专家检索研究工作按其检索方式可分为两大类：基于主题的专家检索与基于文本的专家检索

同构学术网络：每个学术文档节点都有一个三元组来描述该文档的发表会议、主题和作者。若两篇学术文档存在边，则意味着它们至少具有一个相同的三元组元素。

具体来说，本文首先针对学术文档显式关系的类型和强度，设计了一个(k,P)-core紧密社区模型（3.2.2小节），用于获取具有紧密的同类型关系的文档集合，然后基于上述集合提出基于采样的三元组数据生成方法（3.2.3小节），利用三元组损失函数以及上述训练数据对预训练模型进行微调（3.2.4小节），以提升文档表示质量，优化专家检索的精度。在检索效率方面，基于上述学术文档表示，设计一种基于邻近图的索引(PG-Index)（4.2.1小节），并提出一种基于阈值算法Threshold Algorithm (TA)（4.2.3小节）的Top-n专家检索算法，从而快速获得前n位最匹配的专家。此外，本文对上述解决方案进行以下优化以提升其精度和效率：在学术文档表示的基础上，引入面向专家的(𝑘, 𝒫)-core紧密社区概念，进行二次学习以获得高质量的专家向量表示（5.2.2小节）；通过计算专家向量与查询匹配的匹配程度，直接返回与用户查询最匹配的Top-n专家，进一步优化专家检索时间。

**候选者成模型（条件概率）**

**主题生成模型（贝叶斯）**

基于学习模型是使用训练数据自动构建排名模型或者表示模型。

基于表示的方法首先对两个查询和文本分别使用编码器进行编码，得到相应的文本语义特征向量表示，然后使用相似度度量的方法判别文本的匹配程度。（嵌入）

**（1） 基于异构学术网络嵌入的文档表示方法（第三章）**

**首先提出了一个有效的社区查询算法（3.2.2小节），以高效的查询具有紧密关系学术文档社区。在此基础上，采样得到三元组训练数据（3.2.3小节），然后利用三元组损失函数进行对预训练模型进行微调，嵌入学术文档间的关系信息，提高文档的表示效果（3.3.4小节）。**

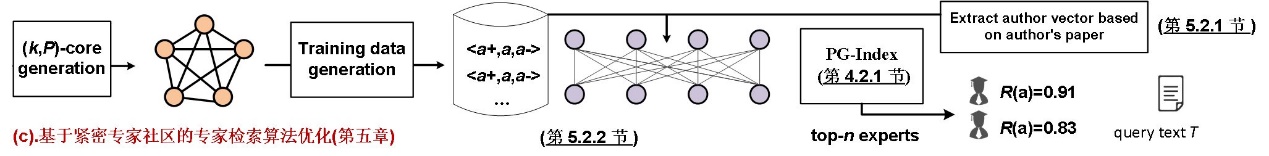
**（2） 基于学术文档表示的Top-n专家检索算法（第四章）**

**基于上述学术文档的表示结果，设计一种基于邻近图的索引(PG-Index)，通过该索引高效地返回与用户查询最匹配的Top-m学术文档（4.2.1小节）。然后，提出一种基于阈值算法Threshold Algorithm (TA)的Top-n专家检索算法，从Top-m学术文档中快速获得Top-n位最匹配的专家**

**（3） 基于紧密专家社区的专家表示方法及专家检索优化（第五章）**

**本章提出“查询文本→专家”的高效检索方法，主要研究设计了一个基于层次聚类的专家表示方法，具体包括：基于层次聚类的初始专家特征向量提取方法**

**上述步骤中需要首先检索到最为相似的学术文档，然后利用检索到的学术文档结果作为专家的“专业证据”对候选专家进行排序评分，这一流程中由于引入了获取Top-m相关学术文档的步骤，从而增加了专家检索的总时间。为了进一步优化上述解决方案的性能和精度，本文提出一种优化方案。如图2.4所示，在学术文档向量的基础之上，再次考虑作者之间的关系(例如A-P-A表示作者有直接的合作关系)直接对作者进行向量表示。在检索过程中通过输入的查询文本直接找到最为相关的Top-n位专家向量，进一步优化了检索的效率。**

