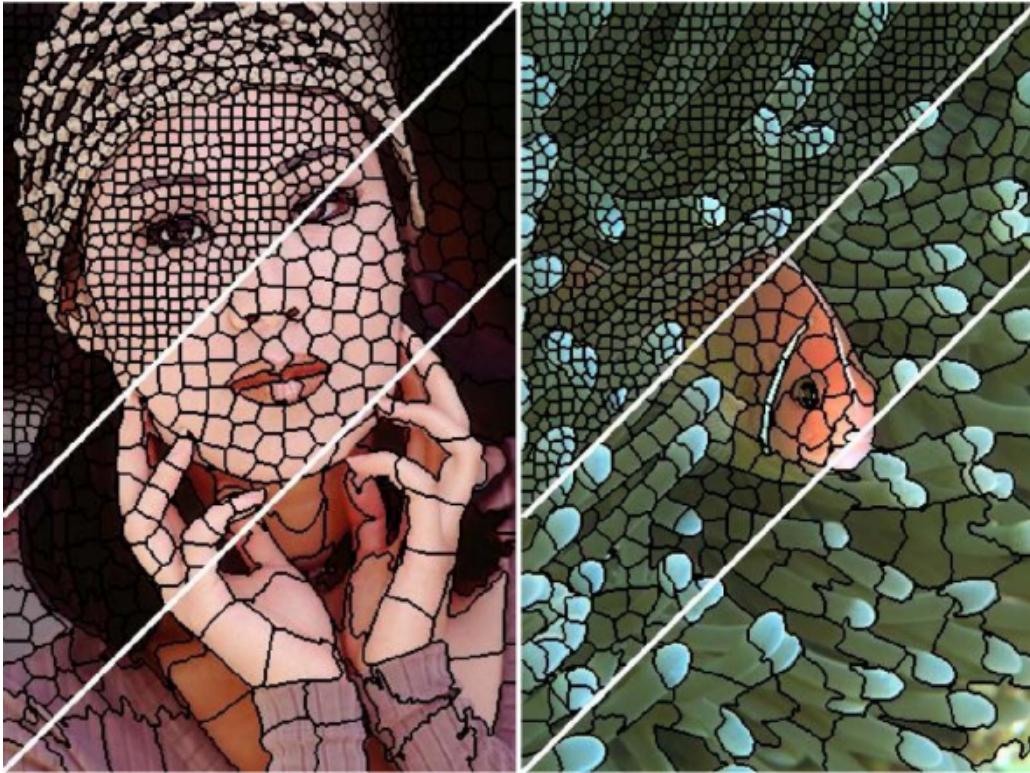


图 1 使用 SLIC 将图像分割成大小为 64、256 和 1, 024 的超像素

中边界检测的性能。作者将边界检测定义为一个区分非边界和边界像素的分类问题，将边界像素视为正例，非边界像素视为负例，并运用 precision-recall curves 的计算方法来展现边界检测的优劣。

$$\text{Rec}(G, S) = \frac{TP(G, S)}{TP(G, S) + FN(G, S)} \quad (1)$$

**Undersegmentation Error (UE) 欠分割误差** <sup>[10]</sup>反映了超像素的边界附着情况，当超像素不能很好的拟合 ground truth 的时候，就会有多出来的部分并不是 ground truth，称之为过度分割的部分。直观上衡量了由与给定的真实片段重叠的超像素引起的“出血”总量，并由片段的面积进行归一化。

$$\text{UE}_{\text{NP}}(G, S) = \frac{1}{N} \sum_{G_i} \sum_{S_j \cap G_i \neq \emptyset} \min \{ |S_j \cap G_i|, |S_j - G_i| \} \quad (2)$$

**Achievable Segmentation Accuracy (ASA)** 反映了超像素的性能上限。它为使用超像素作为单位的对象分割提供了可实现的最高准确度。为了计算 ASA，Liu, Ming-Yu and Tuze<sup>[11]</sup> 使用具有最大重叠的地面实况段的标签来标记每个超像素。

$$\text{ASA}(G, S) = \frac{1}{N} \sum_{S_j} \max_{G_i} \{ |S_j \cap G_i| \} \quad (3)$$

**Explained Variation (EV)** 反映了超像素的几何紧凑度，Moore, Alastair P. and Prince, Simon J. D. and Warrell, Jonathan and Mohammed, Umar and Jones, Graham<sup>[12]</sup> 计算超像素分割之后图像的方差和原始的方差的比值，比值越接近于 1 越好。

$$EV(S) = \frac{\sum s_j |S_j| (\mu(S_j) - \mu(I))^2}{\sum_{x_n} (I(x_n) - \mu(I))^2} \quad (4)$$

**Compactness (CO)** 反映了超像素的几何紧凑度。基于等周商，Schick, Alexander and Fischer, Mika and Stiefelhagen, Rainer<sup>[13]</sup>提出了一种度量来衡量超像素分割的紧凑性 (CO)。对于给定的分割，作者计算每个超像素的等周商的总和，由超像素大小的分数  $|S|$  归一化。

$$CO(G, S) = \frac{1}{N} \sum_{S_j} |S_j| \frac{4\pi A(S_j)}{P(S_j)} \quad (5)$$

### 2.1.3 分割算法

David Stutz and Alexander Hermans and Bastian Leibe 等人在综述 Superpixels: An evaluation of the state-of-the-art<sup>[8]</sup> 中系统全面地对有代表性的超像素算法进行了评测。并将算法大致分为以下几类：

- 基于分水岭算法 (Watershed-based)
- 基于密度算法 (Density-based)
- 基于图论算法 (Graph-based)
- 轮廓进化算法 (Contour evolution)
- 基于路径算法 (Path-based)
- 基于聚类算法 (Clustering-based): SLIC<sup>[14]</sup>, PreSLIC<sup>[15]</sup>, LSC<sup>[16]</sup>
- 能量优化算法 (Energy optimization): SEEDS<sup>[17]</sup>, ETPS<sup>[18]</sup>
- 基于小波分析 (Wavelet-based)

作者发现基于路径和密度的评估算法以及过分割算法显示出较低的视觉质量。另一方面，基于聚类、轮廓演化和迭代能量优化算法大多表现出良好的边界附着，有些算法提供了一个紧凑性参数，如 SLIC<sup>[14]</sup> 和 ETPS<sup>[18]</sup>。然而，良好的边界附着，特别是关于图像中的细节，通常以较低的几何紧凑度为代价，如 ETPS<sup>[18]</sup> 和 SEEDS<sup>[17]</sup>。

**SLIC** 基于 k-means 的聚类过程进行了两点改动：

- 距离度量  
 $d_c$  为色彩空间的欧氏距离,  $d_s$  为像素平面空间的欧氏距离。

$$\begin{aligned} d_c &= \sqrt{(l_j - l_i)^2 + (a_j + a_i)^2 + (b_j + b_i)^2} \\ d_s &= \sqrt{(x_j + x_i)^2 + (y_j + y_i)^2} \\ D' &= \sqrt{\left(\frac{d_c}{N_c}\right)^2 + \left(\frac{d_s}{N_s}\right)^2} \end{aligned} \quad (6)$$

- 聚类搜索空间  
由原来的全局空间变成  $2S$  大小的局部空间。

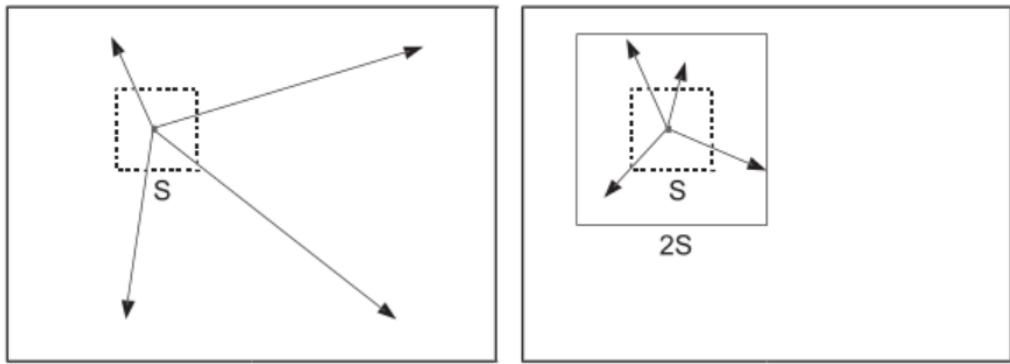


图 2 k-means 搜索空间和 SLIC 搜索空间对比

初始化时在图像空间上进行均匀的撒下聚类中心的种子，然后按 k-mean 算法步骤运行，最终将像素的颜色修改为对应聚类中心的颜色。

Achanta, Radhakrishna and Shaji 在 SLIC: Superpixels Compared to State-of-the-Art Superpixel Methods<sup>[14]</sup> 中进行了更多改进：

- 采用了更多种色彩空间的表达方式来进行距离度量 (CIELAB)，并与传统 RGB 相比取得了更好的效果。
- 在 k-means 距离度量公式中对色彩空间以及欧式空间进行计算时加权求和，引入一个权重的度量因子 m。

使得超像素分割结果对几何空间相似性和色彩相似性的倾向可以控制。m 越大，空间相似性越重要，超像素的几何紧凑性更强。m 越小，超像素的色彩相似性更强，超像素的结果就会顺着图像的纹理蔓延。

