深度学习 期末大作业

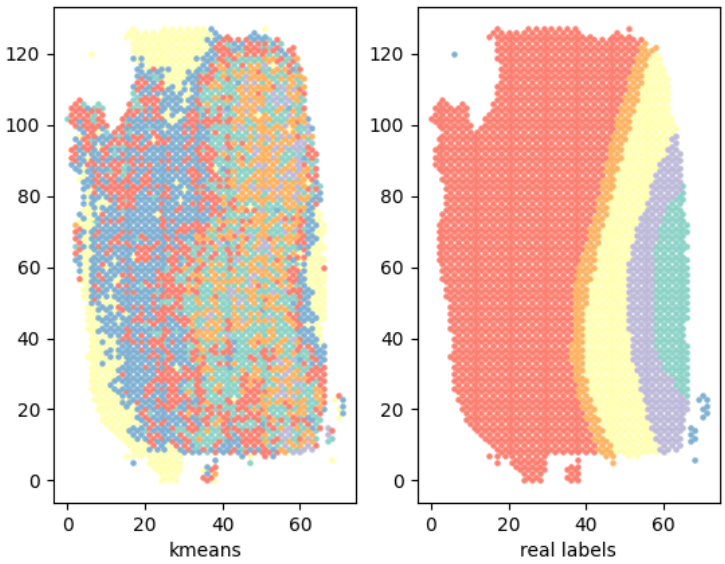
201300096 人工智能学院 杜兴豪

目标分析

针对这次作业，我对h5py包进行一定的了解后，读取了给定数据集文件中的有效信息，文件包含三个种类的数据：X,Y,pos。其意义分别为：

- X：单细胞RNA-seq测序基因表达信息的read count
- pos：需要聚类的点的位置信息
- Y：真实标记，用于测试和反映聚类效果

我首先利用kmeans算法对原数据进行了简单的聚类：



其中的NMI和ARI分别为：

- NMI：0.16120421890797168
- ARI：0.11571031184405306

观察到聚类结果和真实标记出入很大，但并非完全无关：可以看出最右边中心区域，两个结果的近似是很明显的，并且中间在类别变化较为频繁的部分，直接对原始数据进行聚类的结果也有一定程度上的区分，但是并不清晰。查阅资料得知可能是因为基因测序信息的稀疏和噪点影响所致。因此需要降维后再进行聚类。

深度聚类方法

在尝试了诸多模型和方法却始终得不到良好的结果后（见"几种尝试"部分），既然深度学习降维和kmeans聚类方法分开使用都难以提高聚类效果，那如果采用深度学习方法来聚类，让模型迭代式地将聚类结果变得更好，也许会得到更优秀的聚类结果。查阅相关资料，我决定采用深度聚类模型，来尝试解决这个问题。模型思路如下：

- 利用自编码器对原始数据进行降维，学习低维下的特征表示
- 利用训练好的自编码器模型，加入一个参数，用来迭代聚类中心
- 针对聚类中心，再次训练自编码器模型，同时继续改进模型降维部分的相关权重，使其降维后的数据更具有聚类特征性

模型建立

模型的结构如下：

```
myDeepCluster(
  (encoder): Sequential(
    (0): Linear(in_features=1011, out_features=256, bias=True)
    (1): ReLU()
    (2): Linear(in_features=256, out_features=64, bias=True)
  )
  (decoder): Sequential(
    (0): Linear(in_features=64, out_features=256, bias=True)
    (1): ReLU()
    (2): Linear(in_features=256, out_features=1011, bias=True)
  )
  (recon_loss): MSELoss()
)
```

其中输入维度为数据集中每条数据基因信息的维度，随样例文件变化而不同。

在自编码器训练阶段，我采用了以下的参数和相关配置：

- 学习率：0.001
- 训练轮数：200
- batch size：256
- 优化器：`torch.optim.Adam()`
- 损失函数：`nn.MSEloss()`

