基于 Vocabulary Tree 的大规模文物数字图像特征提取和分类

田磊原 ✉ 洪常凯 ✉ 蔡建楠 ✉

摘要

在文物数字化的多视图三维重建中，要求对大规模的多角度拍摄的二维数字图像进行计算，从而获取物体或场景的三维模型。随着摄影设备和图形图像技术的不断发展，数字图像的采集越来越方便，重建问题的场景规模、照片数量，以及场景复杂度也随之不断提升。在多视图三维重建流程开始之前，若能对这些大规模图像进行一个很好的分类，可以有效降低场景重建计算的复杂度并优化重建效果。

本文从大规模数字图像预处理和质量评价两个方面进行研究。首先对高分辨率数字文物图像进行压缩处理，利用 $SIFT$ 特征提取的方法对图像特征进行提取。然后根据 $SIFT$ 尺度变换金字塔，设置采样密度函数对特征进行采样。接着构建图片集词汇树、计算 $TF\text{-}IDF$ 词向量，并解决了传统方法中的过度聚类、维度间尺度不匹配等问题。最后通过小规模数据集对不同的聚类方法、不同的类内和类间评价指标进行分析和研究，确定最优的聚类方法和聚类评价指标的组合，将其应用到大规模数字文物数据集并进行测试。

关键词: SIFT, Vocabulary Tree, TF-IDF, Sample, Clustering

目录

1 绪论

1.1 研究背景

众所周知，文物是不能再生的, 也是不能永生的。尽管采取了许多保护措施，由于各种自然因素和人为因素的影响，各类文物都在遭受不同程度的老化或损坏，其特殊性对文物数字化技术提出了迫切需求。近年来，随着摄影设备和图形图像技术的不断发展，数字图像的采集越来越方便。多视图三维重建的目标是通过对多角度拍摄的二维数字图像进行计算，获取物体或场景的三维模型，并进行数字化保存。而要提高这一流程的效率，需要在计算前针对所得到的大规模二维图像进行一定的预处理分类，从而在后续的重建流程中降低复杂度。

对于目前图像匹配、分类中所用的词袋模型算法，Bag-of-Features 模型仿照文本检索领域的 Bag-of-Words 方法，把每幅图像描述为一个局部区域或关键点特征 (Patches/KeyPoints) 的无序集合。这些特征点可以作为视觉词汇，并且这些特征词汇往往是通过某种聚类算法（如 Kmeans 聚类）得到的。所有的视觉词汇形成一个字典，图像中的每个特征都将被映射到视觉词典的某个词上，统计每个视觉词的出现次数，图像可描述为一个和” 视觉词汇字典 “维数相同的直方图向量，从而构成了图像描述的 “TF-IDF 向量”。

但对于目前文物数字化要求的大规模图像数据集来说，其时间复杂度往往会变得难以接受。对此，目前比较成熟三维重建软件 colmap 则利用树的性质将时间复杂度减少到 log 级别，这也是 VocabularyTree 算法的由来。因此，本文将从词汇树的构建和应用出发，对文物大规模图像分类展开研究。

1.2 本文工作

本文主要针对大规模数据集的预分类问题进行研究。首先回顾了应用于图像特征提取的相关模型和算法，然后提出了适用于文物图像特征提取和词袋模型构建的相关方法和改进，接着研究了在词向量表示形式下的图像分类的相关方法及结果评价指标，同时还在相应的数据集上进行测试和分析，最后对于本文的工作进行总结和展望。

2 图像特征提取和聚类综述

2.1 SIFT 图像特征提取

在多视图三维重建中，需要找到图像局部间的对应关系。图像局部特征对图像局部及其邻域进行检测和描述，理想情况下，这些特征应该具有良好的可区分性，并且在光照和几何变化的情况下能够稳定出现。对于单个图像对，基于图像局部特征可以在图像对之间建立起准确的几何关系；而对于整个图像集合，利用图像局部特征可以快速找到大量图像间的关联关系。相对于语义上的分类、识别等任务，局部特征更侧重于对同一物体进行检测和描述。如果确定不同图像中的物体所属类别相同，仍需进一步检测筛选，以确定是否为同一物体 [1]。图像中的块（blob）、角点（corner），等通常与其他区域相比会有明显的差异，往往作为特征提取的主要对象，但受光照、遮挡、尺度变换等因素的影响，它们之间往往较难区分。

2004 年，David Lowe 对 SIFT（Scale-invariant feature transform, 尺度不变特征变换）[2] 的整理完善使人们发现了 SIFT 特征在图像尺度、方向变化问题中的优异表现。其结合图像尺度、邻域信息，通过尺度空间检测，搜索所有尺度上的图像位置，基于图像局部的梯度方向，在不同的尺度空间上查找关键点 (特征点)，分配给每个关键点位置一个或多个方向。所有后面的对图像数据的操作都相对于关键点的方向、尺度和位置进行变换，从而提供对于这些变换的不变性。因此，该算法具备非常强的匹配能力，能够提取到稳定的特征点，而且能够对移动、旋转、放射变换、视角变换、尺度变换和光照强弱采取很好的处理。

2.2 词袋模型

早期的图像分类主要依赖于文本特征，采用人工方式为图像标注文本，使用的是基于文本的图像分类模式。由于图像标注需要人为地辨识并为其选定关键字，故其分类的效果并不理想且耗时严重。随着计算机技术和数字化图像技术的发展，图像库的规模越来越大，人工标注的方式对图像进行分类已不可能，人们开始逐渐将研究的重点转移到基于图像内容分析的自动分类研究上。上世纪 90 年代后，出现了基于内容的图像检索技术，以图像语义特征为线索进行检索分类，上文提到的 SIFT 特征的出现和深入研究也加速了这一技术的发展。

词袋模型最初被用于文本分类中，随后逐步引入到了图像分类任务中。在文本分类中，一个文本不再考虑词序、语法和句法，而被视为一些不考虑先后顺序，不考虑依赖关系的单词集合。而在图像分类中，图像被视为是一些与位置无关的局部区域的集合，这些图像中的局部区域或关键点也就相当于文本中的单词。在不同的图像中局部区域的分布是不同的，因此可以利用提取的局部区域的分布对图像进行识别。图像分类和文本分类的不同点在于，在文本分类的词袋模型算法中，字典是已存在的，不需要通过学习获得；而在图像分类中，词袋模型算法需要通过监督或非监督的学习来获得视觉词典。

基于词袋模型的图像分类算法一般分为四步。第一步，对图像进行局部特征向量的提取，一般使用的 SIFT 特征，以保证不同变换程度下的不变性；第二步，利用上一步得到的特征向量集，抽取其中有代表性的向量（如通过聚类算法进行聚类得到的聚类中心）作为单词，形成视觉词典；第三步，对图像进行视觉单词的统计，一般判断图像的局部区域和某一单词的相似性是否超

过某一阈值，至此图像可描述为一个维数相同的直方图向量，即 Bag-of-Features。第四步，设计并训练分类器，在某种监督学习的策略下，利用图像中单词的分布进行图像分类。

2.3 Vocabulary Tree

词汇树算法 [3] 和传统的词袋模型算法相比，其主要的不同点在于视觉单词获取方式和词向量构建方式。首先，利用分层聚类的方法，通过设定分支因子 K 和层数 L 来确定每个节点子节点的数量以及树的高度，从而形成树型结构的组织方式。树的每个叶节点称为视觉词汇（这里的“词汇”是一个 128 维的 SIFT 特征向量），可以将图像提取的特征向量量化到视觉词汇中。然后，将图片的特征与每层的视觉词汇进行比对，递归向下遍历整棵词汇树以构建其词汇向量。最后，采用倒排索引文件（TF-IDF，Term Frequency Inverse Document Frequency）存储每幅图像的词汇向量，图像之间的相似度由视觉词汇向量的距离来衡量。