#### SEEDS 基于能量优化对边界的调整生成超像素：

SEEDS 初始化直接将图片划分等大小的方格，然后通过后续的步骤调整方格的边界来修正方格里包含的颜色分布。

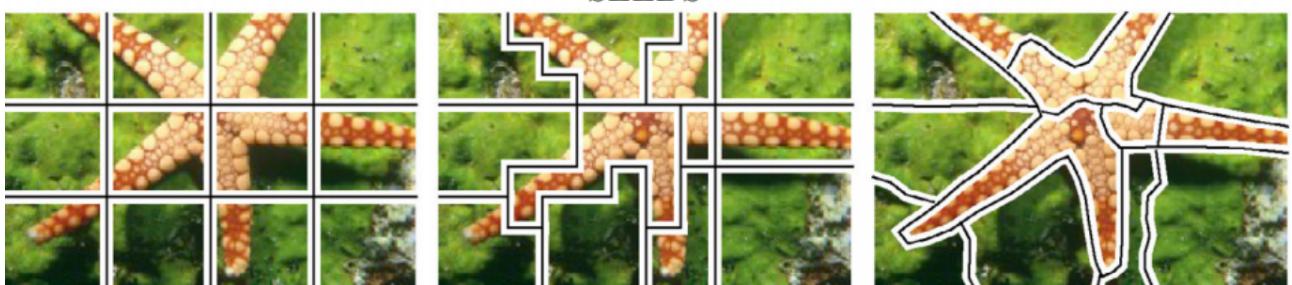


图 3 SEEDS 算法原理过程

Van den Bergh, Michael and Boix, Xavier<sup>[17]</sup> 通过定义一个能量函数，使用爬山法对其进行优化。

$$E(s) = H(s) + \gamma G(s)$$

- 颜色能量分布项  $H(s)$

超像素个体应在视觉上一致，因而颜色应尽可能均匀。作者采用图像处理中经典的概率密度分布直方图。统计颜色的概率密度分布，然后用熵评估直方图中颜色种类分布的多少。当颜色只有一种的时候，熵达到最大为 1，这样处理以用爬山法进行优化。

- 边界能量分布项  $G(s)$

类似于颜色能量分布项，SEEDS 以每一个像素点为中心，构造其周围  $N \times N$  的正方形区域信息。统计其中包含的超像素种数的概率密度分布，同样的使用熵来评估。同理当熵比较高的时候，这个像素周围的超像素的种数单一，说明边界比较规整。

我们也对较为前沿的改进分割算法进行了研究。

## 2.2 深度聚类 自编码器

### 2.2.1 深度聚类部分

深度学习神经网络或人工神经网络试图通过数据输入、权重和偏差的组合来模仿人脑。这些元素协同工作以准确识别、分类和描述数据中的对象。它由多层互连节点组成，每一层都建立在前一层的基础上，以改进和优化预测或分类。这种通过网络进行的计算过程称为前向传播。

深度聚类能够实现特征学习和聚类的联合优化，逐渐成为处理大规模复杂数据聚类问题的首选方案。

- Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.<sup>[19]</sup>

这篇文章探索了一种构建深度网络的原始策略，基于堆叠的去噪自动编码器层，这些自动编码器在本地训练以去噪其输入的损坏版本。生成的算法是普通自动编码器堆叠的直接变体。然而，它在分类问题的基准上显示可以显著降低分类错误，从而弥合深度信念网络(DBN)的性能差距，并在某些情况下超越它。以这种纯粹无监督的方式学习的更高级别的表示也有助于提高后续 SVM 分类器的性能。定性实验表明，去噪自编码器能够从数字图像中学习更大的笔画检测器。这项工作清楚地确立了使用去噪标准作为易于处理的无监督目标来指导学习有用的更高级别表示的价值。

- Generalized Autoencoder: A Neural Network Framework for Dimensionality Reduction<sup>[20]</sup>

自编码算法及其深度版本作为传统的降维方法，通过神经网络强大的可表示性取得了巨大的成功。然而，他们只是使用每个实例来重构自己，而忽略显式地建模数据关系，从而发现潜在的有效流形结构。文章通过所谓的“广义自动编码器”(GAE)，提出了一种通过流形学习的降维方法，它迭代地探索数据关系并利用关系来追求流形结构。GAE 通过最小化重建实例与原始实例之间的加权距离来捕获数据空间的结构。广义自编码器为降维提供了一个通用的神经网络框架。此外，文章提出了一种广义自编码器的多层架构，称为深度广义自编码器，以处理高度复杂的数据集。实验表明，文章所提出的方法实现了更好的性能。

- Masked Autoencoders Are Scalable Vision Learners<sup>[21]</sup>

本文通过比较原始自编码器，提出了 MAE 架构。MAE 是一种简单的自编码器方案，采用了非对称的编解码器架构，这使得编码器仅依赖于部分观测信息(无需掩码 token 信息)，而轻量解码器则接与所得隐式表达与掩码 token 进行原始信号重建。MAE 基于给定部分观测信息对原始信号进行重建，它包含一个将观测信号映射为隐式表达的编码器，一个用于将隐式表达重建为原始信号的解码器。通过实验证明，MAE 方法可以从局部推断出复杂的整体重建，表明它已经学习到了许多视觉概念，即语义。

### 2.2.2 无监督聚类部分

无监督聚类学习是机器学习中的一个重要概念，无监督学习使用学习算法来发现未标记数据集中的未知模式，发掘数据中的相似性并且按照模式分组。自编码器 (autoencoder, AE) 是一类在半监督学习和无监督学习中使用的人工神经网络 (Artificial Neural Networks, ANNs)，其功能是通过将输入信息作为学习目标，对输入信息进行表征学习 (representation learning)。它作为一种基于神经网络的特征提取方法，在生成高维数据的抽象特征方面取得了巨大成功。

- Unsupervised Deep Embedding for Clustering Analysis<sup>[22]</sup>

聚类是许多数据驱动应用领域的核心，并且已在距离函数和分组算法方面进行了广泛的研究。但是相对较少的工作集中在学习聚类表示。这篇文章提出了深度嵌入式聚类 (DEC)，一种使用深度神经网络同时学习特征表示和聚类分配的方法。DEC 学习从数据空间到低维特征空间的映射，在其中迭代优化聚类目标。通过迭代优化具有自我训练目标分布的基于 KL 散度的聚类目标来工作。这个方法可以看作是半监督自我训练的无监督扩展。DEC 的框架提供了一种方法来学习专门用于聚类的表示，而无需真实的聚类成员标签。DEC 提供了改进的性能以及对超参数设置的鲁棒性，这在无监督任务中尤为重要，因为交叉验证是不可能的。DEC 还具有数据点数量的线性复杂性，这使其可以扩展到大型数据集。

- Visualizing data using t-SNE<sup>[23]</sup>

本篇文章提出了一种称为“t-SNE”的新技术，它通过在二维或三维地图中为每个数据点提供一个位置来可视化高维数据。该技术是随机邻域嵌入的一种变体，它更容易优化，并且通过减少将点聚集在地图中心的趋势来产生明显更好的可视化效果。t-SNE 在创建单个地图以显示许多不同比例的结构方面优于现有技术。这对于位于几个不同但相关的低维流形上的高维数据尤其重要，例如从多个视点看到的来自多个类别的对象的图像。为了可视化非常大的数据集的结构，文章展示了 t-SNE 如何在邻域图上使用随机游走，以允许所有数据的隐式结构影响数据子集的显示方式。本文说明了 t-SNE 在各种数据集上的性能，并将其与许多其他非参数可视化技术进行了比较，包括 Sammon 映射、Isomap 和局部线性嵌入。在几乎所有数据集上，t-SNE 产生的可视化效果明显优于其他技术产生的可视化效果。

## 2.3 自表达模型在子空间聚类中的应用

### 2.3.1 Self-Expressive Model 自表达模型

<sup>[5]</sup> 给定数据矩阵  $X = [x_1, \dots, x_N] \in \mathbb{R}^{D \times N}$ ，其中每一列是一个数据点  $x_j \in \mathbb{R}^D$ 。我们可以利用数据进行自表达：

$$x_j = \sum_{i \neq j} c_{ij} x_i$$

其中  $C_{ij}$  是自表示系数矩阵，我们希望这一个系数矩阵满足子空间保持 (subspace-preserving) 性质。

### 2.3.2 SENet

Shangzhi Zhang 等人在 Learning a Self-Expressive Network for Subspace Clustering<sup>[5]</sup> 中对该模型进行了改进，文章主要引入了自表示网络 (SENet) 来求解自表示矩阵元。

**模型架构** 他们选择一个更加多样性的正则化：将一范数与二范数混合：

$$r(\cdot) = \lambda \|\cdot\|_1 + \frac{1-\lambda}{2} \|\cdot\|_2$$

这样可以得到更稠密的自表示矩阵。

为了将自表示模型与深度学习进行结合从而可以利用神经网络直接推断出矩阵元  $C_{ij}$ ，他们将矩阵元  $C_{ij}$  直接使用神经网络参数化：

$$f(x_i, x_j, \Theta) = \alpha \mathcal{T}_b(u_j^T v_i)$$

并基于自表示模型的损失函数给出一个增量更新法 (Two-Pass-Algorithm)：

$$\frac{\partial l}{\partial \Theta} = \sum_{i \neq j} (r'(f(x_i, x_j; \Theta)) - \langle x_i, q_j \rangle) \frac{\partial l(x_i, x_j; \Theta)}{\partial \Theta}$$

$$q_j \equiv \gamma \left( x_j - \sum_{i \neq j} f(x_i, x_j; \Theta) x_i \right)$$

实验的结果表示 SENet 拓展了原有的自表示模型的应用范围，使之能高效训练与处理较大规模数据和新样本的情形。

## 2.4 特征提取

### 2.4.1 纹理特征

纹理特征是一种全局特征，反映的是图像中同质现象的视觉特征，体现物体表面的具有缓慢变换或周期性变化的表面组织结构排列属性。在我们的模型中，使用 XCS-LBP 算法提取像素的纹理特征。

- An eXtended Center-Symmetric Local Binary Pattern for Background Modeling and Subtraction in Videos<sup>[2]</sup>
- 背景减法 (BS) 是许多计算机视觉应用中的主要步骤之一，例如对象跟踪、行为理解和活动识别。BS 过程基本上包括：a) 背景模型初始化，b) 背景模型维护和 c) 前景检测。在过去几年中已经开发了许多 BS 方法。在本文中，我们扩展 LBP 的变体。通过引入一种新的相邻像素比较策略，该策略允许描述符对噪声像素不太敏感并产生短直方图，同时保持对光照变化的鲁棒性。