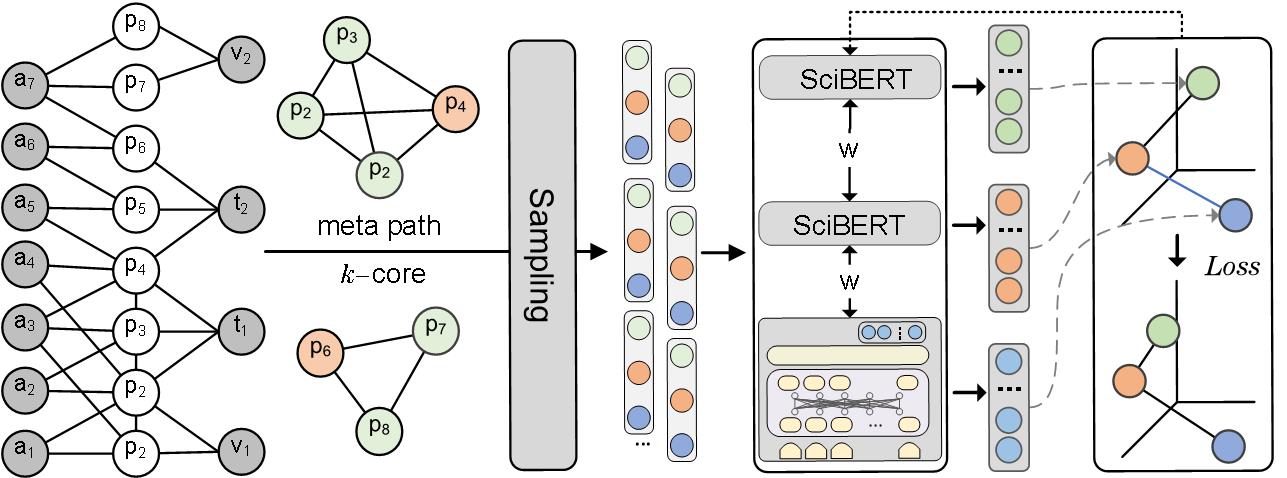


**图2.4基于紧密专家社区的专家检索示意图**

**具体来说，首先，根据利用作者a\_i以及已经学习学术文档嵌入E ⃗\_(a\_i )和通过层次聚类的方法提取专家初始向量组A[a\_i ]=\{(a\_(i,j) ) ⃗,⋯,(a\_(i,n) ) ⃗\}。由于专家可能在多个领域都有所发展，所以每一位专家的初始向量的个数可能是不同的，而具体数量是根据的聚类算法确定的（5.2.1小节）。其次，利用提出的(𝑘, 𝒫)-core查询算法，通过元路径P=A-P-A，得到的高紧密度包含合作关系专家社区。然后，根据给定的专家社区进行专家正负样本三元组采样数据进行模型训练（5.2.2小节），从而对初始专家向量进行优化，最后对专家向量表示进行索引构建。**

**首先，设计了一个异构网路上的社区查询算法，以高效的查找到特定关系和强度的紧密社区。其次，根据查询得到的紧密社区通过基于正负样本采样的方法获得三元组训练数据，最后，再利用三元组损失函数在在预训练模型的基础上进行微调。**

**本文设计了一种采样方法来获取正样本学术文档（属于(𝑘, 𝒫)-core的绿色节点）和负样本学术文档（不属于(𝑘, 𝒫)-core的蓝色节点）。对于给定的种子节点（从(𝑘, 𝒫)-core中选择的红色节点）作为训练数据（第3.2.3节）。最后，本文应用一个预训练的表示模型作为编码器，将学术文档的文本语义和结构关系组合成一个向量，并定义一个三元组损失函数来微调预训练模型（第3.2.4节）。**



**(𝑘, 𝒫)-core：**

* **社区查询**

给定同构网络和输入节点，社区搜索旨在找到包含的最大子图，其中所有节点都具有更大的紧密度。回到本文的案例，给定一个异构网络、元路径和种子节点，期望找到包含的(𝑘, 𝒫)-core作为社区的。