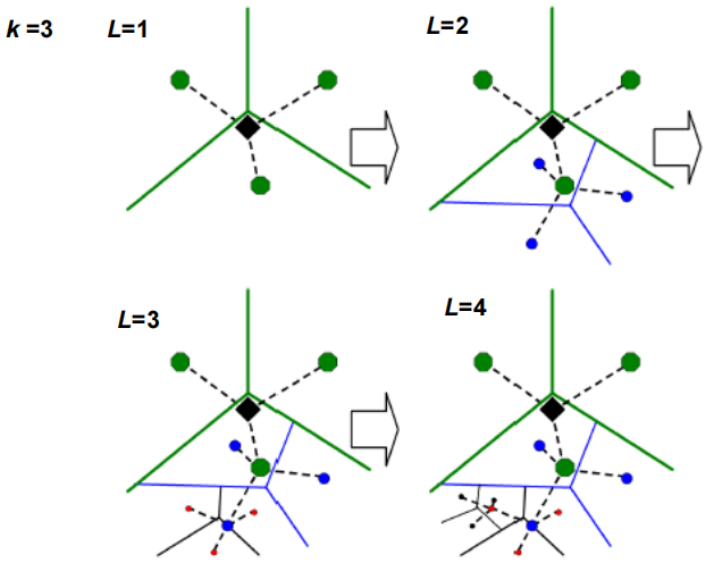

图 1: 词汇树逐层聚类

通过 Vocabulary Tree 方法，特征向量空间通过层级聚类方式被离散化成多个视觉词汇，图像间的匹配度通过图像中视觉图像的相似度进行打分，根据相似度分值排序即可实现邻近图像的筛选。其中特征量化和匹配时都无需遍历所有的视觉单词，整个查找过程是一个树形结构，极大地缩短了量化和查询检索的时间，提升了图像检索系统的性能，并且有效利用了图像局部特征，所以在多视图三维重建中被普遍应用。

3 基于 SIFT 特征的词袋模型构建

3.1 引言

在自然语言处理中，词袋模型广为使用。对于一个文本，其词袋向量的表示方法可以采用离散表示 (one-hot expression) 或分布式表示 (distribution expression)。离散表示的形式是一种极其浪费空间且计算效率差的稀疏向量，基本上不使用。而分布式表示是将高维度的离散向量通过奇异值分解等方法映射到一个低维度空间的连续向量，不仅节省空间，还可以包含更多的语义信息。而词袋模型同样可以用于对图片的表示，对于词向量的计算和处理等方法都可以参考自然语言处理中的思想和方法。

3.2 本章工作

3.2.1 图像特征提取

仿照自然语言处理中的词袋模型构建，我们需要提取每张图片的所有向量，作为“单词字典”。而一张图像可以有多种特征，如颜色、纹理、形状、位置关系等。图像特征的表示和提取也有很多种方法。本文中选用了 SIFT(Scale-invariant feature transform) 尺度不变特征，该方法对于物体的尺度变化，刚体变换，光照强度和遮挡都具有较好的稳定性，可在图像中检测出关键特征点。

3.2.2 图像选择和压缩

在文物三维重建中需要处理的是大规模的图像输入，在构建词汇树的过程中如果将所有的图片数据都用于构建词汇树，会带来巨大的时间和空间开销。因此在构建词汇树时，从图像数据库中抽取一部分图片用来建树，用于所有图像的词向量提取。

同时数字文物图像通常由专门的图像设备进行信息采集，图像分辨率非常高，直接进行特征提取会非常耗时，因此可以先对图像进行压缩处理，降低分辨率，在损失很少信息的情况下较大提高特征提取速度。

3.2.3 图像特征采样

本文主要处理的是数字文物图像，通常由专门的图像设备进行信息采集，图像分辨率很高，可以提取出很多特征点。但是在后续的计算中，如果使用所有的特征点信息不仅会花费大量时间，还会因为特征点数目过多而使得特征“失真”，即携带的图像信息丢失，给词汇树构建和图像聚类引入大量误差。

因此对于一张图像，我们只需要提取其部分的特征信息用于后续的处理。但考虑到所有的特征点按照金字塔形状以尺度进行分布，而且覆盖范围越大的特征点数目会越少，同时也会处在金字塔的越上层。因此为了保证不丢失各个尺度上的特征点，同时还要满足不同尺度上的特征点的分布，我们设置一定的采样密度函数对于一张图像的所有特征点进行采样。

3.2.4 词汇树构建

对照自然语言处理的词袋模型，同样我们不能使用一张图片在所有图片的所有特征向量下的 one-hot 向量来对图片进行表示，因此我们还需要对“单词字典”进行降维处理。容易想到我们可以将相似的原始特征点当作同一个图像特征进行处理，因此需要对于图像特征进行聚类。但是常见的 KMeans 等聚类方法需要预先规定好聚类的数目，因此我们构建一个树形结构，逐层对于图像特征聚类。最后的有效特征都处在树的叶子节点。

3.2.5 词袋模型构建

获得了一棵词汇树之后，我们将所有的叶子节点作为有效特征表示。为了获得一张图片的词向量表示，首先要计算每张图片中的特征向量在每个叶子节点上出现的次数。我们需要将一张图片的所有或者部分特征向量放入词汇树，逐层向下和每一层的多个聚类中心进行比较并选择出最邻近的聚类中心，直到比较到叶子节点为止。但是这只是原始的词向量的表示，为了获得 TF-IDF 表示，我们还需要获取每个叶子结点的权重对于获得的词向量进行修正。

3.3 SIFT 特征提取

SIFT 特征具有尺度不变性，可在图像中检测出关键点，是一种局部特征描述子。SIFT 算法的实质是在不同的尺度空间上查找关键点 (特征点)，并计算出关键点的方向。SIFT 所查找到的关键点是一些十分突出，不会因光照，仿射变换和噪音等因素而变化的点，如角点、边缘点、暗区的亮点及亮区的暗点等。

3.3.1 SIFT 特征提取步骤

SIFT 算法可以分解为如下四步：

（1）尺度空间极值检测：搜索所有尺度上的图像位置。通过高斯差分函数来识别潜在的对于尺度和旋转不变的关键点。

（2）关键点定位：在每个候选的位置上, 通过一个拟合精细的模型来确定位置和尺度。关键点的选择依据于它们的稳定程度。

（3）关键点方向确定：基于图像局部的梯度方向, 分配给每个关键点位置一个或多个方向。所有后面的对图像数据的操作都相对于关键点的方向、尺度和位置进行变换, 从而保证了对于这些变换的不变性。

（4）关键点描述：在每个关键点周围的邻域内, 在选定的尺度上测量图像局部的梯度。这些梯度作为关键点的描述符, 它允许比较大的局部形状的变形或光照变化。

3.3.2 尺度空间极值检测

在不同的尺度空间是不能使用相同的窗口检测极值点，对小的关键点使用小的窗口，对大的关键点使用大的窗口，为了达到上述目的，我们使用尺度空间滤波器。高斯核是唯一可以产生

多尺度空间的核函数。一个图像的尺度空间 $L(x,y,\sigma)$，定义为原始图像 $I(x,y)$ 与一个可变尺度的 2 维高斯函数 $G(x,y,\sigma)$ 卷积运算，即：

$$L(x,y,\sigma) = G(x,y,\sigma) * I(x,y) \tag{1}$$

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2}$$

σ 是尺度空间因子，它决定了图像的模糊的程度。在大尺度下（σ 值大）表现的是图像的概貌信息，在小尺度下（σ 值小）表现的是图像的细节信息。

下面需要构建图像的高斯金字塔，它采用高斯函数对图像进行模糊以及降采样处理得到的，高斯金字塔构建过程中，首先将图像扩大一倍，在扩大的图像的基础之上构建高斯金字塔，然后对该尺寸下图像进行高斯模糊，几幅模糊之后的图像集合构成了一个 Octave，然后对该 Octave 下选择一幅图像进行下采样，长和宽分别缩短一倍，图像面积变为原来四分之一。这幅图像就是下一个 Octave 的初始图像，在初始图像的基础上完成属于这个 Octave 的高斯模糊处理，以此类推完成整个算法所需要的所有八度构建，由此构建了一个高斯金字塔。