训练之前先对原始数据进行规范化，把X矩阵中的每个元素都映射到0和1之间的范围，消除绝对值大小对训练的阻碍，提高训练过程中的收敛速度。采用的规范化公式为：

$$x'_{ij} = \frac{x_{ij} - \min x_{\_j}}{\max x_{\_j} - \min x_{\_j}}$$

观察到和做规范化操作之前相比，做完规范化操作后损失降低变得平滑，并且速度加快。

在迭代聚类过程中，我首先初始化了一个参数`self.centers`，用来存放聚类中心。在每轮训练的过程中，可以根据聚类中心计算出对数据集中所有点的软分配概率，根据相关文献，我们可以确定这里的软分配分布服从自由度为1的student分布[1]，再根据软分布情况P和真实分布Q之间的KL散度，来衡量聚类效果，作为损失函数，因此有：

$$cluster\_loss = KL(P||Q)$$

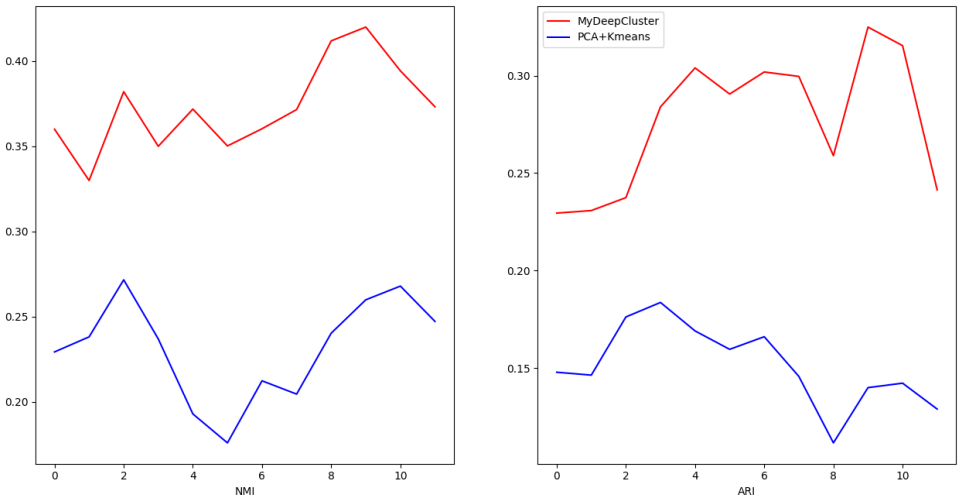
以此构建聚类网络，每次迭代更新聚类中心。注意到这里的聚类中心是通过梯度下降来迭代的，因此最后结果中聚类中心将不是数据集中一个真实存在的点。聚类过程中相关参数和配置如下：

- 学习率：0.01
- 训练轮数：100
- 优化器：`torch.optim.SGD()`
- 损失函数：`nn.MSEloss() + KLloss()`

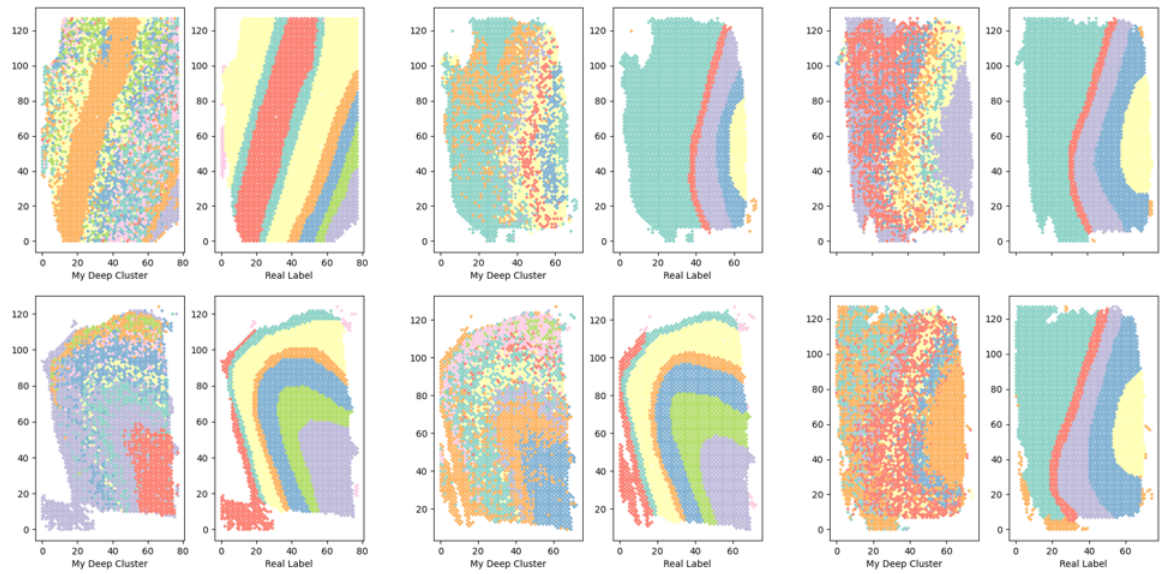
聚类中心更新完毕后，将重新对数据集中的每个点进行软指配，将其划分到概率最高的类中。划分完毕后，计算本轮分配的loss，NMI，ARI。最终从每轮迭代中挑选NMI和ARI指数最高的作为模型输出。