**额外的解决方案：**

**作为替代方法，它主要包括两个主要步骤：（1）标记搜索，（2）清理节点。第一步，它使用标记搜索方法找到通过路径实例P连接到种子节点p\_s的每个节点。每个访问过的学术文档节点，以确保每篇学术文档只被访问一次。然后，它从所有探索过的学术文档中重复上述过程，直到找不到更多的P-neighbor。最后，本文得到一组可能涉及(𝑘, 𝒫)-core的候选学术文档，用S表示。这种BFS遍历避免了列举一些不必要的学术文档。**

**优化版本：**

**（1）在标记搜索步骤中，如果这篇学术文档至少有k个P-neighbor，那么从一个访问过的学术文档节点扩展搜索空间。否则，修剪这篇学术文档并停止从中扩展搜索空间。**

**（2）不仅返回了一个包含种子节点Ps的严格(𝑘, 𝒫)-core社区，而且还包括度数小于k的少量P-neighbor。**

**基于正负样本采样的三元组训练数据生成：**

**（1）种子学术文档选择**

**（2）正负样本采样：近邻负样本法。实际上，这些学术文档与随机学术文档相比与G\_P^k更为接近。因此，对于每个正样本p^+∈G\_P^k，从D 中随机选择r篇学术文档作为负样本，形成r个三元组。实验发现r = 3足以在训练效率和专家检索效率之间取得平衡。并且使用邻近负样本比随机负样本能取得更好效果。**

对于给定一组三元组，文档网络嵌入训练分为三个步骤：（1）本文使用预训练模型将每篇学术文档的文本信息编码为初始语义表示。（2）本文基于正负样本利用三元组损失来微调每篇学术文档的初始语义表示，更准确地说，使用三元组训练数据来调节预训练模型参数。（3）通过梯度下降训练模型。

（1）**种子学术文档选择。**给定元路径，本文通过简单的随机抽样从异构网络中随机选择一部分学术文档节点作为本文的种子学术文档，即，，其中表示的所有学术文档节点（节点类型为）。对于每篇种子学术文档计算一个包含的(𝑘, 𝒫)-core，并利用该社区生成一组训练数据。经过将每个(𝑘, 𝒫)-core社区为每个生成的所有训练数据组合起来，得到最终的训练数据。

（2）**正负样本采样**。首先给出正负样本定义，进而讨论如何有效获取样本。

**定义 6：**正样本。给定一个包含种子学术文档的(𝑘, 𝒫)-core，，本文将中的每个学术文档节点（不包括）定义为正样本，用表示。

**定义 7：**负样本。给定一个包含种子学术文档的(𝑘, 𝒫)-core，本文将 中的每个学术文档节点定义为一个负样本，用表示。

请注意，包含的(𝑘, 𝒫)-core通常是一个小的、有强紧密关系的子图，其中包含少量学术文档，因此本文直接使用中的所有学术文档作为正样本，相反，中的学术文档数量明显大于。因为随着训练数据规模变大，训练时间显着增加，所以不能直接将中的所有学术文档作为负样本。假设本文在中有100篇学术文档，在中有105篇学术文档，那么一个社区就有107个三元组。更糟糕的是，正负样本数量的不平衡会导致训练后的模型具有明显的偏差[39]，负样本收集策略对训练效率和效果有重要影响。因此，本文提出了两种不同的负样本收集策略。

**近****邻负样本**。正如算法1中提到的，本文将那些少于个-neighbor的学术文档节点添加到删除队列中进行剪枝。实际上，这些学术文档与随机学术文档相比与更为接近。因此，对于每个正样本，从中随机选择篇学术文档作为负样本，形成个三元组。实验发现足以在训练效率和专家检索效率之间取得平衡。并且使用邻近负样本比随机负样本能取得更好效果。