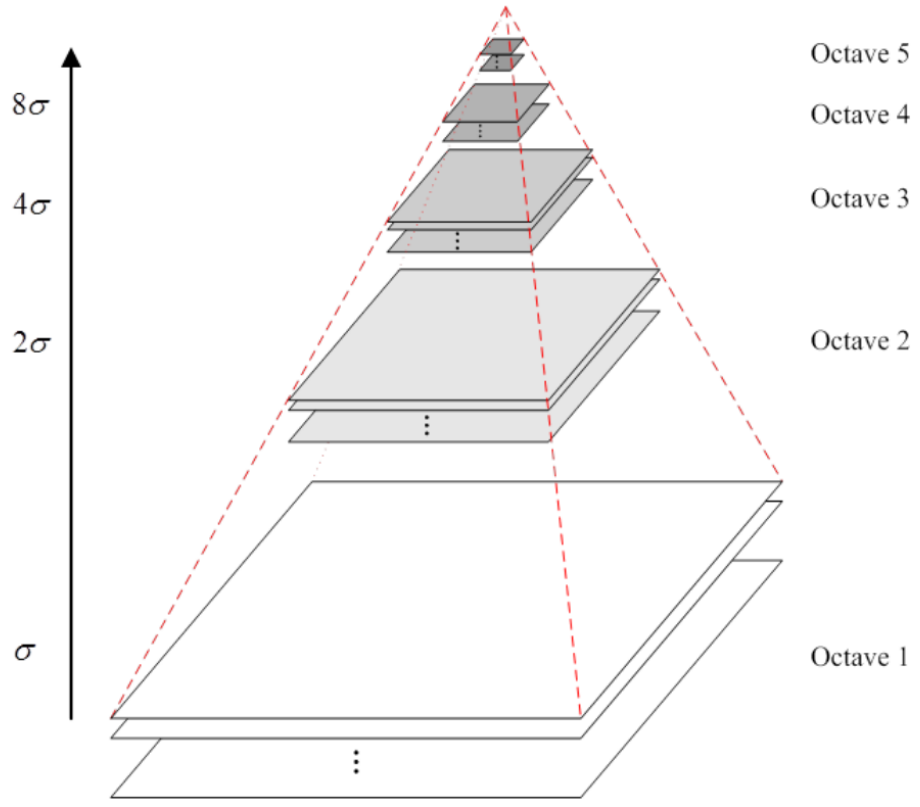

图 2: SIFT 特征提取金字塔

利用 LoG(高斯拉普拉斯方法)，即图像的二阶导数，可以在不同的尺度下检测图像的关键点信息，从而确定图像的特征点。但 LoG 的计算量大，效率低。所以我们通过两个相邻高斯尺度空间的图像的相减，得到 DoG(高斯差分) 来近似 LoG。

为了计算 DoG 我们构建高斯差分金字塔，该金字塔是在上述的高斯金字塔的基础上构建而成的，建立过程是：在高斯金字塔中每个 Octave 中相邻两层相减就构成了高斯差分金字塔。

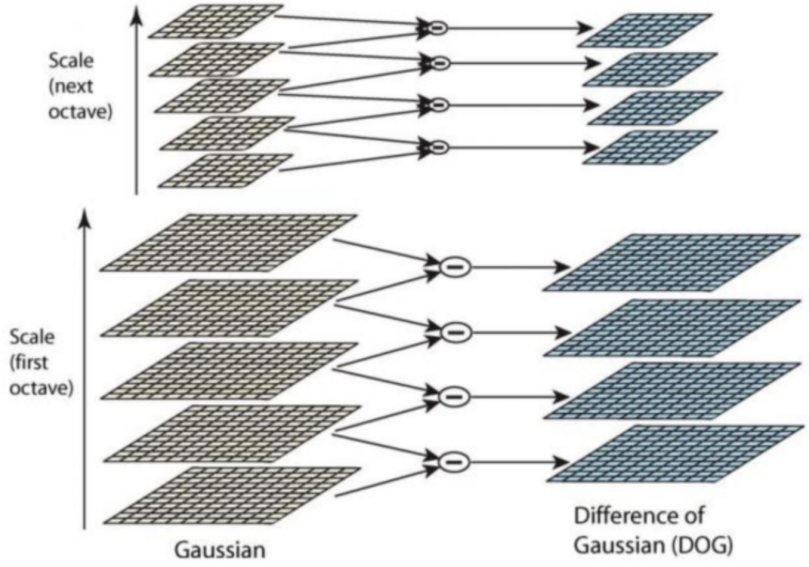

图 3: SIFT 尺度变换

高斯差分金字塔的第 1 组第 1 层是由高斯金字塔的第 1 组第 2 层减第 1 组第 1 层得到的。以此类推，逐组逐层生成每一个差分图像，所有差分图像构成差分金字塔。概括为 DOG 金字塔的第 o 组第 n 层图像是有高斯金字塔的第 o 组第 n+1 层减第 o 组第 n 层得到的。后续 Sift 特征点的提取都是在 DOG 金字塔上进行的。

3.4 图像特征采样

在现有的 SIFT 函数中，我们可以设置参数，确定提取特征的最大数量，但是这种方法下只能获取金字塔顶端的特征点，即尺度小的模糊特征点。为了保证可以采集到不同尺度下的特征点，我们需要提取出所有尺度下的图像特征点，之后按照密度函数进行采样。而为了满足不同需求，我们可以设置不同的采样密度函数。

3.4.1 正向密度采样

如果需要突出大尺度下精细的特征点的识别作用，我们需要根据尺度进行正向密度采样。由于尺度分布满足 $1:\sqrt{2}:2\ldots\ldots$ 等比例分布，我们设置采样密度函数 $y=C(\sqrt{2})^x$

3.4.2 反向密度采样

如果需要保证不同尺度下的点都可以被均匀采集到，由于分布密度函数随着尺度正向增加，我们需要进行反向密度采样。为了保证所有尺度下的采集到的特征点数目相当，我们将概率密度函数设置为分布密度函数的倒数，即 $y=C(\sqrt{2})^{-x}$

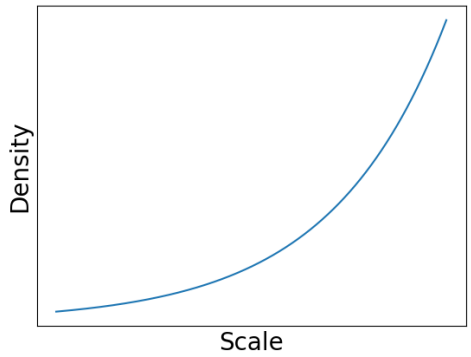

图 4: 正向密度采样

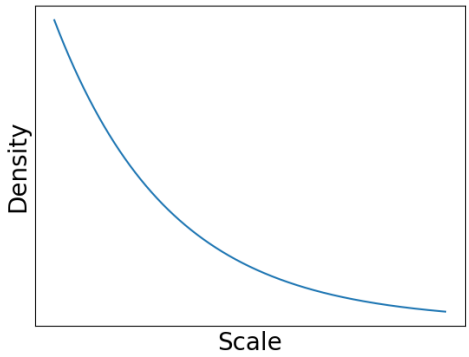

图 5: 反向密度采样

3.4.3 有偏正态分布采样

在 SIFT 特征提取中，较小尺度变换下的特征点往往覆盖范围更大，SIFT 对其的评分会更高，如果需要保证较小尺度下的特征点可以尽可能多的被采集到，我们设置带偏正态分布采样。此时的正态分布中心不设置在尺度范围的最中心，而是在尺度范围中心偏左的位置。