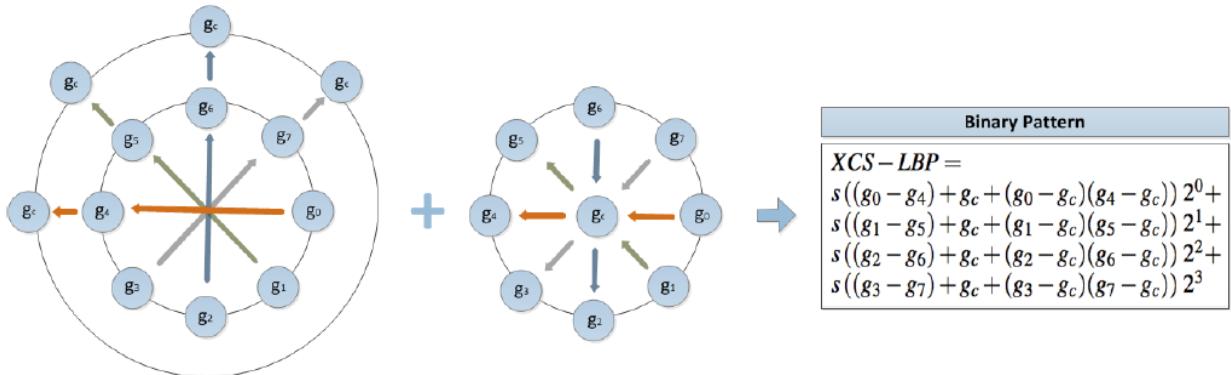


图 4 XCS-LBP 算法描述

## 3 研究报告

### 3.1 项目进程概览

- 2021.10 - 2021.11 基础知识储备、基础环境配置
- 2021.11 - 2021.12 学习 pytorch 使用并进行项目实践
- 2022.12 - 2022.1 阅读超像素相关论文，分析源码
- 2022.1 - 2022.2 阅读深度聚类相关论文，分析源码
- 2022.2 - 2022.3 设计将深度聚类用于超像素分割的框架并用代码实现
- 2022.3 - 2022.4 测试算法性能

### 3.2 项目研究过程

#### 3.2.1 基础环境配置

**Python 环境** 在和导师交流后，我们决定采用轻量级开发环境，使用 Miniconda+VsCode 的基础配置作为我们的代码编写环境。

**Linux 服务器环境** 由于使用 CPU 训练深度神经网络耗时较长，我们使用 VsCode 连接并登录了学校 k-80 服务器。在简单熟悉 Linux 指令后，我们在服务器上配置了虚拟环境并进行了显卡测试。

```
(base) caiyinghao@max:/data/cyh/superpixel/sp_dec$ nvidia-smi
Fri Apr 8 15:42:01 2022
+-----+
| NVIDIA-SMI 410.73     Driver Version: 410.73      CUDA Version: 10.0 |
| GPU Name Persistence-M Bus-Id Disp.A Volatile Uncorr. ECC |
| Fan Temp Perf Pwr/Usage/Cap| Memory-Usage | GPU-Util Compute M. |
+-----+
| 0 Tesla K80      On   57W / 149W | 337MiB / 1144MiB | 0% Default |
| 1 Tesla K80      On   106W / 149W | 2338MiB / 1144MiB | 22% Default |
| 2 Tesla K80      On   82W / 149W | 2342MiB / 1144MiB | 34% Default |
| 3 Tesla K80      On   72W / 149W | 2344MiB / 1144MiB | 0% Default |
| 4 Tesla K80      On   75W / 149W | 2338MiB / 1144MiB | 0% Default |
| 5 Tesla K80      On   153W / 149W | 2434MiB / 1144MiB | 98% Default |
| 6 Tesla K80      On   132W / 149W | 2342MiB / 1144MiB | 98% Default |
| 7 Tesla K80      On   73W / 149W | 2702MiB / 1144MiB | 0% Default |
| 8 Tesla K80      On   147W / 149W | 3921MiB / 1144MiB | 84% Default |
| 9 Tesla K80      On   74W / 149W | 2888MiB / 1144MiB | 0% Default |
+-----+
```

图 5 k-80 服务器

**Git 仓库** 为方便小组间文件管理以及代码版本管理，同时也是为了最后将我们的成果开源，我们创建了 Git 仓库进行分布式部署。

提交者	提交信息	时间
Fuyao233	add README.md. 3a22244 刚刚	95 次提交
Superpixels_Li @ f65b382	1	23小时前
	//again try	3个月前
DEC	Merge branch 'master' of https://gitee.com/Fuyao233/superpixel	23小时前
SLIC	1	23小时前
data/MNIST	sp_dec	24小时前
runs/Apr03_21-29-39_amax	sp_dec	24小时前
sp_dec	sp_dec	24小时前
srtp日记.assets	1	23小时前
srtp日记相关	DEC	24天前
文档	update	1个月前
测试图片	push	3个月前
论文	1	23小时前
.gitignore	论文更新	1个月前
README.md	add README.md.	刚刚

图 6 项目 Git 仓库

### 3.2.2 基础知识储备

我们的项目目的是用深度聚类的方法实现无监督的超像素分割。因此，我们从两个方面进行知识储备：超像素分割算法和深度学习。

**超像素分割算法** 在超像素分割算法方面，我们阅读了该领域的综述<sup>[7]</sup>。该篇论文的作者系统全面地对有代表性的超像素算法进行了评测，为我们建立起超像素分割领域的初步认知。从中我们了解到，超像素分割领域的主要目的是根据相似性将像素分组，用少量的大块的超像素代替大量的个体的像素来表达图片特征，在极大降低图像后续处理的复杂度的同时尽可能保留图像信息。该领域的传统算法根据不同原理大致可以分为以下几类：

- 基于分水岭算法 (Watershed-based)
- 基于密度算法 (Density-based)
- 基于图论算法 (Graph-based)
- 轮廓进化算法 (Contour evolution)
- 基于路径算法 (Path-based)
- 基于聚类算法 (Clustering-based): SLIC<sup>[14]</sup>, PreSLIC<sup>[15]</sup>, LSC<sup>[16]</sup>
- 能量优化算法 (Energy optimization): SEEDS<sup>[17]</sup>, ETPS<sup>[18]</sup>

根据超像素分割的目的，该领域确定了一些定性指标，比如：部分性 (Partition)、连通性 (Connectivity)、边界附着 (Boundary Adherence) 等。根据这些定性指标衍生出许多量化指标：

- **Boundary Recall (Rec)** 边界召回率是给定 Ground Truth 情况下最常用的评价指标。反应了超像素的边界附着情况，Martin, D.R. and Fowlkes, C.C. and Malik, J.<sup>[9]</sup>提出使用 precision-recall curves 来度量图像分割中边界检测的性能。作者将边界检测定义为一个区分非边界和边界像素的分类问题，将边界像素视为正例，非边界像素视为负例，并运用 precision-recall curves 的计算方法来展现边界检测的优劣。
- **Undersegmentation Error (UE)** 欠分割误差<sup>[10]</sup>反映了超像素的边界附着情况，当超像素不能很好的拟合 ground truth 的时候，就会有多出来的部分并不是 ground truth，称之为过度分割的部分。直观上衡量了由与给定的真实片段重叠的超像素引起的“出血”总量，并由片段的面积进行归一化。
- **Achievable Segmentation Accuracy (ASA)** 反映了超像素的性能上限。它为使用超像素作为单位的对象分割提供了可实现的最高准确度。为了计算 ASA，Liu, Ming-Yu and Tuze<sup>[11]</sup>使用具有最大重叠的地面实况段的标签来标记每个超像素。
- **Explained Variation (EV)** 反映了超像素的几何紧凑度，Moore, Alastair P. and Prince, Simon J. D. and Warrell, Jonathan and Mohammed, Umar and Jones, Graham<sup>[12]</sup>计算超像素分割之后图像的方差和原始的方差的比值，比值越接近于 1 越好。
- **Compactness (CO)** 反映了超像素的几何紧凑度。基于等周商，Schick, Alexander and Fischer, Mika and Stiefelhagen, Rainer<sup>[13]</sup>提出了一种度量来衡量超像素分割的紧凑性 (CO)。对于给定的分割，作者计算每个超像素的等周商的总和，由超像素大小的分数  $|S|$  归一化。

此外作者还定量测量比较的各种算法的指标，下图展示了比较排名的结果：

AMR	AUE		Rank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
2.05	6.07	<b>ETPS</b>	1	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1.57	7.98	<b>SEEDS</b>	3.8	0	2	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3.38	6.28	<b>ERS</b>	3.8	0	1	1	2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4.16	6.29	<b>CRS</b>	4.8	0	0	2	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4.18	6.77	<b>EAMS</b>	5.4	0	1	1	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4.55	6.34	<b>ERGC</b>	5.8	0	0	0	0	2	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4.91	6.50	<b>SLIC</b>	6.2	0	0	0	0	2	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5.40	7.24	<b>LSC</b>	9.2	0	0	0	1	0	1	0	0	2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
5.49	7.02	<b>preSLIC</b>	9.2	0	0	0	0	0	0	0	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6.60	6.41	<b>CCS</b>	10.6	0	0	0	0	0	0	2	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6.49	7.19	<b>CW</b>	11	0	0	0	0	0	0	0	0	2	2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8.24	7.83	<b>DASP</b>	12	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
6.51	7.44	<b>W</b>	12.8	0	0	0	0	0	0	0	0	0	0	2	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
7.44	6.97	<b>WP</b>	13.4	0	0	0	0	0	0	0	0	0	0	0	0	1	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0
7.85	7.58	<b>POISE</b>	13.8	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0
8.13	6.90	<b>NC</b>	15.25	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
7.66	7.43	<b>MSS</b>	15.8	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0
8.90	7.67	<b>VC</b>	17.8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	0	0	1	0	0	0	1	0	0	0	0	0
8.31	7.85	<b>PB</b>	18.4	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	1	0	0	0	0	0	0
8.53	8.15	<b>FH</b>	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	3	0	1	0	0	0	0	0	0	0	0
8.35	7.32	<b>RW</b>	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	1	0	0	0	0	0	0
11.45	6.70	<b>CIS</b>	20.8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	0	0	0	1	0	0	0	0	0	0
11.45	7.19	<b>TPS</b>	21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	1	0	0	0	0	0
11.05	7.93	<b>TP</b>	21.6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	0	0	0	0	0	0
14.22	13.54	<b>VCCS</b>	23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0
11.97	8.36	<b>SEAW</b>	24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	0	1	0	0	0
15.72	10.75	<b>QS</b>	24.8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	2	0	1	0	0	0
24.64	14.91	<b>PF</b>	26.2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	3	0	1	0	0	0	0