效果展示

通过对所有样例文件进行聚类，得到NMI和ARI数据后绘制图表如下。其中红线是我实现的深度聚类算法，蓝线是通过PCA降维到和AE结果维度一致后，通过kmeans进行聚类得到的结果。



观察到我的实现对聚类的准确度提升很大。一些聚类结果展示如下：



从我的聚类结果中，可以清楚地看出真实聚类的形状和轮廓，证明方法有效。

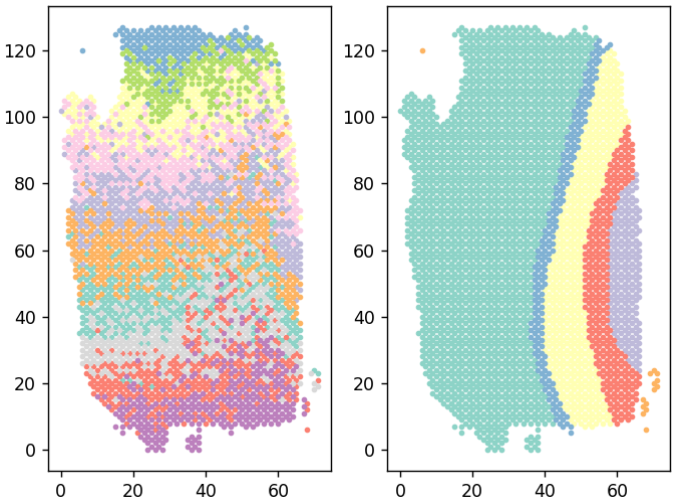
几种尝试

在开始最终的模型之前，我对本次目标进行了各种方向的摸索。

意义不明的摸索

在起初，我对文件中给定数据还意义不明的时候，曾认为X中是每个数据点的基因内容，pos是数据点在生理系统中的具体位置。因此我曾将X和pos拼接成一个大的矩阵，将其作为每个数据点的属性，借用AE来把它降到2维，形成新的坐标，作为用于聚类的数据。最后将得出的聚类标记按照pos来绘图，得出聚类结果。

最后的结果由于位置信息的干扰，不具有规律性。为了找出我这种聚类方法的问题，我观察了X中的内容。发现X中含有大量的0，并且其值也很小，而pos中记录的值都较大，降维结果一定着重包含pos中的信息，而X中的信息将被忽视。为了解决这个问题，我对这个整体进行了标准化操作，让pos和X的数据范围相近，以此进行降维并聚类，得到了以下的图：



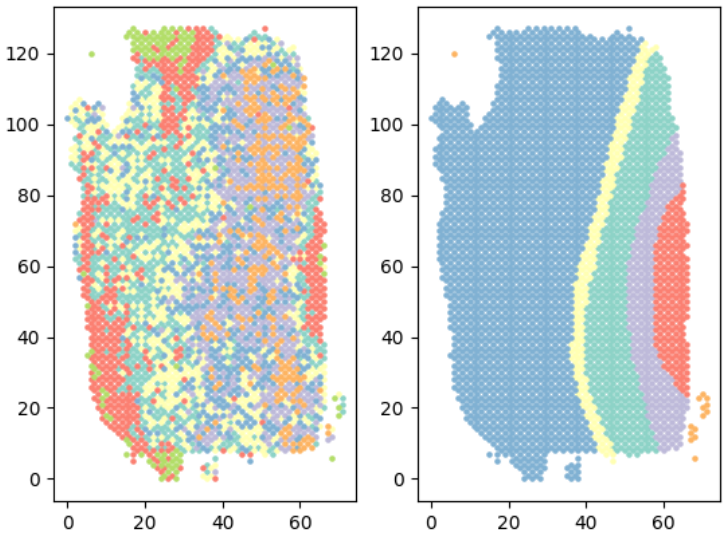
结果居然显现出了纵向的分层！这直接让我质疑我对数据的初始操作有误。也正是有了这次尝试，我才开始正确分析X和pos的信息内容，才能真正开始任务。