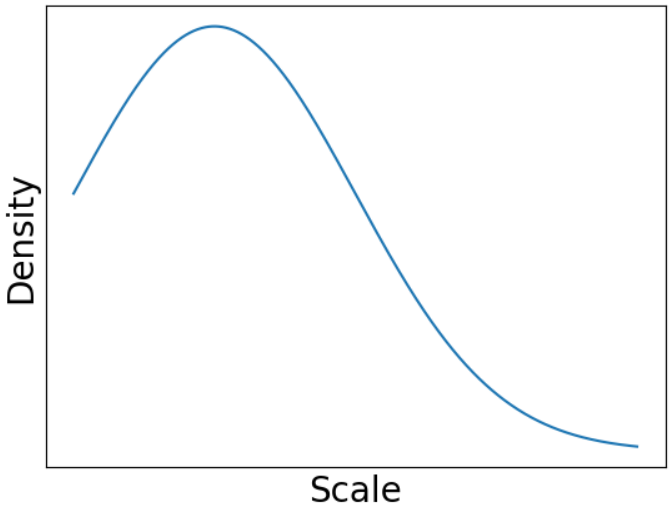

图 6: 有偏正态分布采样

经过后续实验发现，上述三种采样密度函数中效果最好的为反向密度采样，由于保证了不同尺度下的特征点数目相当，那么对于整个图片数据库来说会有更加均匀的聚类效果。

3.5 词汇树构建

目前已经获得了所有的图像特征，那么需要构建树形结构对其进行逐层聚类。首先需要确定超参数：最大分支数目和最大深度。最大分支数目决定了每一层的聚类数目，最大深度决定了递归构建词汇树的结束条件。上面的方法虽然理论上可行，但是不能直接使用。如果在实际应用中直接使用，会出现很多的问题，下面针对两个对结果影响较大的问题进行分析。

3.5.1 过度聚类

如果只规定了最大分支数目和最大深度，那么在最后一层聚类的过程中，可能会存在过度聚类的情况。由于最后一层的节点包含的特征点总数可能和最大分支数目几乎接近，这就导致最终的每一个叶子节点中只包含很少的特征点，甚至会出现一个叶子节点中只有一个或两个特征点的情况，此时的词向量维数和 one-hot 向量相比没有明显的下降，效果很差。

为了避免过度聚类，我们设置了最小节点尺寸，该超参数保证了每一个叶子节点中需要包含的最少的特征点数目，保证适度聚类。在引入该超参数之后，在逐层聚类的过程中，每一层聚类的数目不再是固定的，而是动态的。每一层的聚类数目的选择都需要满足如下两个条件：不超过最大分支数目、保证平均子节点大小满足最小尺寸要求。

3.5.2 维度间尺度不匹配

同时考虑到所有的特征向量在不同维度上的分布情况可能不同，如果直接使用距离函数进行度量和聚类，可能会出现因为不同维度尺度不同带来的错误聚类。因此在聚类之前首先对于所有的特征向量在对应的维度上进行归一化。每个向量在每一个维度上有：

$$V = (v_1, v_2, ..., v_n), \qquad v_i = \frac{v_i - \overline{v_i}}{\sigma_i} \tag{3}$$

其中 $\overline{v_i}$ 是第 i 个维度上的均值，σ_i 是第 i 个维度上的标准差。经过该处理之后，所有维度上的数据都符合标准正态分布，消除了原有的尺度差异。

3.5.3 修正算法

考虑了上述出现的两种问题之后，将归一化和动态确定分支数目的思想应用到词汇树当中。整个算法的实现可以参考如下的伪代码。

Algorithm: Dynamic Culstering and Constructing the Vocabulary Tree

Input : The N vectors :
Output: The $Vocabulary\ Tree$ (with cluster centers stored in nodes)
Hyper-parameters: B(Max Branches) D(Max Depth) MS(Min Cluster Size)
Process:
Standardize the data on each dimension
for each node(with n vectors) on each level **do**
  Determine $n\_cluster = min(B, \frac{n}{MC})$
  Cluster these n vectors by KMeans method
  Terminate when $n \leq MC\ or\ depth = D$
end

表 1: The pseudocode for constructing Vocabulary Tree

3.6 词袋模型构建

构建词汇树之后，我们可以通过词袋模型构建所有图片的词向量数据库。首先需要对每张图片进行特征提取和采样。如果在构建词向量的过程中使用图片的所有特征点，会极大提高时间和空间开销，因此明智的做法是进行采样。为了保证采样得到的数据点可以代表一张图片的所有特征点，因此需要选择和建树过程一致的反向采样函数。

得到一张图片的采样后的特征点后，通过词汇树逐层向下寻找最邻近聚类中心，直到寻找到叶子节点，统计一张图片对每一个叶子节点的访问顺序，即可得到一张图片的初始词向量表示。但同时需要考虑到不同叶子节点的权重，因此还需要计算每个叶子节点的权重并对词向量进行修正。$TF\text{-}IDF$ 词向量的计算方法如下：

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \tag{4}$$

$$idf_i = \log \frac{|D|}{|\{j : c_i \in d_j\}|} \tag{5}$$

$$TF\text{-}IDF = TF \times IDF \tag{6}$$

$TF(Term\ Frequency)$ 是词频，表示某一“特征”在某一个图片中出现的频率，这个数字通常会被归一化。其中 $tf_{i,j}$ 表示特征中心 c_i 图片 d_j 中的出现频率，$n_{i,j}$ 表示特征中心 c_i 图片 d_j 中的出现次数，$\sum_k n_{k,j}$ 表示图片 d_j 中的总特征数。

$IDF(Inverse\ Document\ Frequency)$ 是逆向文件频率，也即后文中所言叶子节点的权重。可以由总图片数目除以包含该“特征”的图片的数目得到，一般为了方便表示还会再将得到的商取对数。如果包含某一“特征”的图片越少，IDF 越大，则说明该“特征”具有很好的类别区分能力。其中，$|D|$ 是数据库中的图片总数，$|\{j : t_i \in d_j\}|$ 表示”特征“c_i 的图片数目 (即 $n_{i,j} \neq 0$ 的图片数目) 。

$TF\text{-}IDF$ 综合考虑了一个“特征”在一张图片中的出现频率和每一个“特征”的区分能力 [4]，可以较好衡量一张图片在“特征”空间中位置信息，可以将其用于后续的查询和搜索。

3.7 图片搜索测试

3.7.1 测试方法和步骤

为了判断构建的词汇树和计算得到的图片词向量是否有效，我们进行图片搜索测试。首先根据图片数据库构建出词汇树和词向量，然后输入需要查找的图片，并根据词向量获取数据库中与之最相似的五张图片。输入一张新的图片后，首先需要获取其词向量，方法同上。提取并采样特征之后，根据词汇树和权重向量得到对应的 $TF\text{-}IDF$F 向量。之后在整个数据库中通过距离函数进行查找，选择最优的五个结果即可。

3.7.2 测试数据集

为了进行上述的测试，从网络上下载到了带有类别信息的数据集并进行测试。数据集的基本信息如下。

图片总数	图片类数	平均每类数目
502	190	2.64

表 2: 图像搜索测试 - 数据集

3.7.3 测试结果及分析

由于图片数量适中，测试时将全部图片用于构建 Vocabulary Tree，之后将所有图片都在数据库中进行搜索，利用先验的标签信息对于测试结果进行分析，计算每张图片的搜索结果的覆盖率（返回正确图片数量 / 本类图片总数量）得到如下数据。

测试图片数量	平均搜索正确数目	平均搜索覆盖正确率	最大搜索覆盖率
502	1.89	65%	100%

表 3: 图像搜索测试 - 测试结果

结果发现搜索结果的覆盖正确率并没有很高，经过分析，原因在于类数过多，每一类的图片数量较少。在这种情况下进行聚类时，每一类图像的特征点很少，在聚类中被稀疏化，所以不能较为有效的搜索，而包含图片数目较多的类别搜索的覆盖率会更高，更能反映词汇树的聚类效果。

