中山大学

硕士学位论文

依赖于查询的排序学习算法研究

姓名：罗烨敏

申请学位级别：硕士

专业：计算机软件与理论

指导教师：汤庸

20090524

依赖于查询的排序函数学习算法研究

专业：计算机软件与理论

硕士生：罗烨敏

指导教师：汤庸 教授

摘 要

排序是信息检索的核心问题，因为一个搜索系统的好坏主要由它的排序结果的正确率来评价。信息检索中提出了许多排序模型。直到最近，被称为“排序学习”的机器学习技术被深入应用于这个任务。在过去的大部分工作中，一个单一的排序模型被用来处理所有的查询。这可能是不太恰当的，因为不同的查询之间往往有比较大的差异。

本文首先论述对于不同的查询，采用不同的排序模型的必要性。然后提出一个通用的依赖于查询的排序学习框架。该框架是基于查询聚类的。该框架将训练查询表示成查询特征空间的一个点。在排序的时候，给定一个测试查询，框架找出与它接近的训练查询，构建一个依赖于测试查询的训练集，然后用这个训练集训练一个模型，最后使用这个模型对与测试查询关联的文档进行排序。这个框架对于改善排序的精度确实是有帮助的，因为它利用了相似查询的有用信息，同时避免了不相似查询的负面影响。

当进行查询聚类的时候，使用的查询特征以及查询相似性的计算，对于聚类的精度是非常关键的。从以前的工作中，我们知道同样的特征对于不同的查询，它的区分度是不一样。本文提出了一种新颖的查询相似性表示方法：利用特征的区分度将查询表示成一个特征排序，然后本文使用特征排序的相似度来表示查询的相似度。本文构建依赖于测试查询的训练集时，使用了两种方法来选择相似的查询：KNN 和固定距离的算法。

本文在 LETRO（TREC，OHSUMED）数据集上进行实验。实验结果表明，依赖于查询的排序方法要优于使用单一排序模型进行排序的方法。

关键词：依赖于查询的排序，排序学习，查询聚类，查询相似性

Research on Query Dependent Learning to Rank Algorithm

Major: Computer Software and Theory

Name: Luo Yemin

Supervisor: Professor Tang Yong

ABSTRACT

Ranking is the central problem for information retrieval, because the goodness of a search system is mainly evaluated by the accuracy of its ranking results. Many ranking models have been proposed in information retrieval. It is only recently machine learning technologies called 'learning to rank' have been intensively applied to the task. In most of the previous work, a single ranking model is used to handle all queries. This may not be appropriate, because there are significant differences between queries.

This paper first discusses that it is necessary to employ different ranking functions for different queries. Then this paper presents a general framework for query dependent learning to rank. The framework is based on query clustering. The framework positions the training queries into the query feature space in which each query is represented by a point. In ranking, given a test query, the framework retrieve its neighboring training queries to construct a query dependent training set, learn a ranking model with the training set, and then rank the documents with respect to the test query using the learned model. This framework can really help because it can leverage the useful information of the similar queries and avoid the negative effects from the dissimilar ones.

When we conduct query clustering, the query features used in the method and queries' similarity calculation are critical to its accuracy. From the previous work, we know that the same features' discriminations are different for different queries. This paper proposes a novel method which utilizes features' discriminations to represent queries as a feature order, and then uses feature orders' similarity to represent queries'

similarity. When we construct query dependent training set for a test query, we use two different methods to select similar queries: KNN and fixed radius method.

This paper evaluates the methods on the LETOR (TREC, OHSUMED) dataset. Experimental results show that query dependent ranking methods outperform the baseline method of using a single ranking model.