基于Scanpy的降维和聚类

由于本次任务的目标是单细胞RNA测序数据的降维和聚类过程，因此自然想到利用python自带的处理单细胞基因数据的包Scanpy来做，具体过程可以描述为：

- 利用PCA算法，将原始数据的维度降低到30维
- 利用louvain聚类算法，对降维后的数据进行聚类分析，得出聚类标签
- 根据标签进行绘图和相应指标的计算

根据上述步骤，得出实验结果如下（样例文件为sample_151670.h5）：



其中相关指标的计算结果分别为：

- NMI：0.15512000
- ARI：0.08635000

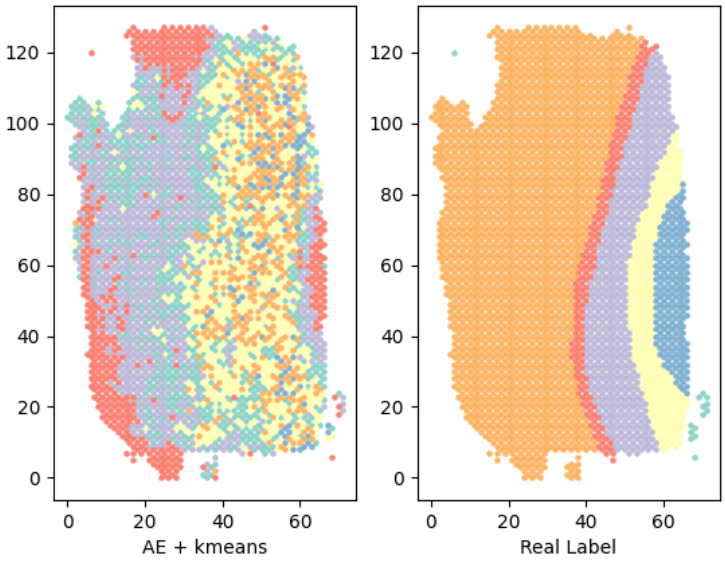
效果并不理想。查阅相关资料得知，通过PCA等线性算法对基因信息进行降维，会导致其中的生物学意义变得不可解释，以至于可能丢失关键信息。因此接下来的方式将不采用线性方式降维。

基于AE的降维

由于基因信息的生物信息不能用线性方式表示，因此容易想到用同样不可解释的非线性方法来拟合其降维过程，从而筛选出包含生物生物学意义的重要特征。因此我利用到自编码器来对它进行降维处理。自编码器的数据如下：

- learning rate=0.001
- batch size=256
- hidden size=(input_dim + latent_dim) * 2 / 3
- latent size=30（最终维度）
- optimizer: `torch.optim.Adam()`
- loss function: `torch.nn.MSELoss()`

模型训练过程中，观察到采用MSE损失时，训练到第5000轮就可以达到1e-6数量级的误差，因此停止训练。根据encoder的降维结果，重新利用kmeans对新数据进行聚类，结果如下：



观察到在数据类别较少的左侧部分，降维后的数据表现比原始数据要好，左半部分数据类别判断纯度也有了提升。此时的NMI和ARI分别为：

- NMI：0.16827275796598865
- ARI：0.12584583412826303

观察到两种度量指标均和直接聚类的结果有了提升，得知AE的编码结果确实具有去噪的作用。

基于MLP进行半监督聚类

我还尝试过半监督学习思路来做这个聚类问题。注意到大部分的数据集都具有相同的聚类总数，因此我希望通过一个深层的神经网络来学习到基因表达之间的联系，以及其对聚类的影响，从而帮助无标签数据集进行聚类。聚类时对数据的降维体现在每个数据点的权重上：无关紧要的基因赋予较低的权重，对聚类结果影响不大。