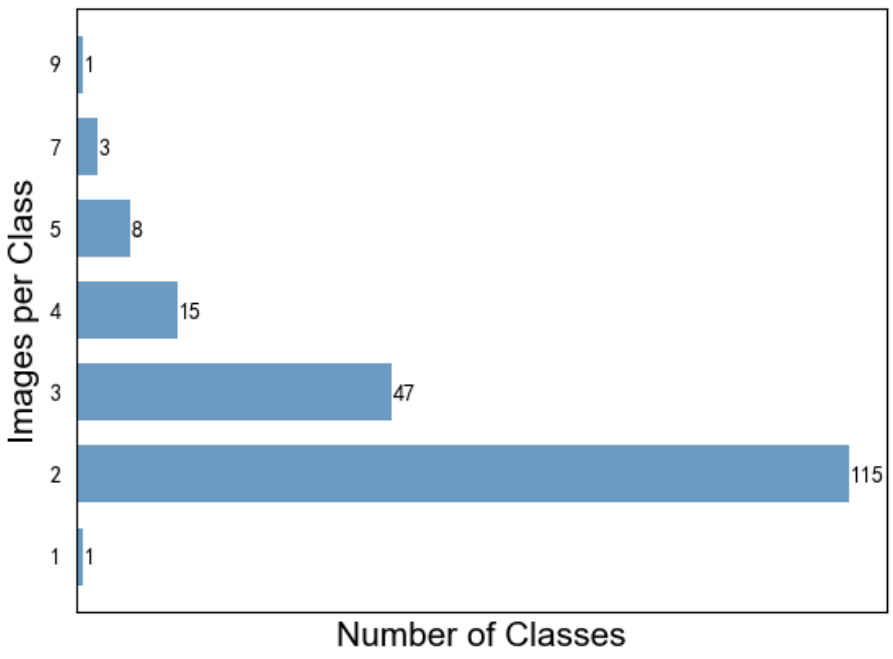

图 7: 图像搜索测试 - 类别尺寸计数

对于尺寸较大的图片类别进行搜索测试，发现其效果很好，多组测试中的正确搜索覆盖率可高达 100%。由此我们认为 Vocabulary 构建有效，词向量计算正确。

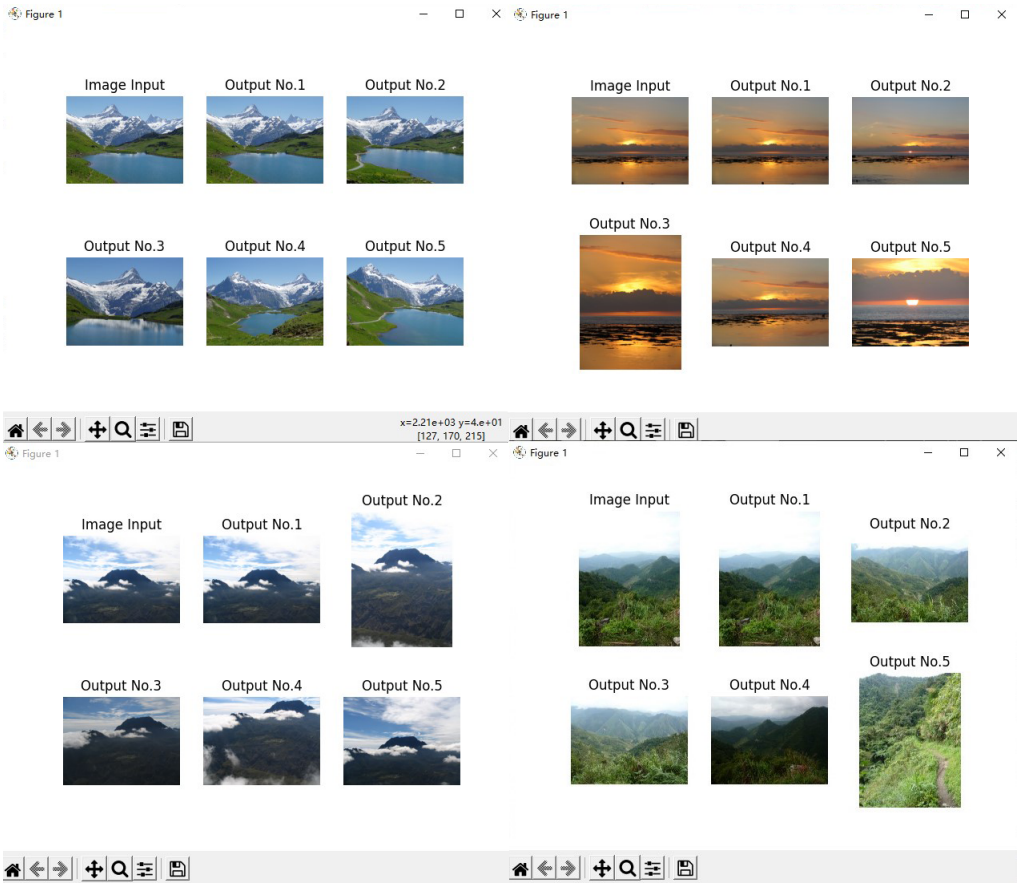

图 8: 图像搜索测试 - 大尺寸类别搜索结果

3.8 本章小结

通过本章节的系列操作之后，可以有效进行图像特征的提取和采样、Vocabulary 的构建和词向量的计算，在数据集上的搜索测试也有着较好的结果。同时本章节还进行了相关的创新性工作。首先，考虑到了传统的 Vocabulary Tree 只能在固定分支数目下进行聚类的不足，消除了词汇树构建过程中的过度聚类问题。其次，预先对于特征向量进行归一化处理可以有效解决维度间尺度不匹配问题。最后，本章节还提出了很多的优化时间空间开销的方法，如在图像特征提取时进行特征点的概率密度采样、在构建词汇树前对于整个数据集上的图片进行压缩和筛选。至此，图像分类的前期准备工作已经成功完成。

4 聚类方法和评价

4.1 引言

由于本文中处理的图像数据都不带有标签，并且没有训练和使用模型来进行类别区分，因此本文中的分类是纯粹的无监督聚类，可以采用朴素的无监督聚类的方法。常见的无监督聚类方法有 K-Means 聚类和层次聚类，前者需要事先确定聚类数目，后者对于分类数目的确定没有强制要求。为了研究和比较两种聚类方法，本文中需要为两种聚类方法动态确定最佳的聚类数目，同时用聚类结果评价指标来衡量聚类的效果。

4.2 聚类方法选择

4.2.1 最优聚类数目选择

为了确定最优的聚类数目，需要首先确定一系列的聚类数目，在不同的聚类数目下分别聚类，并评价其聚类的效果，最终选定一个最好的聚类数目。聚类评价指标会在下一小节中阐述，除了使用常用的若干个指标外，还可以对不同的指标进行组合使用。

4.2.2 两对象间距离函数选择

在均值聚类的过程中，需要根据距离来选择最邻近的聚类中心。而常用的距离函数包括：欧氏距离 (Euclidean Distance)、曼哈顿距离 (Manhattan Distance)、余弦距离 (Cosine Distance)、海明距离 (Hamming Distance) 等。

Euclidean Distance 表示最简单的空间距离，使用简单，但是容易受到不同维度尺度不一致的影响。因此在使用此距离度量之前，需要对数据进行标准化。Manhattan Distance 计算的是各个维度下的距离之和，对于高维度的向量来说没有明显的数值含义，并且也会受到不同维度尺度不同的影响。Cosine Distance 通过不同向量在空间中的方向信息来判断其相似性，只考虑方向，不考虑距离。Hamming Distance 通常用于离散的分布，而词向量被不同特征中心的权重加权后，形成了连续空间上的分布，因此也不能使用该方法衡量。综合考虑来看，欧氏距离和余弦距离相对更加适合不同词向量之间的距离衡量，下面对二者进行更加详细的比较和分析。