Key Words: query dependent ranking, learning to rank, query clustering, query similarity

论文原创性声明

本人郑重声明：

所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

论文作者签字：罗辉敏

2009年5月24日

学位论文使用授权声明

本人完全了解中山大学有关保留、使用学位论文的规定，即：学校有权保留学位论文并向国家主管部门或其指定机构送交论文的电子版和纸质版，有权将学位论文用于非赢利目的的少量复制并允许论文进入学校图书馆、院系资料室被查阅，有权将学位论文的内容编入有关数据库进行检索，可以采用复印、缩印或其他方法保存学位论文。

学位论文作者签名：罗辉敏　　导师签名：

日期：2009年5月24日　　日期：2009年5月24日

第1章 引言

本章节先介绍排序函数学习算法研究的应用背景——搜索引擎，接着对排序在搜索引擎中的意义和作用进行阐述，指出本论文研究的出发点和创新点，最后给出了论文的文章组织结构。

1.1 搜索引擎的产生和发展

随着互联网的普及和迅速发展，互联网上的信息越来越丰富，到了一个“信息爆炸”的时代，人们要从大量的网页数据中得到自己需要的信息。为了使人们能够从海量的网页中找到自己所需要的网页，搜索引擎(search engine)出现并迅速发展，同时它也是学术领域的一个研究热点。搜索引擎首先使用爬虫程序定期从互联网上下载网页，然后对这些网页进行索引。它接受用户提交的查询请求，从索引中检索相应的信息，然后对这些信息排序后返回给用户。现在人们能够通过搜索引擎方便地得到各种各样的信息。

现在公认的搜索引擎的鼻祖是 Archie，它是一个 FTP 上的文件搜索工具，由加拿大麦吉尔大学计算机学院的师生在 1990 年时开发出来。Archie 定期搜集多个 FTP 上的文件名信息，然后对这些信息进行索引。如果用户输入文件名进行搜索，Archie 就可以告诉用户该文件的下载地址。Archie 和搜索引擎的基本工作方式是类似的：定期搜集信息、建立索引、接受用户的检索请求。Archie 和搜索引擎的主要区别是它搜集的信息是文件名而不是网页。

随着互联网的发展和以 HTML 为格式的信息迅速膨胀。1994 年，Yahoo 公司创立，它提供基于目录的信息检索服务，支持简单的数据库查询。这就是早期的目录导航系统，他们的缺点是网站收录和更新都要由人工来维护，在信息量剧增的条件下，它的维护就变得非常困难了。

同年，真正意义上的搜索引擎 Lycos 诞生，它使用了基于 robot 的数据发掘技术，将爬虫结合到索引程序中，并支持搜索结果的相关性排序。同时，它也首先在搜索结果中使用网页自动摘要。这种搜索引擎跟我们现在使用的搜索已经非常相似了。

目前国际上比较有名的搜索引擎有 Google，Yahoo!，Live Search 等，而专

门针对中文搜索的搜索引擎有 Baidu，Sogou，天网，有道等。无论哪方面的内容，这些搜索引擎都能根据用户输入的关键字帮助人们快速地找到相关的网页。

由于知识时代信息的急剧膨胀，搜索引擎已经成为人们获取信息的重要渠道。为了满足人们获取某个行业或某种类型信息的需要，垂直搜索诞生了。垂直搜索是针对某一特定领域、某一特定人群或某一特定需求提供的有一定价值的信息和相关服务。其特点就是“专、精、深”，且具有行业色彩。例如，“新闻搜索”，“音乐搜索”，“图片搜索”，“地图搜索”，“旅游搜索”，“生活搜索”等都是常见的垂直搜索。

随着信息检索和自然语言领域研究的发展，出现了自动问答系统。传统的搜索引擎只能根据用户输入的关键字返回一系列包含该关键字的文档，用户需要从千千万万个文档中提取出他所需要的信息，因为搜索引擎不能理解用户的意图，也不能回答用户用自然语言提出的问题。自动问答系统允许用户用自然语言句子的提问，系统会自动分析用户的提问，并将问题的答案返回给用户。用户通过自动问答系统能够充分表达他的检索需求，能够很快地找到他们所需要的答案[1,2]。