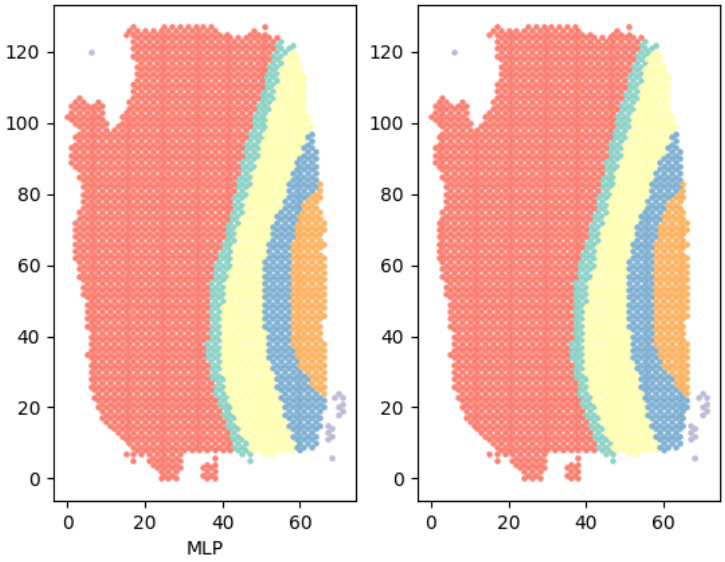
根据这个思路，我希望通过在某些数据集上带标签学习，通过标签来学到获得一个能精准分类的多层感知机之后，再利用这个精准分类模型，去预测别的基因表达结构的聚类情况。实现流程为：

- 在某些数据集上利用所给标签，对其进行独热编码后投入网络学习
- 利用训练好的网络来聚类其他无标签数据集

网络结构显示如下：

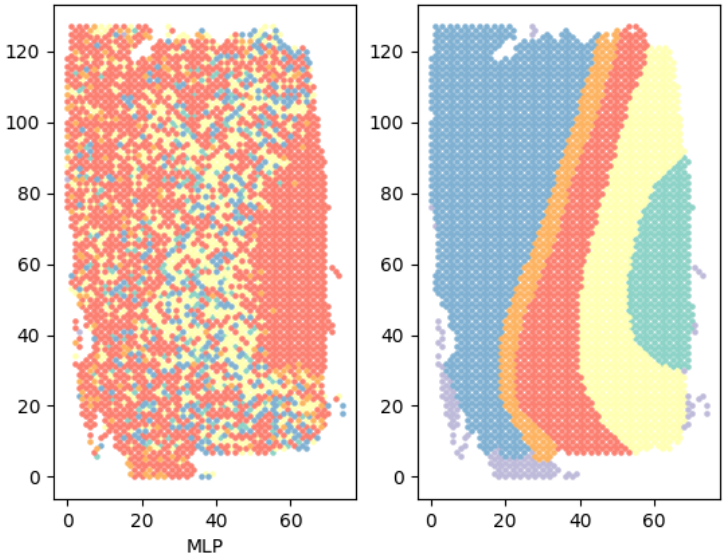
```
myMLP(
  (model): Sequential(
    (0): Linear(in_features=300, out_features=316, bias=True)
    (1): ReLU()
    (2): Linear(in_features=316, out_features=316, bias=True)
    (3): ReLU()
    (4): Linear(in_features=316, out_features=6, bias=True)
  )
)
```

其中输入维度为300，是通过自编码器进行降维后获得的数据，目的是统一所有文件中的输入维度，便于之后预测。模型在训练集上的训练成果如下：



- NMI:1.0
- ARI:1.0

可见MLP在训练集上的分类可以达到100%成功，然而在测试集上的成果却很难让人满意：（训练集采用sample_151670.h5，测试集采用sample_151672.h5）



- NMI:0.06735152922731469
- ARI:0.031660488753541084

预测误差之大让我放弃了通过半监督学习来使模型获得从基因信息直接映射到聚类的权重。但是经过这次尝试，也让我对数据集给出的X、Y、pos三种数据之间的区别和联系得到了更加深的了解。