图 7 超像素分割算法 AMR、AUE 指标排名

**深度学习** 因为我们做的任务属于计算机视觉 (CV) 领域，所以我们选择斯坦福大学公开课 CS231n 作为入门课程。在这部分学习中我们学习了 CV 领域的基础概念，比如 ground truth、边缘检测算子等。我们还学习了包括 SGD、Momentum、NAG、Adam 等各种优化器模型。同时深度学习领域的许多经典技巧也为之后训练深度神经网络打好基础。此外，课程中 ResNet、Googlet 等高性能卷积神经网络模型体现的设计思想也为项目后期发展提供了更多可能。

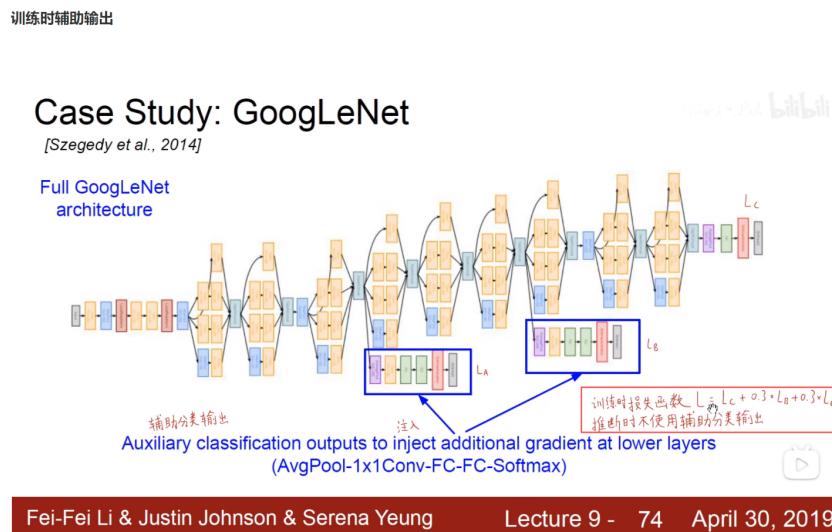


图 8 学习过程的部分笔记：Googlet 的模型细节

### 3.2.3 Pytorch 学习 & 实践

接着，通过阅读 Pytorch 的官方文档以及《Dive-into-DL-PyTorch》，我们逐步建立起对 Pytorch 框架的认知。最后我们动手实践，搭建 Lenet-5 并在 MNIST 手写数字数据集上进行测试。



图 9 Lenet-5 模型结构可视化

### 3.2.4 阅读超像素算法论文 & 分析源码

完成以上准备工作后，我们根据综述<sup>[7]</sup>寻找到这个领域的经典算法。我们选取了基于  $k-means$  的超像素分割经典算法：SLIC<sup>[14]</sup>。通过搜索引擎，我们找到了 SLIC 算法的 C++ 实现和 Python 实现。结合论文，我们分析了源码并画出了算法实现的流程图。

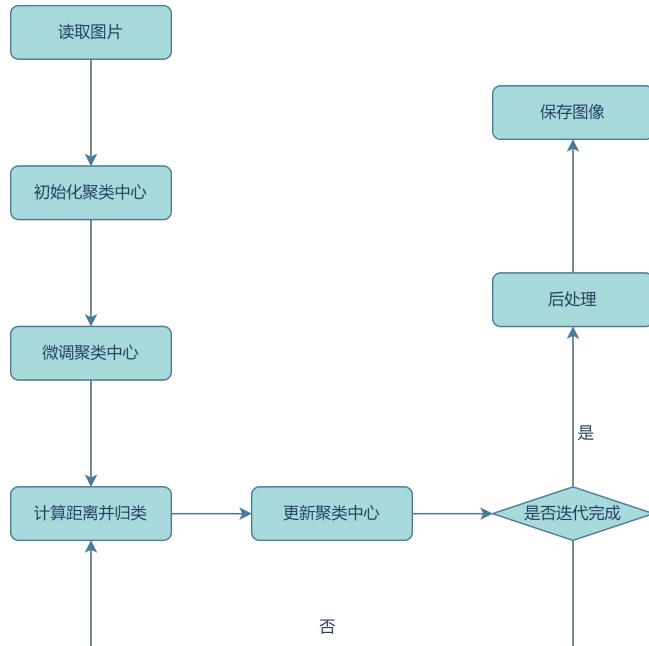


图 10 SLIC 算法实现流程图

- **初始化聚类中心** 为了方便后续任务处理和尽可能多的保留信息，SLIC 算法选择均匀初始化聚类中心，即将聚类中心均匀网格状分布在图像上
- **微调聚类中心** 在聚类中心临近像素搜索梯度较小的像素作为新的聚类中心。这样做的目的是避免初始得到的聚类中心位于边界上，影响最终的分割效果

- **计算距离并归类** 遍历图片像素，计算其与  $2S$  ( $S$  为一个超像素块的大小) 范围内聚类中心的距离。这里的距离采用欧氏距离： $d_c$  为色彩空间的欧氏距离， $d_s$  为像素平面空间的欧氏距离。其中  $N_s$ ,  $N_c$  是空间和像素颜色的相对重要性的度量。选择距离最小的聚类中心作为像素的标签。

$$d_c = \sqrt{(l_j - l_i)^2 + (a_j + a_i)^2 + (b_j + b_i)^2}$$

$$d_s = \sqrt{(x_j + x_i)^2 + (y_j + y_i)^2}$$

$$D' = \sqrt{\left(\frac{d_c}{N_c}\right)^2 + \left(\frac{d_s}{N_s}\right)^2}$$

**更新聚类中心** 选择同一个类的平均像素点作为新的聚类中心

- **迭代** 迭代次数一般选择 10-20 次
- **后处理** 图像聚类后需要进行强制连接性后处理，此步将孤立的小块的超像素归入临近的大的超像素块中
- **保存图像** 用每个超像素的平均像素作为填充并保存

通过分析 SLIC 的源码，我们得以更深入理解超像素的分割过程，了解超像素分割的流程。同时，SLIC 作为基于聚类的超像素分割算法也给我们用深度聚类实现超像素分割提供了许多思路。



图 11 SLIC 分割名画《星空》

### 3.3 基于自编码器的超像素分割算法

#### 3.3.1 原理

通过 SLIC 算法源码的分析，我们深深体会到分析源码的好处。只有读懂源码，才算是彻底理解了算法，才能从算法中理解作者设计的精妙之处。在深度聚类方面，在导师指导下，我们阅读了 Unsupervised Deep Embedding for Clustering<sup>[3]</sup>。这篇文章使用自编码器进行特征的嵌入，通过构造分布  $P$ ,  $Q$  利用 KL 散度对自编码器进行微调来实现深度聚类。我们找到相关在 MNIST 数据集上测试的代码进行分析，并根据代码绘制了以下流程图。

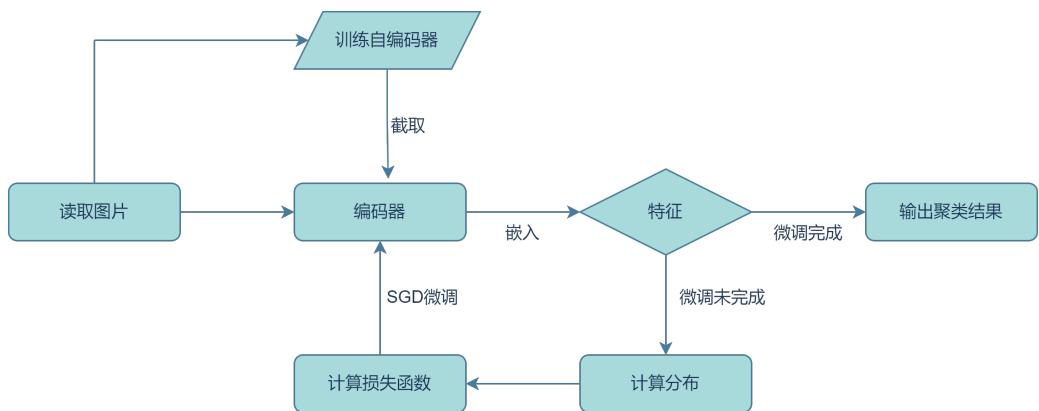


图 12 DEC 算法实现流程图

- **训练自编码器** 此处作者采用的自编码器为栈式降噪自编码器<sup>[19]</sup>。“降噪”体现在在输入中引入噪声，这样可以训练得到鲁棒性更好的嵌入映射。“栈式”体现在训练方式上，通过一层一层训练最终堆叠的方式保证了每一层都能提取到上一层的有效特征。