1.2 排序函数学习算法研究的意义

排序模块是搜索引擎的重要组成部分。因为一个搜索系统的好坏主要由它的排序结果的正确率来评价。所以排序算法的研究一直是信息检索领域的一个研究热点。

1.2.1 搜索引擎的体系结构

一个搜索引擎的体系是一个比较复杂的系统，由好几个相互协作的模块组成，按功能来划分模块，一般而言搜索引擎的体系结构如图 1-1 所示：

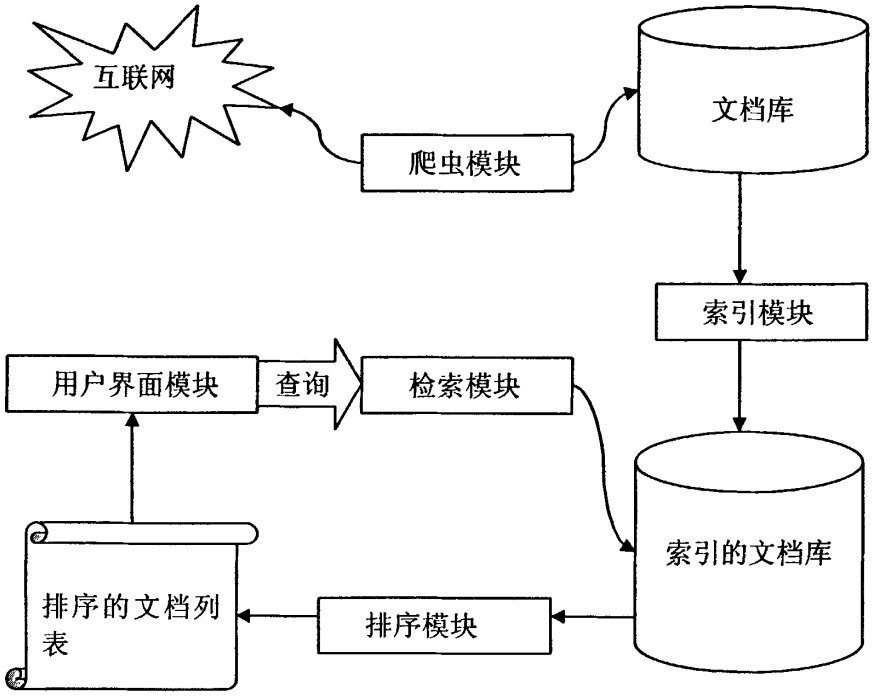


图 1-1 搜索引擎的体系结构

搜索引擎从互联网上搜集大量网页，然后对这些网页建立索引，它可以接收用户的查询请求，并根据文档与查询的相关度返回相关的网页的链接。一般的搜索引擎系统由以下 5 部分构成：

- **用户界面** 管理与用户之间的交互：

  获取用户的查询。

  获取用户的反馈信息。

  以可视化的方式显示搜索结果，包括文档的标题以及包含查询关键字的一段摘要。

- **爬虫** 是一个自动提取网页的程序，它为搜索引擎从万维网上下载网页。它的目标是尽可能多地下载万维网上的网页，搜集到尽可能多的信息。传统爬虫从一个或若干初始网页的 URL 开始，获得初始网页上的 URL，在抓取网页的过程中，不断从当前页面上抽取新的 URL 放入队列，直到满足系统的一定停止条件。

- **索引** 理解爬虫所抓取到的网页，从中抽取出索引项，用于表示文档

以及生成文档库的索引表。现代搜索引擎一般使用的是倒排索引。倒排索引结构由“词典”和“出现情况”两部分组成。对于每一个索引项，都会有一个列表记录索引项在文档库的哪些文档中出现，出现的次数以及出现的位置等。因为一般的索引结构建立的是一种“文档到单词”的映射关系，而倒排索引建立的则是一种“单词到文档”的映射关系。在日常的检索中，通常都是按照关键字进行检索的，所以，倒排索引可以更好地适合这种检索机制的需要。对中文的文档建立倒排索引，首先要对文档进行分词，也就是将一段中文的文字按照语义或者语法信息切割成一些词语或者短语。对于英文文档的建立倒排索引，首先要对单词进行词形还原（lemmatization）和词干抽取（stemming)。无论是中文文档还是英文文档，在建立倒排索引之前都要去掉停止词（stopwords)。