**训练数据生成。**从这些正负样本中，将三元组作为一组训练数据。对于三元组中的每篇学术文档节点，本文将学术文档的文本信息作为嵌入模型的输入，即，其中。（字符串拼接）

基于预训练模型的网络训练

（1）**编码器。**模型采用预训练模型SciBERT[40]作为编码器，因为它已经针对学术文档进行了优化，更适合表示学术文档内容。根据SciBERT的输入格式的要求，首先使用Wordpiece[41]将每篇学术文档的文本信息拆分为一个个的令牌(单词)。如果令牌的数量超过SciBERT的默认最大长度（512），将截断文本超出部分以满足输入限制。

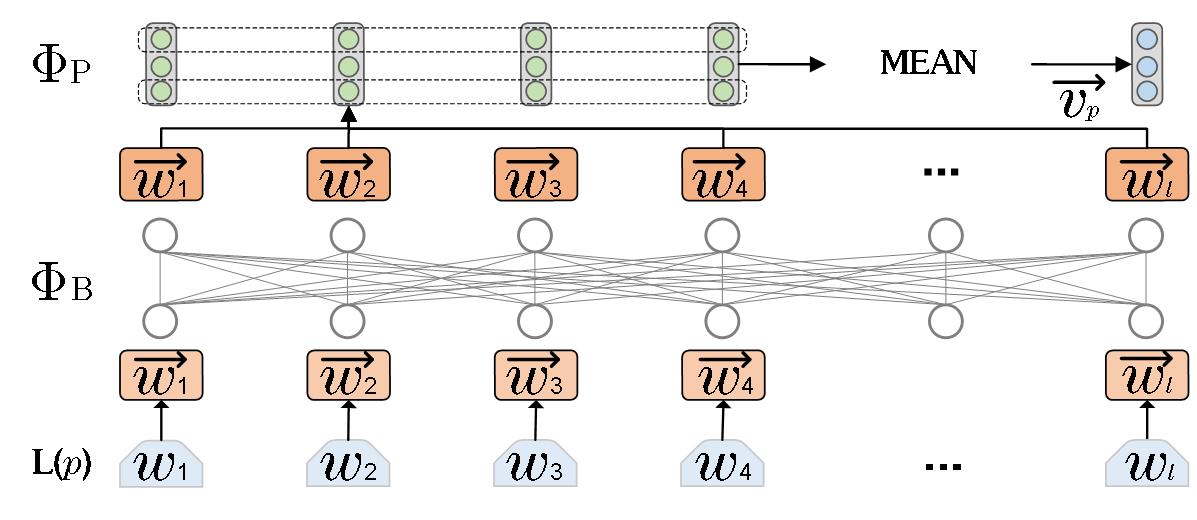


图3.2文档编码器示意图

图3.2展示了本文编码器简单结构，对于给定一个学术，其文本内容被Wordpiece标记，每个表示中的一个令牌(单词)。文档编码器旨在获得的初始向量表示，表示为：

请注意，是SciBERT的参数：用于通过使用对每个令牌 进行编码。然后，本文通过池化方法从中提取最有价值的特征并生成的表示，即。对于文档级别的表示两种池化策略已经被广泛使用：（1）**Mean pooling**输出所token的平均表示，即。（2）**Max pooling**取所有令牌向量表示的第维中的最大值作为向量的第维，即其中是的第维（每个有维）。本文默认采用均值池化进行向量提取，因为它比最大池化有更好的性能[17]。通过应用这个编码器，学术文档的隐含文本语义被嵌入到中。接下来，基于三元组损失来微调。

（2）**三元组损失函数。**

三元组损失的基本思想非常简单，即使那些结构上有紧密关系的学术文档在向量空间中靠得更近，而使没有紧密关系的学术文档尽可能远离。

模型训练采用Adam优化器并按照建议的超参数值来训练模型参数[4,42]，即。请注意，超参数、控制指数衰减率，感兴趣的读者推荐阅读[42]以获取更多详细信息。通过最小化损失函数（公式3.2）从而优化模型参数。对于每篇学术文档可以通过应用公式3.1和优化后的来获得它的向量表示。