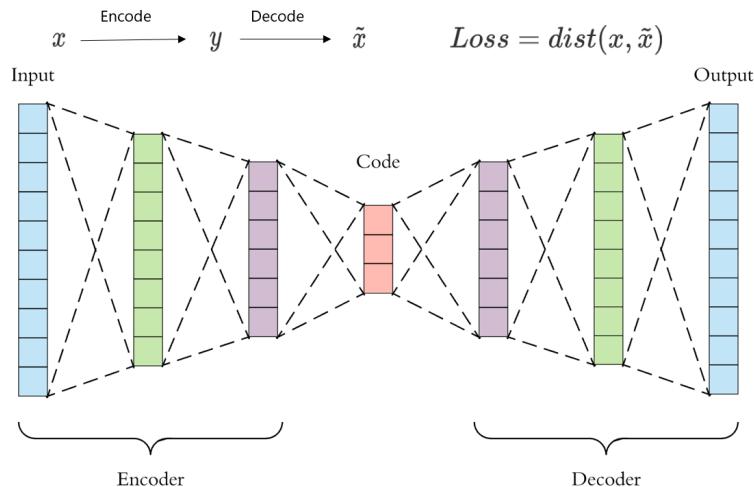


图 13 自编码器的一般结构

- **截取** 自编码器的一般结构见图13，分为编码器 (encoder)、嵌入特征 (code)、解码器 (decoder)。嵌入特征的维度低于输入 (input) 的维度，而输入和输出维度是相同的。数据输入自编码器后先经过编码器映射到低维空间再通过解码器映射到原始维度空间。自编码的损失函数通常是输入输出的均方误差。通过最小化输入输出的均方误差，自编码器可以保证编码器部分提取到输入的有效特征。在 DEC 框架中，自编码器的作用是得到数据从高维原始空间到低维嵌入空间的非线性映射。在低维空间的聚类更为有效。

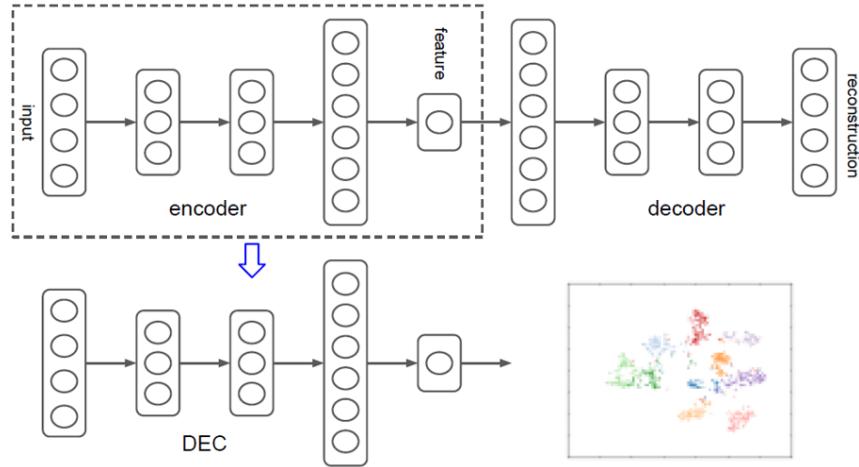


图 14 从自编码器中截取编码部分

- **计算分布** 作者从 t-SNE<sup>[23]</sup> 中得到启发，用自由度为 1 的 t 分布来衡量嵌入空间的点到聚类中心的距离并做了归一化，得到了 Q 分布：

$$q_{ij} = \frac{\left(1 + \|z_i - \mu_j\|^2 / \alpha\right)^{-\frac{\alpha+1}{2}}}{\sum_{j'} \left(1 + \|z_i - \mu_{j'}\|^2 / \alpha\right)^{-\frac{\alpha+1}{2}}}$$

其中  $z_i$  为嵌入空间的点， $\mu_j$  为聚类中心。之后，又为了提高高置信点的置信度，作者将  $q_{ij}$  平方再归一化得到 P 分布：

$$p_{ij} = \frac{q_{ij}^2 / f_j}{\sum_{j'} q_{ij'}^2 / f_{j'}}$$

- **计算损失函数** 这里作者采用  $KL(P||Q)$  作为损失函数，即

$$L = KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

$KL$  散度计算两个分布的熵差，可以用来衡量两个分布的相似度。 $KL$  散度越小，两个分布越接近。

- **微调** 随机梯度下降优化器 (SGD) 被用来进行优化迭代。这一步的目的是微调编码器 (encoder) 部分的参数，使其得到的编码分布更加集中，置信度更高。

总体来看，DEC 的创新点就在构造辅助分布 P 并使用  $KL$  散度训练提高点的置信度。在分析源码后，我们再 MNIST 数据集测试算法。最终准确率达到了 83.33%。传统聚类算法比如 SLIC 算法中使用的 k-means，在 MNIST 数据集上的精度只有 47.14%<sup>[3]</sup> 这相比传统的聚类算法是巨大的进步，我们决定以 DEC 算法为基础设计基于自编码器机制的超像素分割算法。

### 3.3.2 算法描述

相比于在 MNIST 上测试的 DEC 是将一张图片作为一个样本进行聚类，超像素分割任务需要进行像素级别的聚类。相比之下，进行像素级别的深度聚类有以下难点：

- **输入维度低** 单个像素维度低，信息少。对一张 RGB 图片，即使算上坐标特征也只能得到五维特征  $(x, y, R, G, B)$ 。而较少的特征就会导致图片信息不能充分利用，聚类效果不佳
- **空间约束性差** 在超像素分割领域，要求只有相邻的像素才能并入一个超像素中，因此一个像素只可能属于“九宫格”（网格为初始超像素块）。而 DEC 构造 P、Q 分布时会计算嵌入空间点到所有聚类中心的距离，最终可能会导致一个像素被归入一个空间距离很远但纹理、梯度等特征相似的超像素中

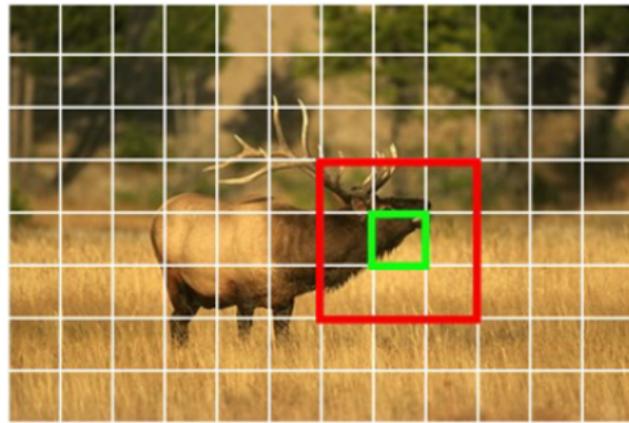


图 15 绿色区域像素所属聚类中心一定在红色区域内

为了解决以上问题，我们依托 DEC 的框架，结合超像素分割领域的处理流程与经典思想，做了许多创新与改进，提出了基于自编码器机制的超像素分割算法：

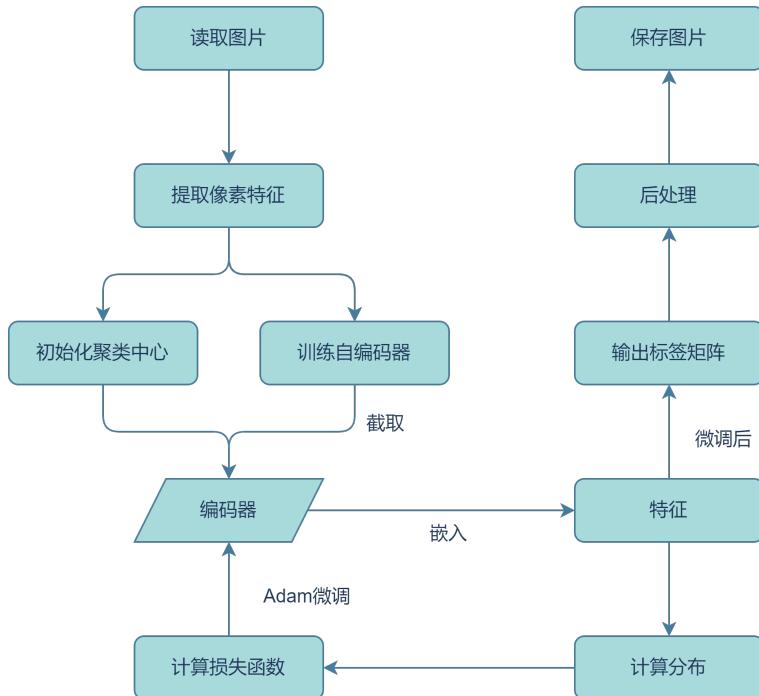


图 16 基于自编码器机制的超像素分割算法流程图

- **提取像素特征** 为了解决输入维度少的问题，我们决定进行特征提取。我们从一篇基于子空间聚类的超像素分割算法<sup>[24]</sup>中得到启发，从原始图像中提取特征，将原始的五维特征提升至 19 维。这十九维特征包括像素的坐标、灰度、梯度、梯度幅值、梯度方向、纹理特征<sup>[?]</sup>。如此高维特征在原始空间聚类效果很差，而我们的编码器就可以将高维特征嵌入到低维空间进行聚类。
- **初始化聚类中心** 此处我们先在原始图像上网格化初始聚类中心，然后将其提取特征并嵌入到低维空间作为嵌入空间的聚类中心。

- **局部的聚类** 为了解决空间约束性差的问题，我们决定对  $Q$  分布进行改进。首先定义  $k_{ij}$ :

$$k_{ij} = \begin{cases} \frac{1}{1+\|z_i - \mu_j\|^2}, & \|x_i - \mu_j\|^2 > 2S \\ 0, & \|x_i - \mu_j\|^2 \leq 2S \end{cases}$$

