# 检索综述

检索不能等同于图像检索。图像检索是有计算机视觉的，而一般所说的检索是数据库检索，两者在问题规模和问题定义上相去甚远。

## 常用算法

1）倒排索引（inverted index）

将特征向量聚类成codewords，相当于把浮点向量离散化到一维或者多维的格子里，每个格子对应一个codewords，查找时离散化到格子里再更精细地查找。

2）哈希（hash code）

哈希成二进制比特串，之后用multi-index的数据结构来做汉明空间内的最近邻搜索，或者直接穷举地查找。"水了大波论文，不过搜索精度不够，只能先用哈希粗糙地定位最相似的一千个，然后再用精确的特征距离来排序。

1. 倒排索引和哈希的结合

我再列一些我最近看的比较多的。locality sensitive hashing，ITQ，product quantization，inverted multi-index，笛卡尔k-means，hamming embedding。可以到ccf会议推荐那里的，数据库和信息检索类中的信息检索，图形学与多媒体类也有图片检索，人工智能类的计算机视觉。

## 实际应用

这种数据结构或者离散化方法对于提高检索效率特别有用，谷歌微软这些大公司的检索引擎很用的上的。

## 难点和要点

大多数可用的索引结构都对应了一下的几点权衡：

搜索时间

搜索质量

每个索引向量使用的内存大小

训练时间

无监督训练对外部数据的需求

# 相似性搜索

相似性搜索中存在的问题也是搜索复杂对象时的固有问题。复杂对象会导致大多数技术对大范围集合的抓取能力等问题。而在相似性搜索时，大部分情况下对象都是复杂的。

主要有两种：1）最邻近搜索（nearest neighbor search）；2）范围查询（range queries）

## 工作原理

相似性搜索工具可用于识别哪些候选要素与要匹配的一个或多个输入要素最相似（或最相异）。

相似性的基础是数值属性（感兴趣属性）的指定列表。

如果指定了一个以上的要匹配的输入要素，相似性将基于每个感兴趣属性的平均值。输出要素类（输出要素）将包含要匹配的输入要素以及找到的所有匹配的候选要素，这些要素以相似程度排序（由最相似或最不相似参数指定）。返回的匹配数基于结果数参数的值。

## 实际应用

人力资源经理可能想要证明其公司薪资水平的合理性。找出在城市规模、生活成本和便利设施方面相似的城市后，她便可以查看这些城市的薪资水平，从而确定它们是否与本公司的薪资水平一致。

犯罪分析师希望搜索数据库以查看某罪行是否属于较重犯罪形式或有重罪趋势。

课外健身计划在 A 城极其成功。计划提倡者期望找到与其计划推广的候选城市具有相似特征的其他城市。

执法机构用此方法揭露毒品种植地或生产地。标识具有相似特征的地方可能有助于制定未来的搜索目标。

大型零售商不仅拥有数个成功店铺，也有少数业绩不佳的店铺。找到一些具有相似人口特征和环境特征（交通便利性、知名度以及商业互补性等等）的地方有助于标识新店的最佳位置。

# 哈希检索

在哈希算法中，通常的目标是将样本表示成一串固定长度的二值编码(通常使用0/1或-1/+1表示其中的每个bit)，使得相似的样本具有相似的二值码，使用汉明距离（Hamming distance）计算两者相似度。

## 发展历程

在最初的工作中，作者提出在特征空间中随机选择一些超平面对空间进行划分，根据样本点落在超平面的哪一侧来决定每个bit的取值。这类方法虽然有严格的理论证明保证其效果，但是在实际操作中通常需要比较多的bit才能得到令人满意的检索效果。

在之后的工作中，为了得到编码长度更短、检索效果更好的二值码，人们进行了很多尝试，包括构建不同的目标函数、采用不同的优化方法、利用图像的标签信息、使用非线性模型等。

## 具体使用流程

与常见的机器学习算法不同，哈希方法的目标是得到二值编码，所以优化过程中经常会遇到离散取值的约束，因此通常来说无法使用基于梯度的方法对目标函数进行优化。

为了简化问题，通常的做法是改用一个更宽松的约束，比如不再要求“二值码”是二值的，而是只要在一个规定的范围中即可。优化结束后，再对松弛过的“二值码”进行量化，得到最终的真二值码，深度哈希算法通常也采用这种做法。"

## 应用实例

### 跨模态检索的例子

在搜索引擎中输入一些关键词，找相关的图像。通常来说，关键词(文本)和图像并不在同一个空间中，因此无法直接比较。

在2016年年初，李武军老师带领的研究团队在arXiv上发布了一篇文章，其中介绍了一种跨模态深度哈希算法DCMH(Deep Cross-Modal Hashing)。这篇文章中，作者利用一个两路的深度模型将两种不同模态的数据(文章中是文本和图像)变换到一个公共空间，并要求相似的样本在这个公共空间中相互靠近，如下图所示。通过同时对图像和图像、图像和文本、文本和文本这几种不同类型的样本对施加这个约束，可以保证两种模态样本的对齐。如此一来，即可实现在公共空间中的跨模态检索。

