TemplateVocabulary需要两个模板类参数

TDescriptor:用来存储特征描述子的类,例如可以是cv::Mat类型

F:用来操纵特征描述子的类,例如计算多个特征描述子的均值啦、计算两个特征描述子的距离啊。例如聚类时,就需要用到。

TemplateVocabulary类并不直接操作特征描述子TDescriptor,而是通过类F中的函数来操作特征描述子,这样子的实现方便用户进行扩展,因为用户知道特征描述子是怎么存的,以及如何操作特征描述子。

Start TemplateVocabulary类对传进来的特征点vector<vector<TDescriptor>>构建Bag-of-Words。 vector<vector<TDescriptor>>存储了若干张图像特征点的特征描述子,而vector<TDescriptor>存储的是一张图像的特征 点的特征描述子,因此TDescriptor存储是一个特征点的特征描述子。 getFeatures getFeatures函数就是将vector<vector<TDescriptor>>转换成vector<TDescriptor>,因为构建Bag-of-Words时并不需 要区分特征描述子是来自哪张图像的,所以就把特征描述子都放到了一个vector中,方便之后的处理。 将特征描述子聚成K类,根据特征描述子到聚类中心的距离,将特征描述子划分到相应的类中。这样就得到了当前结点的K个孩子结点。 **HKmeansStep** 每个类都是树的一个结点,对每个结点中的特征描述子继续执行上面的步骤进行聚类、划分,直到树达到规定的深度L。 每个叶子结点都是一个单词,对每个叶子结点进行编号。 createWords 每个单词记录其对应的叶子结点的指针,这样做是方便找到第i个单词对应的叶子结点是哪个结点。例如在计算单词的权重时就会用到。 计算每个叶子结点的权重,需要注意的是,如果使用的是TF-IDF权重,那么该函数只计算了叶子结点的IDF权重。 则TF则是用户调用transform函数将图像的特征描述子转换成BoW向量时才会使用,因为只有知道了要转换的图像的特征描述子(视觉单词),才 知道TF词频。 setNodeWeights 至于第i个单词结点IDF的计算,IDF i = log(N/N i), N表示图像文档数目,自然就是vector<vector<TDescriptor>>的大小;而N i表示第i个单词 在多少个图像文档中图像,这个只需要统计每个图像文档中有没有出现第1个单词即可。 end

TemplateVocabulary的transform函数会将用户传进来的图像的特征描述子,将图像转换成BoW向量表示。

由于一般图像只有几百到几千个特征描述子,而BoW词汇树的叶子结点有几百万到几千万个,所以图像的BoW向量是稀疏向量。

因此DBoW2使用STL中的map容器来存储这个向量。