检索 根据用户的查询在倒排索引中快速检索文档，并找出相关的文档。查询通常包含多个关键词，而每一个关键词都对应一个索引的列表，因此查询时会涉及到多表合并的情况。为了使合并能够更高效，通常的倒排表都会设置有跳过点（skip pointers)。所谓的跳过点，是指倒排表中某个文档后面若干个文档的编号，它是一种有效的方式来允许我们跳过倒排表中查询结果不会出现的某些部分的处理。

如果包含查询词的文档比较少，查询经常使用一种叫“查询扩展”的技术。 查询扩展是指将原来的查询增加新的关键字来重新查询。搜索引擎会将使用者输入的查询先做一次检索，根据检索出来的文件，选取出适合的关键字，加到查询中重新检索，借此来找出更多的相关文件。查询扩展也可以使用查询词的同义词或者近义词作为新增的关键字来进行重新查询。

- 排序 检索模块检索到与查询相关的文档后，排序模块以文档与查询之间的相关性为依据对这些文档进行打分，并按照分数的高低对文档进行排序。前人对排序模块作出了许多的研究。在过去的研究中，提出了很多定义查询与文档之间相似性的依据，并被广泛使用，如 BM25，PageRank，Hits 等。随着人们提出的排序依据的增多，我们很难通过

手工的方式来调整不同依据之间的权重。于是，有人提出用机器学习的方法来学习排序模型（Learning to Rank for IR），这是近年来信息检索领域的一个研究热点。最近也有人提出个性搜索的概念，它是指根据个人的兴趣，爱好，搜索习惯等，进行文档的排序，同样的查询词，对于不同的人，排序的结果是不一样的。搜索引擎接受的是自然语言的查询，不同查询之间的区别是很大的，因此有人提出对于不同的查询，使用不同的排序方法。

1.2.2 排序模块在搜索引擎中的作用

网页排序是信息检索的关键步骤，它的排序结果直接影响使用者对搜索引擎的评价。通常包含用户查询关键字的网页达到数以千万计，有的常见词甚至达到亿的数量级，而用户通常所关心的仅仅是前几页的搜索结果，他们很少对搜索引擎返回的结果进行翻页查找，就算进行翻页查找，翻页的页数也不会太多。所以，他们关心的也就是前一百条，甚至是前十条的搜索结果。在数以千万计的相关网页中，如何将用户感兴趣的网页排在比较前的位置，对所有搜索引擎来说都是一个极大的挑战。

对排序模块的研究一直是信息检索领域的一个研究热点。许多学术机构和商业的搜索引擎公司都致力于改善搜索引擎中的排序模块，来提升客户的满意度。各种搜索引擎的技术改进和优化，都直接反应到搜索结果的排序上。在过去的研究中，研究人员也提出了很多行之有效的排序方法。例如，Google 提出的 PageRank 算法。因为使用了 PageRank 算法，Google 对搜索结果的排序比其它搜索引擎都要好，它保证让绝大部分用搜索的人，都能在搜索结果的第一页找到他想要的结果。优秀的排序结果，使 Google 迅速占领了搜索引擎的市场份额，创造了互联网界乃至 IT 业界的一个奇迹。可以说，Google 的成功与 PageRank 算法是分不开的。

目前大多数的搜索引擎，它的搜索结果都是单一化的，任何人搜索同一个词的结果都是一样。为了更好地满足使用者的需要，改善排序的结果，有人提出了“个性化搜索”，对于不同的用户返回不同的结果，也就是说，对于不同的使用者，使用的排序方法是不一样的。为了收集到使用者的相关信息，各大