其中， $x_i$  为原始图像空间点， $\pi_j$  为原始图像空间聚类中心，且  $x_i, z_i, \pi_j, \mu_j$  有对应关系：

$$\mathcal{F}(x_i) = z_i$$

$$\mathcal{F}(\pi_j) = \mu_j$$

$\mathcal{F}$  为编码器构成的从原始空间到嵌入空间的非线性映射。可以看出，我们仍然使用自由度为 1 的  $t$  分布来衡量  $z_i$  与  $\mu_j$  的距离，但是当原始空间中对应点到聚类中心距离超过  $\sqrt{2S}$ ，即聚类中心超出“九宫格”范围时，直接将  $r_{ij}$  记为 0。相应， $Q$  分布为

$$q_{ij} = \frac{k_{ij}}{\sum_{j'} k_{ij'}}$$

如此一来，在九宫格以外的聚类中心和嵌入空间点的  $q_{ij}$  就为 0，即嵌入空间点属于该聚类中心的概率为 0，这些聚类中心将不再影响聚类结果。在实际操作中，为了避免发生分母为 0 的情况，用小偏置项（本项目中选择  $1e^{-19}$ ）代替 0，同样可以达到效果。

- **更好的优化算法** 为了更快收敛并减小陷入局部最优解的可能，我们使用 Adam 优化器替代 SGD 进行梯度下降求解。

### 3.3.3 实验

#### 数据集介绍

- **BSD500**<sup>[25]</sup> 包含 500 张大型自然图像的数据集。数据集已被手动分割并标注边界。该数据集常用来作为比较不同分割和边界检测算法的基准。
- **MSRC-v2**<sup>[26]</sup> 包含 456 张 14 类物体图像，包括房子、牛、羊、椅子等种类，常用于目标检测以及语义分割任务。

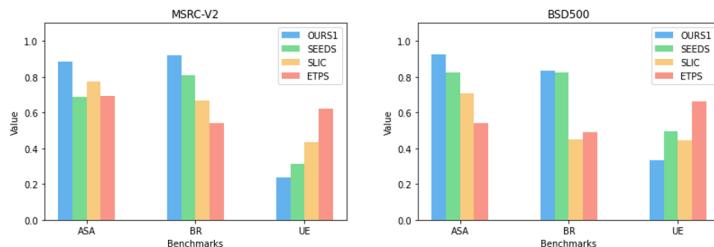


图 17 基于自编码器的超像素分割算法（Ours1）在 BSD500 和 MSRC-v2 上的实验结果与传送算法 SLIC、SEEDS、ETPS 对比

**指标参数** 由图17可见，基于自编码器的超像素分割算法在三项指标上都优于传统算法。

### 3.4 基于自表示机制的超像素分割算法

#### 3.4.1 原理

**自表示机制** 对于高维数据，我们一般相信数据具有某种低维结构，最简单的建模方式就是线性模型，即利用多个线性子空间来近似高维数据，这一个思路可以带来子空间聚类算法。子空间聚类算法中一个常用的方式是自表达 (Self-Expressive) 模型，细致的说，给定数据矩阵  $X = [x_1, \dots, x_N] \in \mathbf{R}^{D \times N}$ ，其中每一列是一个数据点  $x_j \in \mathbf{R}^D$ ，我们需要寻找数据的线性表达，即利用数据自己来确定线性子空间。

$$x_j = \sum_{i \neq j} c_{ij} x_i$$

其中  $\{c_{ij}\}_{i \neq j}$  是自表示系数矩阵，我们希望这一个系数矩阵满足子空间保持 (subspace-preserving) 性质，即矩阵元  $c_{ij}$  不为零说明  $x_i, x_j$  一定在同一个线性子空间上，得到系数矩阵之后可以将数据看成无向图，自表示系数矩阵对称化后 ( $c'_{ij} = c_{ij} + c_{ji}$ ) 看成邻接矩阵，就可以使用谱聚类来将数据聚类。

#### SENet [5]

在自表达模型的应用中，我们一般需要增加正则项以表达先验知识：

$$C = \arg \min_{C \in \mathbb{R}^{N \times N}} \sum_i \left\| x_i - \sum_{i \neq j} c_{ij} x_j \right\|_2^2 + \sum_{i \neq j} r(c_{ij})$$

我们一般还是希望所使用的子空间尽量少，因此最简单的思路正是对矩阵施加稀疏先验  $r(\cdot) = \|\cdot\|_1$ ，但是这样一个稀疏先验会带来很大的 False Negative，即当  $c_{ij} \neq 0$  时不代表  $x_i, x_j$  不处于同一个线性子空间。换句话说，我们施加的稀疏先验过强，实际上我们希望在子空间保持的情况下让矩阵更加稠密，因此我们可以选择一个更加多样性的正则化：将一范数与二范数混合：

$$r(\cdot) = \lambda \|\cdot\|_1 + \frac{1-\lambda}{2} \|\cdot\|_2$$

这样我们就可以得到更稠密的自表示矩阵。

为了将自表示模型与深度学习进行结合从而可以利用神经网络直接推断出矩阵元  $c_{ij}$ ，我们可以将矩阵元  $c_{ij}$  直接使用神经网络参数化，即让神经网络看到  $x_i, x_j$  之后直接给出  $c_{ij}$ ，因此对应的损失函数可以写成：

$$\min_{\theta} \sum_i \|x_i - \sum_{i \neq j} f(x_i, x_j, \theta)\|_2^2 + \sum_{i \neq j} r(c_{ij})$$

我们将神经网络写成：

$$f(x_i, x_j, \Theta) = \alpha \mathcal{T}_b(u_j^T v_i)$$

其中：

$$u_j = u(x_j, \Theta_u) \in \mathbf{R}^p, v_i = v(x_i, \Theta_v) \in \mathbf{R}^p$$

而且  $\mathcal{T}(\cdot)$  是参数可学习软阈值算子：

$$\mathcal{T}_b(t) = \text{sgn}(t) \max(|t| - b)$$

$b$  是可学习参数，结构如图18.

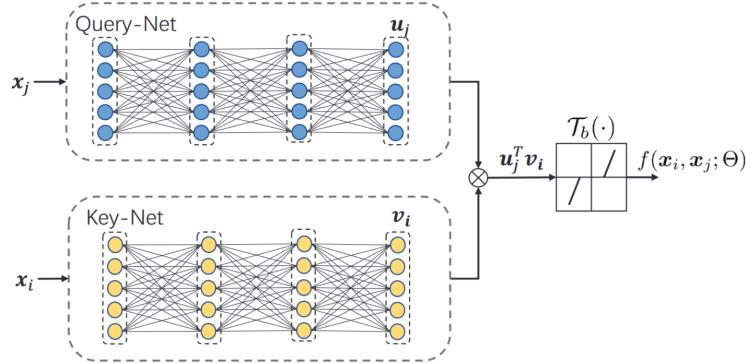


图 18 SENet 网络结构

### 3.4.2 算法描述

**简介** 我们用部分图片训练一个合适的 SENet。分割的时候根据输入的图片 SENet 会输出相应的自表示矩阵。自表示矩阵经过去噪和对称化后得到图片的亲和力矩阵 (affinity matrix)。在亲和力矩阵上，我们采用谱聚类进行聚类并得到最终的结果。

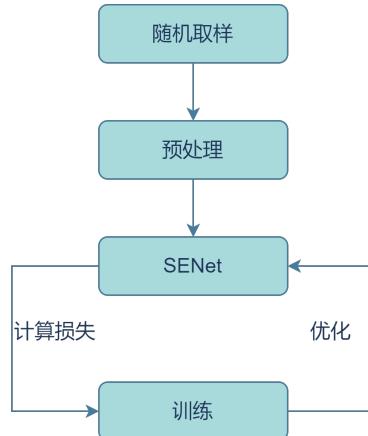


图 19 SENet 训练

### SENet 训练

- 预处理：加入坐标信息并归一化
- 训练 SENet，损失函数为

$$\min_{\theta} \sum_i \|x_i - \sum_{i \neq j} f(x_i, x_j, \theta)\|_2^2 + \sum_{i \neq j} r(c_{ij})$$

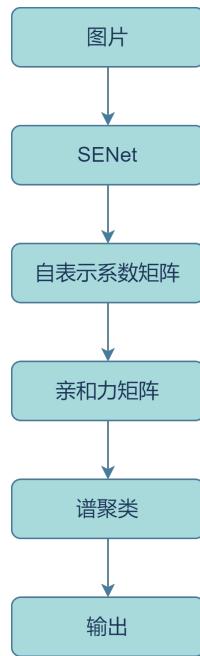


图 20 将 SENet 用于超像素分割

### 进行超像素分割

- 图片输入 SENet 计算得自表示系数矩阵

构建亲和力矩阵选取合适的阈值  $\epsilon$ , 去噪, 过滤掉表示系数较小的像素

$$\text{if } c_{ij} < \epsilon, c_{ij} = 0$$

对称化处理得到邻接矩阵 (affinity matrix)

#### 3.4.3 实验

**SENet泛化性能验证** 为了说明训练得到的 SENet 具有良好的泛化性能, 我们用少量图片训练得到的 SENet 计算其在数据集中其他部分图片的损失函数值。最终损失函数值分布如图 21 了, 损失值标准差为  $2.484 \times 10^{-5}$ 。由此可见, SENet 在少量图片上训练得到的结构可以准确预测其他图片的低维空间结构。我们将其归因于图片中像素信息相对简单, 对于无语义的像素聚类而言, 不同种类的图片像素差异程度并不大。

