**多媒体基础大作业2**

信息005 王靳朝 2206113602

**结合视觉&文本分析技术，给出：**

1. **度量文档相似性的几种方法。**
2. **度量两幅图像相似性的几种方法。**
3. **如何对文本与图像进行快速的检索，请给出详细的实现过程。**

1.在衡量文本相似性时，核心思想时度量对于已经将文本转换后的数值向量之间的距离。因此相似度的计算方法存在两个重要组件，即文本表示模型和相似度的度量方法。前者是将文本表示为计算机可以计算的数值向量，也就是提供特征。后者是基于前面得到的数值向量进行计算。

文本表示模型中主要是文本切分粒度和特征构建的方法。粒度切分有n-gram、分词、句法分析、主题模型等方法。特征构建的方法主要有TF和TF-IDF，也就是概率分布直方图。由此得到的距离度量方式主要有以下几种。

欧式距离计算方法，也就是计算两个数值向量在欧氏空间中的位置，即两向量相减后的模值。

余弦相似度分析，来源于余弦定理。

曼哈顿距离，与欧式距离相似，将平方换为了绝对值并去掉根号。

直方图交。二者相交与二者相并做除法。

同时还存在基于LDA的文档描述。它使用了词袋的方法。将文档视为次品向量，从而得到易于建模的数字信息。它的核心公式为：**p(w|d)=p(w|t)\*p(t|d)。**

2.衡量图片相似性的方法有：

SSIM算法。由亮度对比、对比度对比、结构对比三部分构成。分别采用平均灰度作为亮度测量，灰度标准差作为对比度测量以及结构测量，并设定相关的对比函数。SSIM函数具有对称性、有界性和最大值唯一性。

均值Hash算法。该方法将图片进行哈希转化，生成一组二进制数字，然后通过比较不同照片的Hash值距离得到相似性的度量。

差异Hash算法。由于均值Hash算法对于均值较为敏感，因此改进在对图片进行Hash转化时，通过对比左右两个像素，得到最终的Hash序列。

感知Hash方法。利用DCT将图片从时域转化为频域，并且DCT矩阵从左上到右下代表越来越高的频率系数，只保留左上角的低频区域，代表缓变也就是较高相似度。

余弦相似度。与文本分析类似，两个向量之间的余弦值越接近1，代表二者越相似，反之越不相似。

直方图法。用来衡量两张图片在颜色分布上的相似度。通过统计不同图像中不同颜色的像素数量并通过直方图的形式展现，进而比较图片的相似度。

互信息的方法。通过衡量两个变量之间的信息共享程度，计算联合概率分布和各自的概率分布评估相似性。

特征匹配。使用特征提取算法例如SIFT、SURF等提取特征点并对特征点计算其特征描述子，之后对特征点进行相似度量，一般利用欧式距离等，并根据匹配的特征点对计算相似度指标。

还有基于深度学习的衡量方法例如Siamese网络、SimGNN等。

3.文本的快速检索主要是基于倒排索引实现的

倒排索引是首先知道了每个关键词都出现在了哪些文档里，从关键词搜文档（关键词→文档）。倒排索引先记录了每个关键词出现在了哪些文档里，需要哪个关键词，把含有的文档直接找出即可。

当需要查询文档时，在字典中定位关键词，并返回其倒排记录表。要查询同时出现某个关键词的文档时，分别对关键词做上述操作，对二者的倒排记录表求交集。在建立倒排索引时，首先手机需要建立索引的文档，并将文档转化为一个个词条的列表，之后进行语言学的处理，最后对所有出现的词项建立倒排索引，索引中包含一部词典和全体倒排记录表。

在实际应用之中，往往存在一下问题：我们需要模糊匹配。例如输入UK可以返回United Kindom的文档这种缩写匹配，再例如词根的匹配等等。

图片的快速检索是基于SIFT特征实现的。算法流程为：尺度空间的极值检测，这一步通过高斯函数的微分来识别对尺度和方向不变的潜在关注点。之后进行关键点定位，拟合模型来确定位置和比例。之后进行方向分配：根据局部图像梯度方向为每个关键点位置分配一个或多个方向。最后生成关键点描述符，在每个关键点周围的区域中，以选定的比例测量局部图像梯度。

**描述深度网络中的卷积、Pooling、全连接以及激活操作，其中的Pooling和卷积可以用5\*5或者3\*3大小的模板进行示例。**

以3\*3的卷积核描述卷积的原理操作。卷积的原理实际上是对输入块内的元素进行加权求和，权值即为卷积核的数值。假设输入为以下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |

而卷积核选择对角线为1，其他为0：

|  |  |  |
| --- | --- | --- |
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

卷积的操作为，卷积核作为权值，从左上开始，按照从左到右，从上到下的顺序依次和输入结合，输入按照卷积核的权值加权求和后作为当前输出位置的结果。当卷积核作用到左上方的3\*3区域时，表现为以下二者相乘相加：

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 |  |  |  | 1 | 0 | 1 |  |  |  | 1\*1 | 1\*0 | 1\*1 |
| 0 | 1 | 1 |  | \* |  | 0 | 1 | 0 |  | = |  | 0\*0 | 1\*1 | 0\*0 |
| 0 | 0 | 1 |  |  |  | 1 | 0 | 1 |  |  |  | 0\*1 | 0\*0 | 1\*1 |

求和加权并求和之后结果为4.

结束之后卷积核右移一位，对下一3\*3区域做相同的操作。当一行移动完毕后，卷积核向下移动一行，并从最左侧开始重新加权相乘相加，直至输入全部被执行以上操作。

最后的结果为：

|  |  |  |
| --- | --- | --- |
| 4 | 3 | 4 |
| 2 | 4 | 3 |
| 2 | 3 | 4 |

Pooling操作的原理为对于每个块内的数值取最大或取平均，作为该块处理后的输出，以3\*3的最大池化为例，假设输入为6\*6区域：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 10 | 1 | 7 | 1 | 2 | 3 |
| 6 | 5 | 0 | 2 | 7 | 9 |
| 4 | 5 | 9 | 4 | 1 | 5 |
| 4 | 1 | 1 | 1 | 8 | 4 |
| 0 | 6 | 8 | 0 | 2 | 1 |
| 7 | 2 | 3 | 4 | 5 | 3 |

对于每个3\*3的池化区域取最大值，得到的输出结果为

|  |  |
| --- | --- |
| 10 | 9 |
| 8 | 8 |

全连接的原理将特征块作为其权值，对于全连接层的每个块进行整体加权求和。全连接把每一个区域变为一个值作为输出，可以理解为将全部特征整合到一起（高度提纯特征），方便交给最后的分类器或者回归。

由于在此前以上所有运算均是线性运算，即均为乘加运算，对于更加复杂的非线性问题是无法做出较为精准的操作的。因此激活函数是作为一种非线性函数加入到深度网络之中，他的目的是帮助深度网络更好的解决非线性问题，即获得更加复杂的学习模式。在经过卷积、池化全连接等操作后，需要加入激活函数给定对于某个节点来说给定输入或输入集合下的输出结果，由此会引入一个非线性关系，帮助深度网络更好地学习。