首先，我们很容易得到欧氏距离和余弦距离的联系，即数据归一化处理之后，欧氏距离和余弦距离存在负相关关系，二者等价。下面进行简单的证明。假设空间中存在两个点 $A(x_1, y_1), B(x_2, y_2)$，$cos(\theta)$ 表示其余弦相似度，$euc(x, y)$ 表示其欧氏距离，A' 和 B' 分别表示 A 和 B 归一化后的坐标。那么归一化的坐标和两个距离函数计算如下。

$$\begin{cases} A' = (\frac{x_1}{\sqrt{x_1^2+y_1^2}}, \frac{y_1}{\sqrt{x_1^2+y_1^2}}) \\ \\ B' = (\frac{x_2}{\sqrt{x_2^2+y_2^2}}, \frac{y_2}{\sqrt{x_2^2+y_2^2}}) \end{cases} \tag{7}$$

$$\begin{cases} cos(\theta) = \frac{x_1}{\sqrt{x_1^2+y_1^2}} \times \frac{x_2}{\sqrt{x_2^2+y_2^2}} + \frac{y_1}{\sqrt{x_1^2+y_1^2}} \times \frac{y_2}{\sqrt{x_2^2+y_2^2}} \\ euc(x,y) = \sqrt{(\frac{x_1}{\sqrt{x_1^2+y_1^2}} - \frac{x_2}{\sqrt{x_2^2+y_2^2}})^2 + (\frac{y_1}{\sqrt{x_1^2+y_1^2}} - \frac{y_2}{\sqrt{x_2^2+y_2^2}})^2} \\ euc(x,y) = \sqrt{2 - 2 \times cos(\theta)} \end{cases} \tag{8}$$

由此可见余弦距离避免了不同维度间尺度的差异。当一对图片的特征点数目差距很大、但内容相近时，如果使用词频或词向量作为特征，它们在特征空间中的的欧氏距离通常很大；而如果使用余弦相似度的话，它们之间的夹角可能很小，同时保证了较高的相似度。

在图像领域中，研究对象的特征维度往往很高，而欧氏距离的数值则受维度的影响，范围不固定，并且含义也比较模糊。因此选择余弦距离有利于词向量的精细对比，并且可以排除维度数目、维度间尺度差异等因素的影响。

4.2.3 两类间距离计算方法选择

当采用层次聚类的方法时，一个关键问题是衡量两个类的之间的距离，而类间距离的计算有三种方式：Max Distance(两个簇的样本对之间距离的最大值)、Min Distance(两个簇的样本对之间距离的最小值)、Average Distance(两个簇的样本对之间距离的平均值)。在实际应用中，三种方法的效果差异并不明显，一般可以同时使用三种方法并确定最优者。

$$\begin{cases} d_{\min}(C_i, C_j) = \min_{\vec{x}_i \in C_i, \vec{x}_j \in C_j} \operatorname{distance}(\vec{x}_i, \vec{x}_j) \\ d_{\max}(C_i, C_j) = \max_{\vec{x}_i \in C_i, \vec{x}_j \in C_j} \operatorname{distance}(\vec{x}_i, \vec{x}_j) \\ d_{\text{avg}}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{\vec{x}_i \in C_i} \sum_{x_j \in C_j} \operatorname{distance}(\vec{x}_i, \vec{x}_j) \end{cases} \tag{9}$$

4.3 类内评价方法

聚类的目的是一个类内的点尽可能分布密集，不同类之间的距离要尽可能大。针对上述两个目标，衍生出了多种类内和类间的聚类评价方法。本小节简述三种常用的类内评价方法。

4.3.1 Silhouette Coefficient

轮廓系数 (Silhouette Coefficient) 最早由 Peter J. Rousseeuw 在 1986 年提出。它结合了内聚度和分离度两种因素来衡量聚类的结果。为了计算某一个聚类结果的轮廓系数，首先需要计算一个样本的轮廓系数。对于某样本 i，记它到同类其他样本的平均距离 a_i，a_i 越小说明样本 i 和同类之间的其它样本的差异越小。接着选择一个样本到其它类中心的距离的最小值 b_i 来衡量样本 i 和其它类的不相似程度，即 $b_i = min(b_{i1}, b_{i2}, ..., b_{ik})$。$b_i$ 越大说明样本 i 和其它类之间的差异越大。接着就可以计算某一个样本 i 的轮廓系数 s_i。

$$s(i)=\frac{b(i)-a(i)}{\max\{a(i),b(i)\}}=\begin{cases}1-\frac{a(i)}{b(i)}, & a(i)<b(i)\\ 0, & a(i)=b(i)\\ \frac{b(i)}{a(i)}-1, & a(i)>b(i)\end{cases} \tag{10}$$

如果 s_i 接近 1，则说明样本 i 聚类合理；如果 s_i 接近-1，则说明样本 i 聚类失当；若 s_i 近似为 0，说明样本 i 在两个类的边界上。

得到每一个样本的轮廓系数之后，可以使用所有样本的轮廓系数均值代表整个聚类结果的轮廓系数。聚类结果的轮廓系数 S 的取值在 $[-1,1]$ 之间，S 值越大，说明同类样本相距约近，不同样本相距越远，则聚类效果越好。虽然轮廓系数通常被认作类内指标，但是它也充分考虑了类间的因素，是一个比较综合的指标。

4.3.2 Calinski Harabasz Score

CH 分数（Calinski Harabasz Score) 也称之为 Calinski-Harabaz-Index，通过类内方差和类间方差来衡量聚类效果。该方法同时考虑了类内和类间的分布情况，衡量了实际分类情况和理想分类情况（类之间方差最大，类内方差最小）之间的区别。

$$\begin{cases}s(k)=\frac{\mathrm{Tr}(B_k)}{\mathrm{Tr}(W_k)}\times\frac{N-k}{k-1}\\ W_k=\sum_{q=1}^{k}\sum_{x\in C_q}(x-c_q)(x-c_q)^T\\ B_k=\sum_q n_q(c_q-c)(c_q-c_E)^T\end{cases} \tag{11}$$

其中，k 是聚类的数目，N 是样本总数，W_k 表示类内散度矩阵，B_k 表示类间散度矩阵，C_q 表示类 q 中所有数据的集合，c_q 表示类 q 的中心，c_E 表示所有数据的中心点，n_q 表示类 q 数据点的总数。为了校正样本总数的影响，计算式中还加入了归一化因子 $(N-k)/(k-1)$。

一般来说 CH 分数越高，实际分类情况和理想分类情况之间的差异越小，聚类效果越好。但是受到归一化因子 $(N-k)/(k-1)$ 随着类别数 k 的增加而减少的影响，该方法更偏向于选择类别少的分类结果，在后续的实验应用中也证实了这一点。在后续的多组测试中，最优的 CH 分数总是在 $k=2$ 时出现，因此可以选择局部最优点代替全局最优点作为最佳聚类数目。

4.3.3 Davies Bouldin Index

戴维森堡丁指数 (DBI) 是由 David · L · Davies 和 Donald · Bouldin 提出的一种评估聚类算法优劣的指标。在朴素的 KMeans 聚类算法中只衡量了类内距离最小化，忽略了类间距离。而该算法将类内距离和类间距离结合起来，用类间距离对类内距离进行修正，避免 t 停留在局部最优解的情况。同样，为了计算出聚类结果 DBI 指数，需要先计算出每一个类的 DBI 指数，之后用所有类的 DBI 指数的均值作为整个聚类结果 DBI 的衡量。

$$\begin{cases} \bar{R} = \frac{1}{N}\sum_{i=1}^{N} R_i \\ R_{ij} = \frac{S_i + S_j}{M_{ij}} \\ S_i = \left(\frac{1}{T_i}\sum_{j=1}^{T_i} |X_j - A_i|^p\right)^{1/p} \\ M_{i,j} = \|A_i - A_j\|_p = (\sum_{k=1}^{n} |a_{k,i} - a_{k,j}|^p)^{1/p} \end{cases} \tag{12}$$

$\bar{R}$ 表示整个聚类的结果的 DBI 指数，R_{ij} 用于衡量第 i 类与第 j 类的相似度。S_i 计算的是类内数据到类质心的平均距离，代表了类 i 中各样本的分散程度。M_{ij} 计算的是类 i 与类 j 质心的距离，衡量了两个类之间的分散程度。一般而言，$\bar{R}$ 值越小，分类效果越好。

4.4 类间评价方法

在类间方法中，需要已知样本点的类别信息和一些外部基准。这些基准包含了一些预先分类好的数据，比如由人工基于某些特定场景预先生成一些带 label 的数据。这些基准可以看成是准确值，这样就可以在衡量聚类结果时比较实验值和真实值之间的差异。