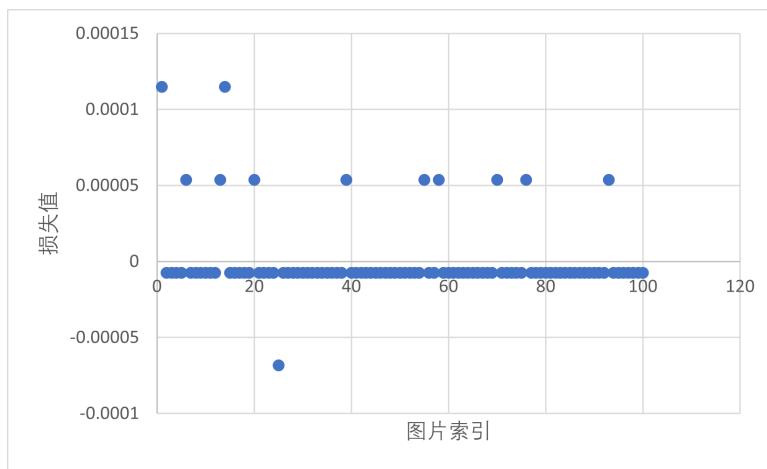


图 21 随机选取 100 张图片获取 SENet 对每张图片的损失值 (已减去均值)

**指标参数** 我们同样选择在 BSD500<sup>[25]</sup> 以及 MSRC-v2 上进行测试<sup>[26]</sup>.

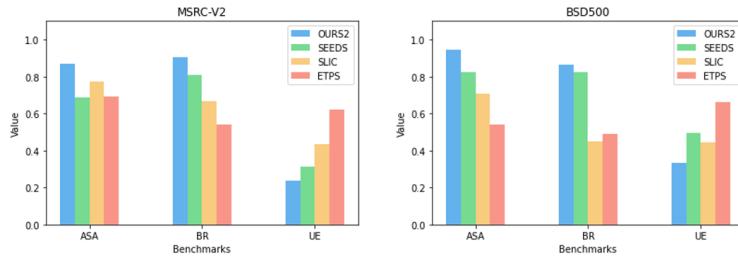


图 22 基于自编码器的超像素分割算法 (Ours2) 在 BSD500 和 MSRC-v2 上的实验结果与传送算法 SLIC、SEEDS、ETPS 对比

## 4 研究设计方案

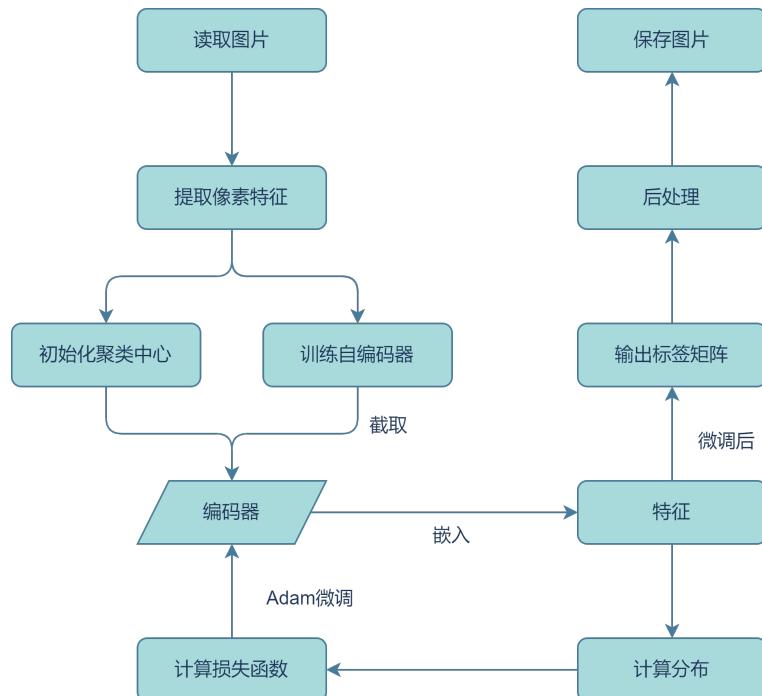


图 23 基于自编码器机制的超像素分割算法流程图

- **提取像素特征** 为了解决输入维度少的问题，我们决定进行特征提取。我们从一篇基于子空间聚类的超像素分割算法<sup>[24]</sup>中得到启发，从原始图像中提取特征，将原始的五维特征提升至 19 维。这十九维特征包括像素的坐标、灰度、梯度、梯度幅值、梯度方向、纹理特征<sup>[21]</sup>。如此高维特征在原始空间聚类效果很差，而我们的编码器就可以将高维特征嵌入到低维空间进行聚类。
- **初始化聚类中心** 此处我们先在原始图像上网格化初始聚类中心，然后将其提取特征并嵌入到低维空间作为嵌入空间的聚类中心。
- **局部的聚类** 为了解决空间约束性差的问题，我们决定对 Q 分布进行改进。首先定义  $k_{ij}$ :

$$k_{ij} = \begin{cases} \frac{1}{1+\|z_i - \mu_j\|^2}, & \|x_i - \mu_j\|^2 > 2S \\ 0, & \|x_i - \mu_j\|^2 \leq 2S \end{cases}$$

其中,  $x_i$  为原始图像空间点,  $\pi_j$  为原始图像空间聚类中心, 且  $x_i, z_i, \pi_j, \mu_j$  有对应关系:

$$\mathcal{F}(x_i) = z_i$$

$$\mathcal{F}(\pi_j) = \mu_j$$

$\mathcal{F}$  为编码器构成的从原始空间到嵌入空间的非线性映射。可以看出, 我们仍然使用自由度为 1 的 t 分布来衡量  $z_i$  与  $\mu_j$  的距离, 但是当原始空间中对应点到聚类中心距离超过  $\sqrt{2S}$ , 即聚类中心超出“九宫格”范围时, 直接将  $r_{ij}$  记为 0。相应, Q 分布为

$$q_{ij} = \frac{k_{ij}}{\sum_{j'} k_{ij'}}$$

如此一来, 在九宫格以外的聚类中心和嵌入空间点的  $q_{ij}$  就为 0, 即嵌入空间点属于该聚类中心的概率为 0, 这些聚类中心将不再影响聚类结果。在实际操作中, 为了避免发生分母为 0 的情况, 用小偏置项(本项目中选择  $1e^{-19}$ )代替 0, 同样可以达到效果。

- **更好的优化算法** 为了更快收敛并减小陷入局部最优解的可能, 我们使用 Adam 优化器替代 SGD 进行梯度下降求解。

#### 4.1 基于自表示机制的超像素分割算法

**简介** 我们用部分图片训练一个合适的 SENet。分割的时候根据输入的图片 SENet 会输出相应的自表示矩阵。自表示矩阵经过去噪和对称化后得到图片的亲和力矩阵 (affinity matrix)。在亲和力矩阵上, 我们采用谱聚类进行聚类并得到最终的结果。

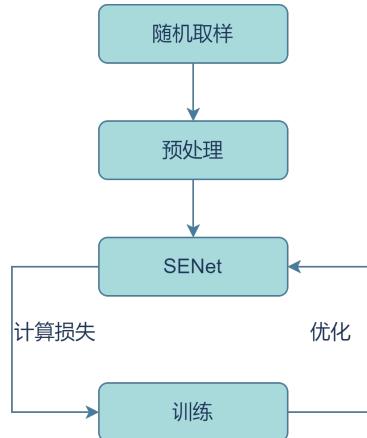


图 24 SENet 训练

#### SENet 训练

- 预处理: 加入坐标信息并归一化
- 训练 SENet, 损失函数为

$$\min_{\theta} \sum_i \|x_i - \sum_{i \neq j} f(x_i, x_j, \theta)\|_2^2 + \sum_{i \neq j} r(c_{ij})$$

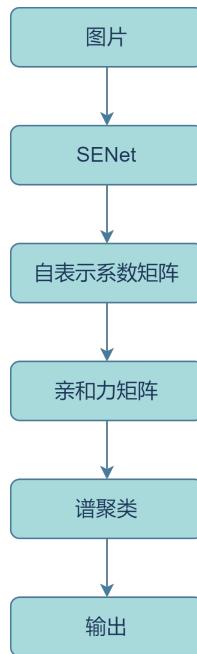


图 25 将 SENet 用于超像素分割

### 进行超像素分割

- 图片输入 SENet 计算得自表示系数矩阵  
构建亲和力矩阵选取合适的阈值  $\epsilon$ , 去噪, 过滤掉表示系数较小的像素

$$\text{if } c_{ij} < \epsilon, c_{ij} = 0$$

对称化处理得到邻接矩阵 (affinity matrix)

## 5 项目成果简介

- 基于自编码器机制的超像素分割算法 提出基于自编码机制的超像素分割算法, 成功将深度聚类方法应用于超像素分割任务并在主要指标上超过之前的方法
- 基于自表示机制的超像素分割算法 提出基于自表示机制的超像素分割算法, 相较于基于自编码器机制的超像素分割算法, 时间效率极大提高
- 搭建介绍项目的网站
- 开源代码仓库

## 6 个人感想

### 6.1 蔡英豪

在历时接近一年的 srtp 项目中, 作为项目负责人, 我收获良多。

首先是科研能力方面。在广泛阅读论文之后, 我逐渐熟悉掌握阅读文献的技巧, 阅读论文的状态也从一开始一篇论文要看许久成长为现在可以快速抓住论文要点与方法。同时, 在项目过程中, 我对超像素分割和聚类算法领域的认知也不断加深, 逐渐掌握了它们之间的联系与差异, 并成功完成了项目工作。

另外, 在工程能力方面, 我也获得了长足的进步。项目之初, 我们就开始学习使用 Git 仓库, 这为我们相互协调项目开源提供了无数便利。项目中, 阅读代码的工作是贯穿其中的。我首先阅读了 SLIC 算法

的源码，了解了超像素分割算法的关注点和领域的特点。然后我在确定将 DEC 算法和 SENet 迁移到超像素分割领域之后，我又去阅读相关代码，深究每一处细节，为最后完成算法的迁移工作打下坚实基础。阅读代码的工作也很大程度上提高了我的代码水平，更加熟悉 python 相关库比如 sklearn、numpy 和 pytorch 等的接口与操作。