通过本文的方法对每个学术文档进行向量化，然后采用简单的主题分类任务验证文档表示的有效性。

通过单独测试三条元路径（）以及它们的所有可能组合，展示了元路径对专家检索有效性和效率的影响，如 +、+、+和++。

Top-n 专家检索：一种基于阈值算法(TA)的方法

专家检索包含了三个主要步骤：（1）为所有学术文档向量建立了一个基于邻近图（Proximity Graph，PG）的索引（PG-Index）（4.2.1小节）。这是一个离线操作，对学术文档向量数据进行一次处理构建PG-Index后则可用于在线专家检索。（2）给定一个查询文本，应用训练好的编码器得到的向量表示，用表示，随后通过PG-Index快速返回与语义最相似的篇学术文档（4.2.2小节）。（3）给定检索到的篇学术文档，提取所有出现在篇学术文档中的作者作为候选专家，并提出基于阈值算法Threshold Algorithm (TA)来有效地获得排名分数最小的Top-专家（4.2.3小节），其中排名分数的定义将用于区分不同作者相对于同一个查询文本的相关度。

PG-Index的基本组成部分实际上是一个最近邻图（NN）[49]，本文提出了一种改进策略来对其进行优化。通过以下步骤来构建PG-Index，如算法4.1所示。

|  |
| --- |
| **算法4.1 PG-Index 构建** |
| ***#*** |

* **导航点的选取**

导航节点是PG-Index的入口节点，算法从入口节点开始近似的学术文档搜索。本文将导航节点定义为所有学术文档的近似质心（是异构网络中的学术文档节点集）。更准确地说，的向量表示是与所有学术文档的质心的向量表示最接近的点，其中（第1-2行）

* **初始化近邻图**

本文应用学术文档嵌入集通过NNDescent[49]算法（一种构建NN图的常用方法）快速构建NN图。具体来说，本文首先通过评估范数距离（第4行）找到一篇学术文档的个最近的-neighbor，()，然后在原来的和之间添加一条边（第6行）。

* **调整邻居**

在原始NN图上搜索存在搜索效率低的问题[50]。给定一个查询文本，标准搜索过程可以描述为：从导航节点开始，从的个最近邻节点中选择离查询最近的学术文档节点作为下一跳节点，并贪婪地扩展搜索空间以一步一步接近。当查询目标与导航节点较远时，由于到目标点的跳数较多，所以需要更多次的搜索扩展来接近，从而增加了搜索开销。这个问题的一个解决方案是在原始NN图中为一些学术文档添加一些长距离邻居，这样就可以通过这些“高速公路”边快速接近，节省计算成本。例如，如图4.1所示，在原始NN图上将搜索空间扩展9次（红色边缘）并访问26篇学术文档。相比之下，对于PG-Index，只需要5次扩展（蓝色边缘）并访问18篇学术文档。为了实现这一点，主要通过以下两个操作调整更新NN图中每篇学术文档的邻居。