收获和感悟

这次作业让我收获颇丰：为了理解任务目标，我阅读了大量的论文和代码；为了获取数据，我学到了一种新的文件类型h5，以及几种很好用的Python包，如h5py, Scanpy等，最重要的是，这次作业为我提供了一次宝贵的“工作经历”：

这次大作业有很多不一样的地方：作业涉及到的细胞基因等内容，都是我们之前从未踏足过的领域。因此，可能更需要文档的精确解释。然而这次作业的文档却很精简，在开始任务前我对这次的任务的全部认知可能只有：这次作业好像是一个生物问题，好像要降维和聚类，好像没给框架。在真正开始做降维工作之前，就连读取文件内容这样最简单的预处理工作，都难以前进一步。在助教和同学的帮助下，我最终放弃了读取.Rdata文件，转而读取更轻量化的spca_dat文件下的.h5文件。文件读取成功了，可是内容又是啥？要做聚类，总得有坐标吧，pos看起来像坐标，但是对pos进行聚类，好像不需要什么深度学习算法，但是对X做聚类，那pos又是干啥的……诸如此类的问题如雨后春笋。此外，给定的文件中为数不多的代码又是R语言所写的，对我们这些只学过C和python的同学又难以阅读，只能另辟蹊径，找到和这次作业有关的一些内容，从而了解这次作业的具体目标。也是因为对目标的难以确定，这次作业同样培养了我的搜索能力。最后是工作的具体内容，代码调试，网络训练等我们一直在做的工作，在这次任务中也同样重要。在写这篇报告的时候，我突然醒悟过来：这次作业的全部过程，也许正是我们工作中所要面对的难题：公司派来的任务不会管你学过什么内容；上级不会为了打工人能弄明白任务而花费时间写文档。这些都是我们需要自己去克服的，而这次作业恰好给我们提供了这样的机会。

做完作业，再回头读文档，才能看出文档的内容十分详尽，很多花费了大量时间去寻找的论文就在文档中有所体现，在了解了很多单细胞RNA测序数据相关的工作之后，才发现文档中对数据的描述其实已经说的很明白了。在探索的过程中，我也见识到了许多优秀的聚类算法，理解了很多深度聚类算法的原理，加深了对于自编码器用于降维方面的作用的印象。这次任务让我明白了深度学习在生物方向上的广阔应用前景，也让我见识到了自己的知之甚少，发现了自己在独立分析项目时的片面和弱小。在将来，我会更加努力去学习并应用深度学习算法，在新的领域中开拓出一片属于深度学习的广阔天地。

运行方法

代码框架组织如下：

```
-myDeepCluster
    -report.pdf              # 实验报告
    -readme.md
    -code
        -ScanpyBased.py
        -AEbased.py
        -VAEbased.py
        -MLP.py
        -model
        -myDeepCluster.py    # 最终实现的模型
        -NMI_ARI.py
        -data_process.py
        -figures
    -spca_dat                # 存放实验数据集
```

对于最终的模型：

- 若要对单个样例文件进行聚类，则需要运行myDeepCluster.py文件，修改其中path为对应想要聚类的文件名（默认为`sample_151510.h5`）
    - 默认显示训练进程。若要关闭训练进程展示，则需要取消注释文件中`model.show=False`的注释
- 若要获得最终的NMI和ARI的对照图，请直接运行NMI_ARI.py文件
- 若需要生成并更新figures文件夹下的聚类图，则将NMI_ARI.py文件中的全局变量`ifplot`设置为True即可。

如果想观察其他尝试中的模型的聚类过程，则需要运行对应的文件：

- AE: AEbased.py

- VAE: VAEbased.py
- MLP: MLP.py
- Scanpy: ScanpyBased.py

它们都从data_process.py中读取数据集，若要更改默认的数据集，可以：

1. 改变data_process.py中的全局变量path为对应的数据集的路径
2. 在模型文件中传入路径，不适用默认值

readme文件和本部分相同。

参考文献

1. 1.Tian, Tian, Ji, et al. Clustering single-cell RNA-seq data with a model-based deep learning approach[J]. Nature Machine Intelligence, 2019.