4.4.1 Purity

纯度 (Purity) 是一种简单而透明的评估手段，在计算纯度时并不需要知道每个簇所对应的真实类别，而是把每个簇中最多的类作为这个簇所代表的类。然后计算正确分配的类的数量在总样本中的比例，该比例即为整个样本的纯度。

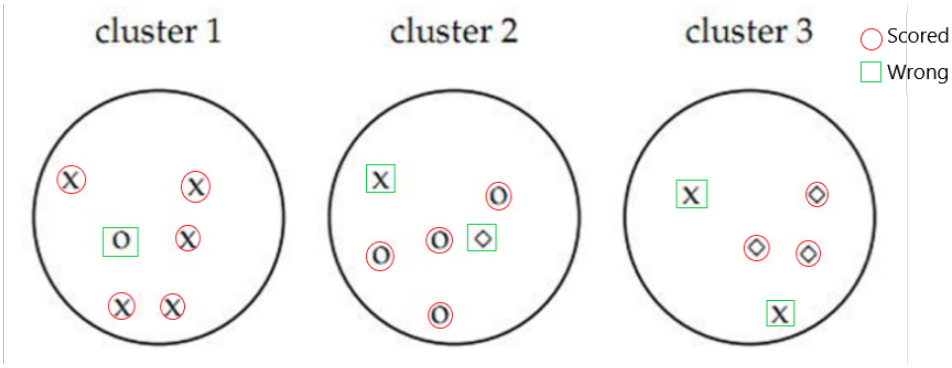

图 9: 纯度计算方法

$$P = (\Omega, C) = \frac{1}{N}\sum_{k} max|\omega_k \cap c_j| \tag{13}$$

其中 N 表示样本总数，$\Omega = \{\omega_1, \omega_2, ..., \omega_k\}$ 表示聚类后的簇，$C = \{c_1, c_2, ..., c_J\}$ 表示正确的类别，ω_k 表示聚类后第 k 个簇中的所有样本，c_j 表示第 j 个类别中的真实样本。P 的取值范围为 $[0, 1]$，一般来说 P 越大表示聚类效果越好。

纯度的计算虽然简单，但是无法用于决策聚类质量和簇个数之间的关系，如果有多个簇混杂在一起，但只有很少的个体在其它簇中，得到纯度值也会很高，特别是如果所有个体均聚类到一起，那么其纯度会是 1，显然这是不准确的。

4.4.2 Gini Coefficient

基尼系数表示在样本集合中一个随机选中的样本被分错的概率。Gini 指数越小表示集合中被选中的样本被聚类错误的概率越小，集合的纯度越高。

$$G(p) = \sum_{k=1}^{K} p_k(1 - p_k) = 1 - \sum_{k=1}^{K} p_k^2 \tag{14}$$

其中，p_k 表示样本属于第 k 个样本的概率。相比于信息熵的计算 $E(D) = -\sum_{k=1}^{K} p_k log_2 P_k$，基尼系数不需要进行对数计算，更加简便，但是其具有和 $Purity$ 同样的问题。

4.4.3 Rand Index

兰德指数 (Rand Index, RI) 将聚类看成是一系列的决策过程，存在接受正确结果、拒绝正确结果等一系列情况，从分好类的图片集取出两张图片，存在四种不同的情况。

TP	将两张相似的图片归为同一类
TN	将两张不相似的图片归为不同类
FP	将两张不相似的图片归为一类
FN	将两张相似的图片归为不同类

表 4: 聚类结果抽样的四种情况

$$RI = \frac{TP + TN}{TP + TN + FP + FN} \tag{15}$$

RI 取值范围为 $[0, 1]$，RI 越大意味着聚类结果与真实情况越吻合，聚类的效果越高。

4.4.4 F-Measure

F-Measure 是于对于 Rand Index 的修正，通过对聚类结果的随机抽样，可以获得相应的准确率 (Precision) 和召回率 (Recall)。

$$\begin{cases} Precision = \frac{TP}{TP+FP} \\ \\ Recall = \frac{TN}{TP+FN} \end{cases} \tag{16}$$

在传统的 Rand Index 计算中，准确率和召回率地位等同。但是如果期望在两者之间更偏向于某一方，可以对于其中的某一方进行加权修正。用 β 作为比例系数，控制 $Recall$ 在最终结果中的占比。同样，该指标和聚类效果正相关。

$$F_\beta = (1+\beta^2)\frac{Precision \times Recall}{\beta^2 Precision + Recall} \tag{17}$$

4.5 图片聚类测试

4.5.1 数据集

由于本文处理的图像的最终目的是用于三维重建，因此本小节的测试采用的是三维重建的相关数据集 (DTU Robot Image Data Sets, 丹麦技术大学 (Technical University of Denmark) 提供的结构光扫面数据)。由于数据集比较庞大，采用其中的部分用于聚类测试。

Total images	534
Clusters	12
Details	[Auditorium 40, Statue 60, Museum 55, Cup 29, Doll 19+49, Beer 49, Boxing Gloves, 49 Vegetables 49, Sculpture 49, Construction Model 49, Train 37]

表 5: 图像聚类测试 - 数据集

4.5.2 测试结果

为了测试前文中提到的不同聚类方法和评价指标，这里对于不同的超参数的设置和评价方法的选择进行系列测试。其中在聚类过程中采用不同的 $Evaluation\ Method$ 来进行最优聚类数目的选择。而聚类结束之后，利用先验信息，通过 $Rand\ Index$ 对于聚类的真实结果进行

Cluster Method	Evaluation Method	Linkage	n_Cluster	Score	Rand Index
KMeans	CH	—	8	8.955914799	0.887457751
KMeans	SH	—	12	0.082237822	0.970973438
KMeans	DBI	—	11	4.375355706	0.973185488
AC	CH	single	9	1.028264222	0.163318366
AC	CH	average	8	8.479624902	0.935879869
AC	CH	complete	8	8.630459726	0.94393265
AC	SH	single	9	-0.015864898	0.163318366
AC	SH	average	10	0.091275817	0.978624281
AC	SH	complete	13	0.094238962	0.991019668
AC	DBI	single	14	2.051917651	0.176838052
AC	DBI	average	14	3.992570376	0.974035739
AC	DBI	complete	13	4.315017036	0.991019668

表 6: 图像聚类测试 - 测试结果

4.5.3 结果分析

首先，通过实验结果发现获取的最优聚类数目并不能完全衡量聚类的好坏，在实验中存在聚类数目等于真实值，但是分类的结果存在混乱、评价函数评分较低的情况。

其次，如前文所言，$CH\ Score$ 在实际应用中效果不理想，当使用该指标寻找全局最优解时会出现偏差，容易出现最优解始终为两类的情况。

最后，层次聚类整体上比 KMeans 聚类效果更好；CH 在实际应用中效果差于 DBI 和 SH；在层次聚类的过程中，类之间距离的度量采用 $Min\ Distance$ 的度量方法会有较大偏差。

4.6 工程实现

在完成了图像特征提取、词汇树构建、词向量计算、图像聚类之后，可以将该方法应用到文物数字图像上。由此考虑，本文作者还将算法移植到了大型服务器上并针对大规模的数字文物图像数据集进行测试。