## 6.2 牟卓翊

在为期一年的 SRTP 项目中，我收获了很多。一方面是对于科研的具体场景，科研的实际要求与产出目标都有了大概的认识。我不仅对从一个 idea 的产生到最后成果的落地实践有了大概的思考，也对其中的风险性有了深刻的领会。但同时我也感受到了 CV 与无监督学习领域的魅力，未来的数据处理趋势必然是数据驱动，不需要先验基础的无监督学习必然会大放异彩。

另一方面是对自我实际工程能力的不足有了反思，在项目进展过程中，从 ssh 使用学校的科研平台，到自己复现无数论文的 code，这个过程十分的艰辛，今后我将更加注重个人手动实践能力的锻炼。

## 6.3 王玟雯

在这次实践过程中，我很清楚的明白了，选择还有方向和努力一样重要。项目进行的过程中我需要明白，什么事情才是真正值得做的，需要去做些什么，以及如果遇到了困难我是应该继续深究下去、还是巧妙的转化思路。在这次小组组队中，我学会了许多图像处理的知识，极大拓展了我的知识面。并且通过深度接触这些看似已经非常常见的图像应用，我知道了即使是非常稀疏平常的图像开发，预处理过程也是非常重要的。同时我也领悟到小组合作是非常重要的。时间管理，任务进度的管理，项目的管理，还有如何和小组组员进行开发上的交流和互动，这些在一个团队任务中都有着不可言喻的重要性。有时候会有难以求解的问题，这时候就要及时沟通。当然，在问题处理的方面，我也懂得了在技术上遇到问题的时候，第一步不是去询问他人，而是到网络上寻找是否已经有解决了的解答。巧妙地利用谷歌和百度进行问题解决，将使得效率大大的增加。

## 7 展板

**东南大学大学生创新创业成果展示**

**基于深度学习的超像素分割算法**

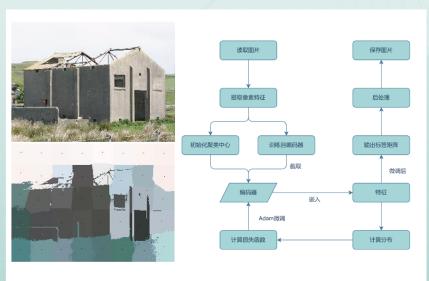
**项目编号** 202209045

**指导教师** 贾育衡 (副教授)

**学生成员** 蔡英豪 (58120127) 牟卓翊 (58120322) 王玟雯 (58120309)

**项目简介**

超像素是一组邻近的且像素信息相似的像素。少量超像素就可以替代图片的原始像素完整表示图片信息。超像素的使用使得下游任务计算负担很大程度上减轻，同时超像素的存在还使得下游任务能够忽略冗余的像素信息，进而达到更好的效果。我们的项目旨在提出基于深度学习的超像素分割算法。基于深度学习的超像素分割算法以深度聚类算法为基础，能够实现更高精度的分割，进而提升下游任务的效果，同时由于深度学习的特点，我们的超像素分割算法还可以根据下游任务动态调整分割策略，对特定任务进行更具有针对性的分割。



The figure shows two images: a real-world scene of a building and its corresponding segmented version where the building is divided into superpixels. To the right is a flowchart titled 'Algorithm Segmentation Results and Flowchart'. The process starts with '读取图片' (Read Image), followed by '提取超像素' (Extract Superpixels). This leads to '划分超像素中心' (Divide Superpixel Center) and '训练后处理' (Post-processing after training). These steps are iterative, with '划分超像素中心' leading back to '提取超像素'. Finally, the process goes through '输出结果' (Output Result), '特征' (Features), '计算分布' (Calculate Distribution), '特征输入' (Feature Input), 'Aster咨询' (Aster Consulting), '计算损失函数' (Calculate Loss Function), '梯度下降' (Gradient Descent), '迭代' (Iteration), '输出结果' (Output Result), '特征' (Features), '计算分布' (Calculate Distribution), and ends with '保存图片' (Save Image).

**研究成果**

- 提出基于自编码机制的超像素分割算法，成功将深度聚类方法应用于超像素分割任务并在主要指标上超过之前的方法
- 提出基于自表示机制的超像素分割算法，相较于基于自编码器机制的超像素分割算法，时间效率极大提高
- 搭建介绍项目的网站
- 开源代码仓库

**创新点**

- 创造性使用深度聚类的方法“自底向上”构建超像素
- 结合超像素分割的特点，将空间约束引入深度聚类算法中，提升了算法性能
- 不同于使用有监督学习进行超像素分割的网络，我们的网络不需要人工标注，节约人力成本，更加顺应大数据时代潮流
- 算法可以与下游任务协同训练，更具有针对性

图 26 展板

## 参考文献

- [1] ACHANTA R, SHAJI A, SMITH K, et al. Slic superpixels compared to state-of-the-art superpixel methods [J]. IEEE transactions on pattern analysis and machine intelligence, 2012, 34(11): 2274-2282.
- [2] YANG F, SUN Q, JIN H, et al. Superpixel segmentation with fully convolutional networks[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 13964-13973.
- [3] XIE J, GIRSHICK R, FARHADI A. Unsupervised deep embedding for clustering analysis[C]//International conference on machine learning. PMLR, 2016: 478-487.
- [4] JAMPANI V, SUN D, LIU M Y, et al. Superpixel sampling networks[C]//Proceedings of the European Conference on Computer Vision (ECCV). 2018: 352-368.
- [5] ZHANG S, YOU C, VIDAL R, et al. Learning a self-expressive network for subspace clustering[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 12393-12403.
- [6] REN, MALIK. Learning a classification model for segmentation[C/OL]//Proceedings Ninth IEEE International Conference on Computer Vision. 2003: 10-17 vol.1. DOI: 10.1109/ICCV.2003.1238308.
- [7] STUTZ D, HERMANS A, LEIBE B. Superpixels: An evaluation of the state-of-the-art[J/OL]. Computer Vision and Image Understanding, 2018, 166: 1-27. <https://www.sciencedirect.com/science/article/pii/S1077314217300589>. DOI: <https://doi.org/10.1016/j.cviu.2017.03.007>.
- [8] STUTZ D, HERMANS A, LEIBE B. Superpixels: An evaluation of the state-of-the-art[J/OL]. Computer Vision and Image Understanding, 2018, 166: 1-27. <https://www.sciencedirect.com/science/article/pii/S1077314217300589>. DOI: <https://doi.org/10.1016/j.cviu.2017.03.007>.
- [9] MARTIN D, FOWLKES C, MALIK J. Learning to detect natural image boundaries using local brightness, color, and texture cues[J/OL]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2004, 26(5): 530-549. DOI: 10.1109/TPAMI.2004.1273918.
- [10] LEVINSHTEIN A, STERE A, KUTULAKOS K N, et al. Turbopixels: Fast superpixels using geometric flows [J/OL]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2009, 31(12): 2290-2297. DOI: 10.1109/TPAMI.2009.96.
- [11] LIU M Y, TUZEL O, RAMALINGAM S, et al. Entropy rate superpixel segmentation[C/OL]//CVPR 2011. 2011: 2097-2104. DOI: 10.1109/CVPR.2011.5995323.
- [12] MOORE A P, PRINCE S J D, WARRELL J, et al. Superpixel lattices[C/OL]//2008 IEEE Conference on Computer Vision and Pattern Recognition. 2008: 1-8. DOI: 10.1109/CVPR.2008.4587471.
- [13] SCHICK A, FISCHER M, STIEFELHAGEN R. Measuring and evaluating the compactness of superpixels [C]//Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012). 2012: 930-934.
- [14] ACHANTA R, SHAJI A, SMITH K, et al. Slic superpixels compared to state-of-the-art superpixel methods [J/OL]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2012, 34(11): 2274-2282. DOI: 10.1109/TPAMI.2012.120.
- [15] NEUBERT P, PROTZEL P. Compact watershed and preemptive slic: On improving trade-offs of superpixel segmentation algorithms[C/OL]//2014 22nd International Conference on Pattern Recognition. 2014: 996-1001. DOI: 10.1109/ICPR.2014.181.
- [16] LI Z, CHEN J. Superpixel segmentation using linear spectral clustering[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2015.
- [17] VAN DEN BERGH M, BOIX X, ROIG G, et al. Seeds: Superpixels extracted via energy-driven sampling [C]//FITZGIBBON A, LAZEBNIK S, PERONA P, et al. Computer Vision – ECCV 2012. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012: 13-26.

- [18] YAO J, BOBEN M, FIDLER S, et al. Real-time coarse-to-fine topologically preserving segmentation[C]// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2015.
- [19] VINCENT P, LAROCHELLE H, LAJOIE I, et al. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.[J]. Journal of machine learning research, 2010, 11 (12).
- [20] WANG W, HUANG Y, WANG Y, et al. Generalized autoencoder: A neural network framework for dimensionality reduction[C]//Proceedings of the IEEE conference on computer vision and pattern recognition workshops. 2014: 490-497.
- [21] HE K, CHEN X, XIE S, et al. Masked autoencoders are scalable vision learners[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022: 16000-16009.
- [22] XIE J, GIRSHICK R, FARHADI A. Unsupervised deep embedding for clustering analysis[C]//International conference on machine learning. PMLR, 2016: 478-487.
- [23] VAN DER MAATEN L, HINTON G. Visualizing data using t-sne.[J]. Journal of machine learning research, 2008, 9(11).
- [24] LI H, JIA Y, CONG R, et al. Superpixel segmentation based on spatially constrained subspace clustering[J]. IEEE Transactions on Industrial Informatics, 2020, 17(11): 7501-7512.
- [25] MARTIN D, FOWLKES C, TAL D, et al. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics[C]//Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001: volume 2. IEEE, 2001: 416-423.
- [26] SHOTTON J, WINN J, ROTHER C, et al. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context[J]. International journal of computer vision, 2009, 81(1): 2-23.