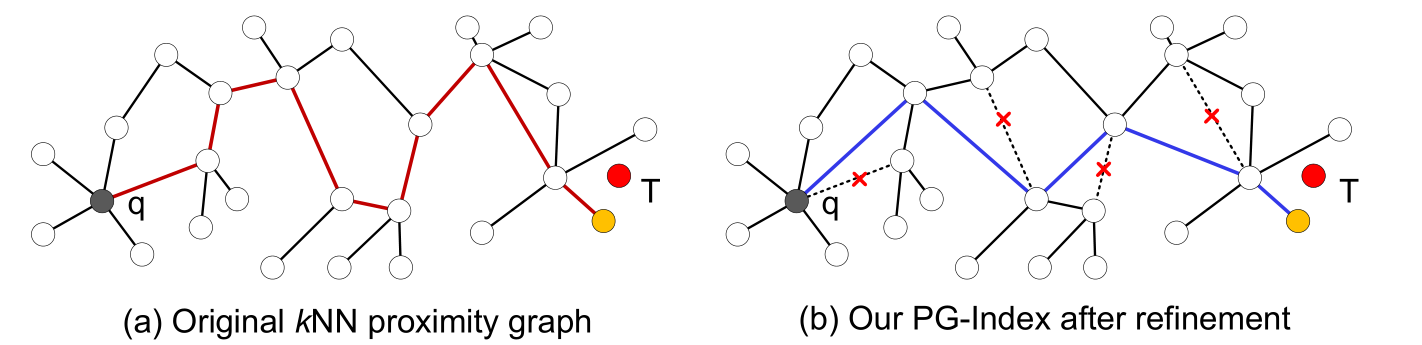


图4.1 PG-Index示意图

**扩展远距离邻居。**对于原始NN图的每篇学术文档，访问所有邻居学术文档并在和每篇学术文档之间添加一条边。通过这样做，本文构建了一些“高速公路”边，可以快速引导搜索到距离两跳的节点（第7-8行）。虽然这个操作可以提高搜索效率，但它引入了额外的索引构建和存储的开销。为了在一定程度上减少这种开销，需要在接下来的操作中去除一些冗余的邻居。

**去除冗余邻居。**给定上面扩展NN图，考虑的两个邻居和，这里可以认为是节点多余的邻居节点（是范数距离），所以可以从中移除（第9-12行）。移除背后的逻辑是，如果和之间的距离足够近，那么本文可以通过从访问来快速接近，而不是从直接到的长距离边，从而减少冗余边。换句话说，期望长距离边可以将搜索引导到远离的节点，从而很好地支持查询文本距离导航节点较远的搜索场景。

本文提出了一种基于阈值算法(TA)的方法来有效地找到前位专家，从而避免遍历和排列所有候选专家。首先，对于每篇学术文档，记录所有候选专家的分数，将以降序排列在列表中（见图4.2）。请注意，如果作者未出现在中，则其评分设置为零（。由于在中有篇学术文档，所以最终得到条排名列表。其次，从上到下访问所有列表，并更新每个访问过的作者的专家排名分数的上限和下限，分别用和表示。最后，如果找到位专家，那么将提前终止查询过程，此时其中最小的大于其他专家的最大。

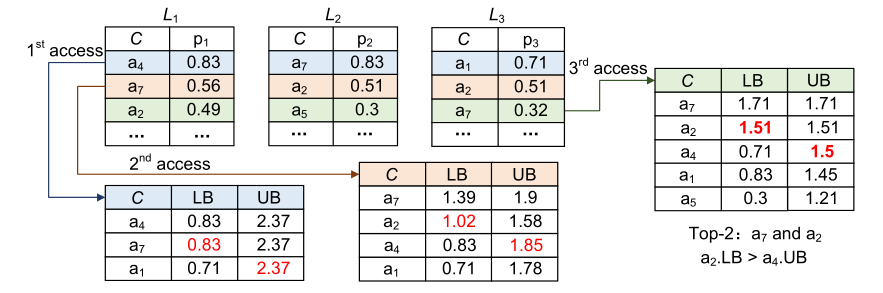


图4.2基于TA的Top-专家排名示意图

**模型训练**。模型采用编码器-解码器框架[57]，将每位专家的文本向量聚类中心作为多层深度神经网络的输入，经过多层神经网络编码组合特征得到更抽象的编码向量，并使用解码器重构了原始信息。通过梯度下降算法得到专家的特征表示，模型的网络结构如图所示。