4.6.1 工程环境

Linux	20.04
Python	3.9.1
opencv-python	4.5.5.62
opencv-contrib-python	4.5.5.62

表 7: 大型服务器环境配置

4.6.2 工程流程

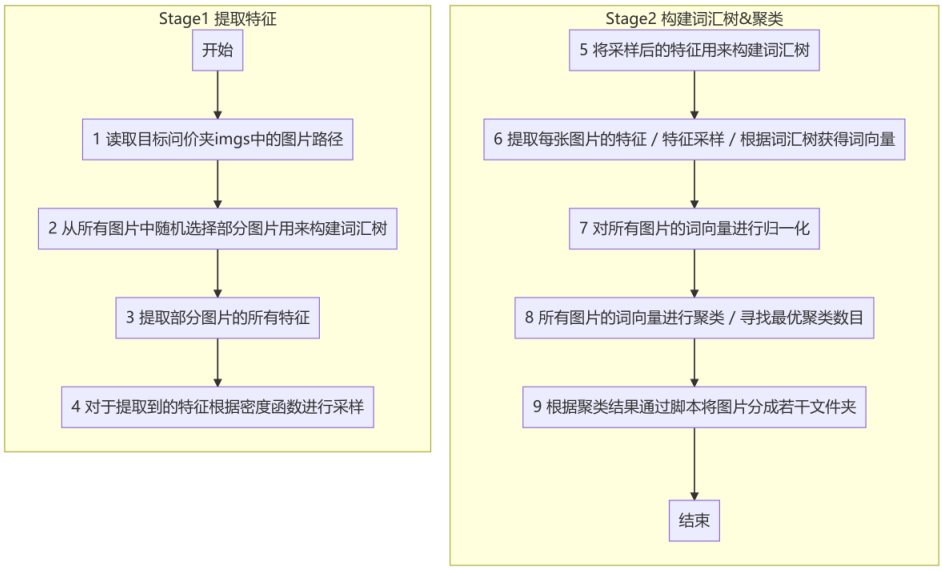

图 10: 工程流程图

4.6.3 大规模文物图像测试

工程搭建完毕之后，通过运行命令可以对指定路径下的图片进行分类。本次大规模测试选取的是浙江大学艺术考古图象数据实验室所采集的须弥山圆光寺第 41 窟数字图像。

图片总数	1296
有效图片	1295
损毁图片	1
图片类数	由于整体空间环境较小，石窟整体特征较为接近，肉眼判别为 4-5 类

表 8: 大规模文物图像测试 - 数据集

对该大规模数据集进行聚类之后，分为 5 类，具体结果如下 (展示其中两个子类的部分图像):

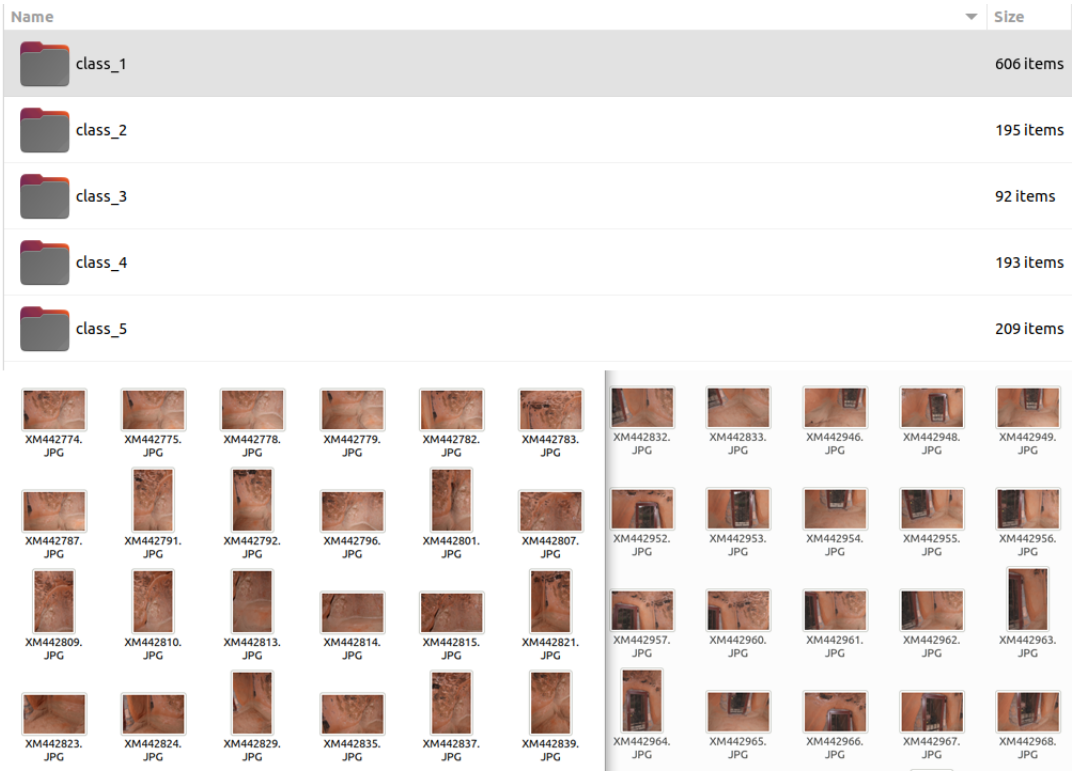

图 11: 大规模文物图像测试 - 测试结果

4.7 本章小结

在本章节中，首先探究了不同的聚类方法和聚类评价指标，其中着重对于不同的聚类评价指标进行了原理推导和优缺点分析。然后通过 *DTU Robot Image* 数据集对于这些方法和指标进行验证，选择一个最优的聚类方法和评价标准以达到最优的聚类效果。最后将确定好的聚类方法和评价方法应用到大规模的文物图像数据集上并进行聚类测试和分析。至此，对于大规模数字文物图像的预处理已经基本完成，本工程的输出可以直接定向到三维重建的输入部分。

5 总结与展望

5.1 本文工作总结

文物三维重建复杂度的增加对多视图三维建模带来了诸多挑战，本文主要聚焦于二维图像计算处理前的匹配度分类预处理，从聚类方法和评价指标两个方面展开研究，利用词汇树的树形结构降低了传统词袋模型算法的时间复杂度。主要工作可以概括如下：

(1) 完成了对文物图片压缩后的特征抽样，根据 SIFT 特征提取的高斯差分金字塔提出了正向、反向密度采样与有偏正态分布采样，以期突出大尺度下精细特征点、保证不同尺度下特征点的均匀采集或突出较小尺度下的特征点的作用。

(2) 完成了普通小规模三维重建图像数据集和大规模数字文物图像集的词汇树构建，以词汇树的树形结构减少了构建图片词向量的时间复杂度。通过动态确定分枝数目与归一化分别结局了过度聚类和维度间尺度不匹配的问题。同时，构建的词汇树可进行保存与导入，在聚类分类时不需重新构建，具有可复用性和可拓展性。

(3) 首先根据词向量的特性确定了余弦距离作为两个对象间的距离计算方法。而后在无监督模式下分别比较了 $KMeans$ 和层次聚类两种聚类方式，Silhouette Coefficient、Calinski Harabasz Score、Davies Bouldin Index 三种类内评价方法。最后以类间评价方法 Rand Index 找到了较为优异的聚类方式，并将其应用到石窟文物图像上，得到了不错的分类效果。

5.2 未来工作展望

本文工作主要在无监督模式下开展，实际上文物三维重建的场景变换并不大，各图片之间关联性较强，类间区分度并不明显。因此仅以普通的聚类方法并不能做到更为严格的场景分类，未来期望可以融入有监督学习方式，并结合多种类内、类间评价指标以选出更优的分类簇数，进行更为精细化的场景分类。

另外，SIFT 特征提取具有尺度变换不变性，会在特征提取的过程中隐去方向和位置等信息，而方向和位置信息在三维重建中具有重要的作用，两张具有相似特征的图像可能分属于空间的不同位置，在实际的三维重建中可能并不能归属到同一个类别当中。因此我们希望结合图片的三维信息例如空间位置等，引入更多其它性质对图像进行区分，并将工程输入定向到三维重建的输入以优化评价指标的呈现。

最后，基于深度学习的方法在图像特征提取、图像分类中展现出巨大的潜力，通过对大量数据进行训练可以学习到相应的先验信息，而深度神经网络的分布式的训练方法为模型的训练提供了更高的效率，我们希望能在大规模数字文物维图像处理中进一步结合这些方法。

参考文献

[1] 刘卓昊. 大规模多视图三维重建计算与质量评价方法研究. PhD thesis, 浙江大学, 2020.

[2] David G. Lowe. Distinctive image features from scale-invariant keypoints. 60(2), 2004.

[3] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2161–2168, 2006.

[4] Thomas Roelleke and Jun Wang. Tf-idf uncovered: A study of theories and probabilities. New York, NY, USA, 2008. Association for Computing Machinery.