### 以文本搜文本

以文本搜文本，即给定一段查询文本，找到和这段文本相似的其他文本。

作为一个经典问题，在IJCAI 2015上，来自中科院自动化研究所的许家铭博士等人提出了一种基于卷积网络的解法

。该方法首先将文本中的每个单词表示成一个词向量，将文本转化为一个宽度等于句子长度，高度等于1，通道数等于词向量维度的张量。之后通过对文本进行卷积、pooling等一系列操作，得到一组中间表示，并对这组中间表示进行一系列操作得到二值码。这个方法中词向量的提取并不能和最终的任务连在一起，因此不能算是完全的端到端模型。作为利用CNN进行文本哈希算法的初步探索工作，这篇文章为深度哈希算法的更广阔应用开启了一扇新的大门。

# 图像检索

检索不能等同于图像检索。图像检索是有计算机视觉的，而一般所说的检索是数据库检索，两者在问题规模和问题定义上相去甚远。

在图像检索中，其基本问题是如何度量图像间的相关度，这可分解为图像表征学习（Representation learning）和距离测度学习（Distance metric learning）。

直观地讲，为提高相关性度量质量，我们可以优化图像标注学习，也可以优化距离测度学习。

然而，与其他视觉任务不同，图像检索面对的数据库规模大，对检索相应时间苛刻，因此一般采用简单的距离测度，比如L1距离或L2距离，这样方便通过施加稀疏性约束来引入倒排索引结构。所以，在很多图像检索方法中，相对于距离测度学习，大家一般更关注在图像表征学习上。

## 一般流程

1）提取特征。包括传统的SIFT。

2）聚合特征（aggregate）。包括BoW，VLAD，FV（fisher vector）

3）降维、量化

大规模图像检索综述传统方法是图片检索的的大体框架大致可以分成两步

1. 抽取特征：深度学习是特征提取的手段。最后的全连接层或者 globel pooling 负责分类，因此一般的做法是直接取前几层卷积的输出。

2）计算相似度：常使用欧式距离/汉明距离/Triplet等方法。

主要会遇到两个问题：首先一个是数据精度问题，因为直接取特征输出多是浮点数，且维度高，这会导致储存这些图像的特征值会耗费大量空间，第二个因为纬度高，所以用欧式距离这种方式计算相似度，可能会触发维度灾难，令使用欧式距离代表相似度这种方法失效，一般会在loss上做文章，例如使用Triplet-loss。

## 算法

主要算法是通过哈希(Hash)或量化(quantization)降低内存消耗。

目前，在近似最近邻搜索(ANN, Approximate Nearest Neighbors)方面，主要存在的方法有：

1）树结构索引方式，但一般都不推荐使用KD树方法。最经典的代表就是KD树了，小白菜用过的KD树的场合有：SIFT匹配、BoW模型。

2）PQ量化以及以PQ量化进行演变的改进方法。对向量进行切分，单独做聚类进行编码。

3）局部敏感哈希(LSH)是高维向量近似查找最常用的算法，代码实现很多，可在github上直接找到。但是LSH毕竟是基于概率的模型，会有一定的False Positive和False Negative，要实现一个准确且速度快（哈希函数越多速度变慢但是准确率上升）的LSH并不是很容易，目前检索TOP会议上还一直在出现对LSH做改进的工作。

哈希方法，这是业界公认的最佳方案。目前很多深度哈希做图像检索的论文几乎都是在网络中实现近似二值编码把Deep特征哈希了，加索引。

## 评价指标

大多数算法都是各自按各自的评价指标来。

常用的包括基于top 20的mAP

## 应用实例

### 以图搜图

以图搜图（query by example）， 即提供一幅查询图像，检索与该图相似的图像。本质上就是比较图像相似度。

类似淘宝搜图这类大规模图像识别，用phash和dhash(速度比phash更快，但准确性差点)。

### 大规模人脸检索算法

对于大规模高维向量数据集的检索任务，查找性能优化是核心问题。高维向量的检索性能优化通常分两种方式，实际应用时，常常结合这两种方式进行检索优化。

1. 查找优化，比如建立倒排索引，这种方式通过优化检索结构进行性能优化，不改变向量本身，这个主要还是看场景。例如，根据应用场景还可以加上出现频率进行优化，用图片特征加索引加重排序方法。
2. 向量优化，通过将高维浮点向量映射为低维向量，或者映射到汉明空间，以此减少距离计算复杂度。例如，特征提取的优化、降维、量化。

这其实是商业秘密了，大规模人脸检索做的好，相当值钱。

主要的困难是：

1.工业界数据量比学术界最大的数据集还要大至少2个数量级；

2.人脸总的ID数也会有至少2-3个数量级的提高。

百万级的人脸搜索主要有

dictionary learning methods

K-SVD

基于PCA的变种

使用聚类算法：基于动态聚类结构/树性结构的类中心；

使用编码算法

embedding的量化和降维(128已经很少了)

如果有更多人脸属性就更好了。比如确定是男是女就可以直接去对应set中去找

比较的肯定是两个embedding

可以参考video retrieval领域，特别是perdition retrieval(行人检测+行人重识别)。

### facenet项目

对整个集合减均值的操作，这能提高精度。注意！！！减均值提升精度只能用于cos距离，欧氏距离中无效。

if subtract\_mean:

mean = np.mean(np.concatenate([embeddings1[train\_set], embeddings2[train\_set]]), axis=0)

# 得到的mean是512维的向量

按照我的想法，这是对类间进行归一化的方式，使类间的距离变大。

# 参考资料

https://blog.csdn.net/lala\_01/article/details/52623957