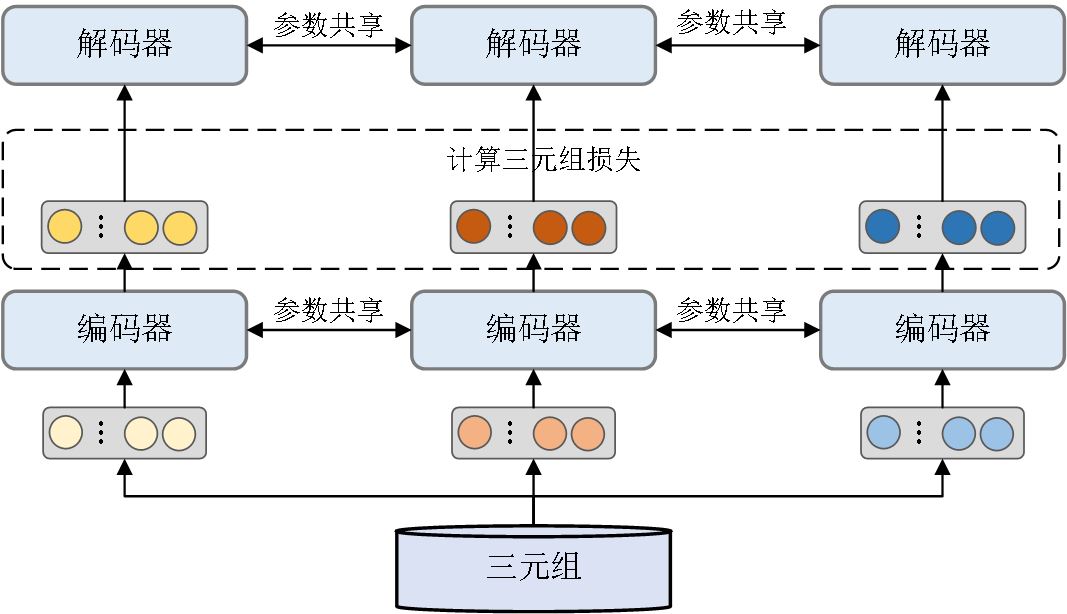


图5.3模型示意图

**问题：**

**1、**如图5.1所示，Doc1到Doc10表示某一位专家的所有学术文档，其中每一个文档一开始被视为单元素聚簇，每个合并由一条水平线表示。水平线的坐标是合并的两个聚簇的相似度，称这种相似度为合并聚簇的组合相似度。例如，Doc2和Doc3组成的聚簇的组合相似度为，合并过程是通过从底层向上移动到顶层节点逐步的合并各个聚簇。例如，Doc6和Doc7的两个文档首先合并，最后一次合并是将Doc1添加到由其他文档组成的聚簇中。层次聚类不需要预先指定聚类的数量，算法通过预先指定切割阀值进行聚类，例如切割阀值为=0.4的聚簇，可以在图5.1中看到在=0.4处切割处将产生7个聚簇，而在=0.1处切割产生3个聚簇。

|  |
| --- |
| **算法5.1 层次聚类算法** |
|  |

算法首先计算的相似度矩阵，然后该算法执行步合并当前最相似的聚簇。在每次迭代中，合并两个最相似的簇，并更新中合并簇的行和列。聚类存储为中的合并列表。指示哪些聚簇仍然可以合并。对于层次汇聚聚类算法的方法只是关于和的函数，函数度量聚簇与聚簇和的合并的相似度。常见的簇间度量的有如下几种策略[56]：

**Single Linkage**。将两个组合数据点中距离最近的两个数据点间的距离作为这两个组合数据点的距离。这种方法容易受到极端值的影响，两个很相似的组合数据点可能由于其中的某个极端的数据点距离较近而组合在一起。

**Average Linkage**。该方法是通过计算两个组合数据点中的每个数据点与其他所有数据点的距离。将所有距离的均值作为两个组合数据点间的距离，这种方法计算量比较大，但结果比前一种方法更合理。

**Ward**。该方法提出的动机是最小化每次合并时的信息损失。具体地，其对每一个聚簇定义了一个（Error Sum of Squares）量作为衡量信息损失的准则，聚簇的定义如下：

其中为聚簇中样本点的均值。衡量的是一个聚簇内的样本点的聚合程度，样本点越聚合，的值越小。Ward方法是希望找到一种合并方式，使得合并后产生的新的一系列聚簇的之和相对于合并前聚簇的之和的增长最小。

本节中默认采用Ward策略对每一位专家的学术文档进行聚类，并且将取得这些聚类中心作为专家的初始特征向量之一，最终为每位专家生成若干的向量其中表示聚簇的数量即专家向量的数量。接下来，利用这些专家的初始特征，结合上文提出的基于紧密专家社区嵌入的方法进行优化。

**2、**