搜索引擎纷纷推出了自己的搜索工具条，如“Google Toolbar”，“ 百度搜霸”，“MSN Toolbar”和“Yahoo 工具条”等。这些工具条可以收集用户的搜索习惯，兴趣爱好等，这样既是为用户提供个性化搜索的一种手段，也是搜索引擎改善排序结果的方法。因为利用收集到的用户反馈信息，可以部分推断出网页与查询的相关性以及网页之间的优先关系，从而改善排序结果。微软亚洲研究院的研究人员利用搜索工具条搜集到的用户信息，提出了一个叫 BrowseRank [49]的排序算法，可以达到比 PageRank 更好的排序结果。

1.3 本文研究的出发点

通过机器学习来进行排序函数学习的研究在近年来受到了广泛的关注。它是机器学习研究中较新的一个领域，旨在通过利用标注好的训练数据和机器学习的算法构建排序模型。过去也有一些学者对搜索引擎中查询的多样性进行研究，他们提出了对于不同的查询应该采用不同的排序模型[8]。

本文的研究是在总结了前人用机器学习的方法进行网页排序以及依赖于查询的排序等研究的基础上提出的。Geng 等[3]使用 KNN 算法对查询进行聚类，然后针对不同的查询进行排序函数的学习，得到依赖于查询的排序函数，获得了很好的效果。本文在 Geng 等[3]工作的基础上，利用 Duh 等[4]在研究中发现的不同特征对于不同查询的区分度不一样的结论，提出了新的定义查询之间相似性的方法，并提出了依赖于查询的排序函数学习算法框架。

1.4 本论文的组织

正文共分为 5 章。

第 1 章首先介绍了搜索引擎的背景和发展，然后介绍了搜索引擎的体系结构和各个主要模块，接着阐述了排序模块对于搜索引擎的重要性，最后指出了本文研究的出发点。

第 2 章介绍了相关工作，包括通过机器学习来进行排序函数学习的研究，以及依赖于查询的排序的研究。

第 3 章对主要阐述了本文的主要工作。首先详细说明网页排序中，对于不

同的查询应该使用不同的排序模型，然后介绍本文提出的基于查询聚类的，依赖于查询的排序学习算法框架，接着介绍查询聚类的方法，最后介绍提高效率的两种缓存方法。

第4章首先介绍实验的数据集和实验的评价尺度，然后从实验数据作对比，用实验结果说明依赖于查询的排序方法要优于使用单一排序模型进行排序的方法。

第5章总结全文的研究结论和成果，探讨文中的算法在下一步研究工作中的展望。

第2章 相关工作

本章介绍搜索引擎中排序模型的相关研究工作。首先介绍传统的排序模型，接着介绍用机器学习的方法进行网页排序的相关研究工作，最后介绍查询多样性的一些相关研究。

2.1 传统的排序模型

传统的排序模型只使用少量的特征，例如词频（tf），逆向文件频率（idf）和文档长度等。

如果一个词在一个文档中出现的频率越高，那么这个词与这个文档的相关性就越高。tf是指某个关键字在某个文档中出现的频率：

tf_{ij} = 词语k_i在文档d_j中出现的频率

如果一个词语在许多不同的文件中出现，那么它的区分度就比较小。逆向文件频率是一个词语普遍重要性的度量。某一特定词语的 idf，可以由总文件数目除以包含该词语之文件的数目，再将得到的商取对数得到：

df_i = 包含词语k_i的文档数量

$idf_{ij} = \log_2(N/df_i)$（$N$：文档库中文档的总量）

tf-idf 是一种统计方法，用以评估一个词语对于一个文档集或一个语料库中的其中一个文档的重要程度。词语的重要性随著它在文档中出现的次数成正比增加，但同时会随著它在语料库中出现的频率成反比下降。tf-idf 加权的各种形式常被搜寻引擎应用，作为用户查询与文档之间相关程度的度量或评级。

在传统的排序模型中，比较经典的排序函数还有 BM25[48]和 LMIR[12, 13]。

BM25 由 Stephen E. Robertson 等人根据概率语言框架提出。给定一个查询 Q，它包含关键字 $q_1,...q_n$，文档 D 的 BM25 分数如下计算：

$$score(D,Q) = \sum_{i=1}^{n} idf(q_i) \cdot \frac{f(q_i,d) \cdot (k_1+1)}{f(q_i,d) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{avgdl}\right)} \qquad (2\text{-}1)$$

$f(q_i,d)$ 是关键字 q_i 在文档 D 中出现的频率。

$|D|$是文档D的长度。

avgdl 是文档库中所有文档的平均长度。

k_1和b是两个参数，通常的取值是$k_1 = 2.0$，$b = 0.75$。

最近，有些研究表明，一些附加的特征对于文档检索也是有帮助的，包括结构化特征（如标题，锚文本和URL等）和独立于查询的特征（如PageRank[14]和HITS[15]等）。

PageRank是由Google提出的一种排序方法，它是据网页之间相互的超链接来对网页进行排序的技术，它体现了网页的重要性。PgeRank通过互联网中的超链接关系来确定一个网页的重要性。PageRank把从A网页到B网页的链接解释为A网页给B网页投票，添加网页B的重要性，PageRank根据投票来源和投票目标的等级来决定投票目标的新等级。也就是说，链接到一个页面相当于对该页投一票，一个高等级的网页可以使其链接到的低等级网页的等级得到提升。一个页面的PageRank是由所有链向它的页面的重要性得到的。一个有较多链入的页面会有较高的等级，相反如果一个页面的链入页面比较少，那么它的等级就比较低。

上面介绍的是传统的排序模型，它们有以下缺点[45, 46]：

- 因为传统的排序模型使用的特征比较少，参数也比较少，所以可以用手工的方法来调整参数的取值。但是，随着研究的深入，越来越多的特征被证明对于提高排序的结果有作用，这使得参数的调整变得越发困难。
- 对于两个传统模型的比较，我们也会遇到困难，因为一个模型可以被调到过拟合（over-fitting），而另外一个则没有。
- 上百种传统模型已经被提出，对于多种有效的模型，我们难以用一种有效的方式将它们结合起来。

基于上述理由，有人提出了用机器学习的方法来进行排序函数的学习（Learing to Rank for IR）。

2.2 用机器学习的方法进行排序函数的学习

排序函数的学习可以定义成如下问题。在训练的时候，已知训练集，它包括许多查询，与查询关联的检索到的文档，以及这些文档的相关性标注。查询q_i通常表示成关键字的列表$\{t_{i1},t_{i2},...,t_{im}\}$。相关性标注通常是通过人工进行的，它用一个整数表示（例如 0，1，2 等），可以表示文档之间的排序。训练的是从训练集训练出一个模型，它的目的是使得这个模型在测试集上的损失函数尽可能地小。在测试的时候，从训练集学习到的模型用来计算查询与文档之间相关性的估计值，然后按照估计值的降序返回一个文档的列表。

假设$Q=\{q_1,q_2,...,q_m\}$是训练集中的查询的集合。每一个查询q_i都有相应的检索到的文档列表$\mathbf{d}_i=\{d_{i1},d_{i2},...,d_{in(q_i)}\}$和一个标注的列表$\mathbf{l}_i=\{l_{i1},l_{i2},...,l_{in(q_i)}\}$，$n(q_i)$表示列表$\mathbf{d}_i$和$\mathbf{l}_i$的大小，$d_{ij}$表示$\mathbf{d}_i$的第$j$个文档，$l_{ij}$表示文档$d_{ij}$的相关性标注。每一个查询-文档对$(q_i,d_{ij})$可以用一个特征向量$\phi(q_i,d_{ij})$来表示，向量的每一维表示一个特征（例如 tf, idf, PageRank 等）。训练集可以表示成$T=\{q_i,\mathbf{d}_i,\mathbf{l}_i\}_1^m$。排序模型是一个从文档的特征向量到文档相关性得分的一个线性函数$f=w^T\cdot\phi(q_i,d_{ij})$。简而言之，训练的过程就是确定$w$的过程。

排序函数学习的框架如图 2-1 所示：

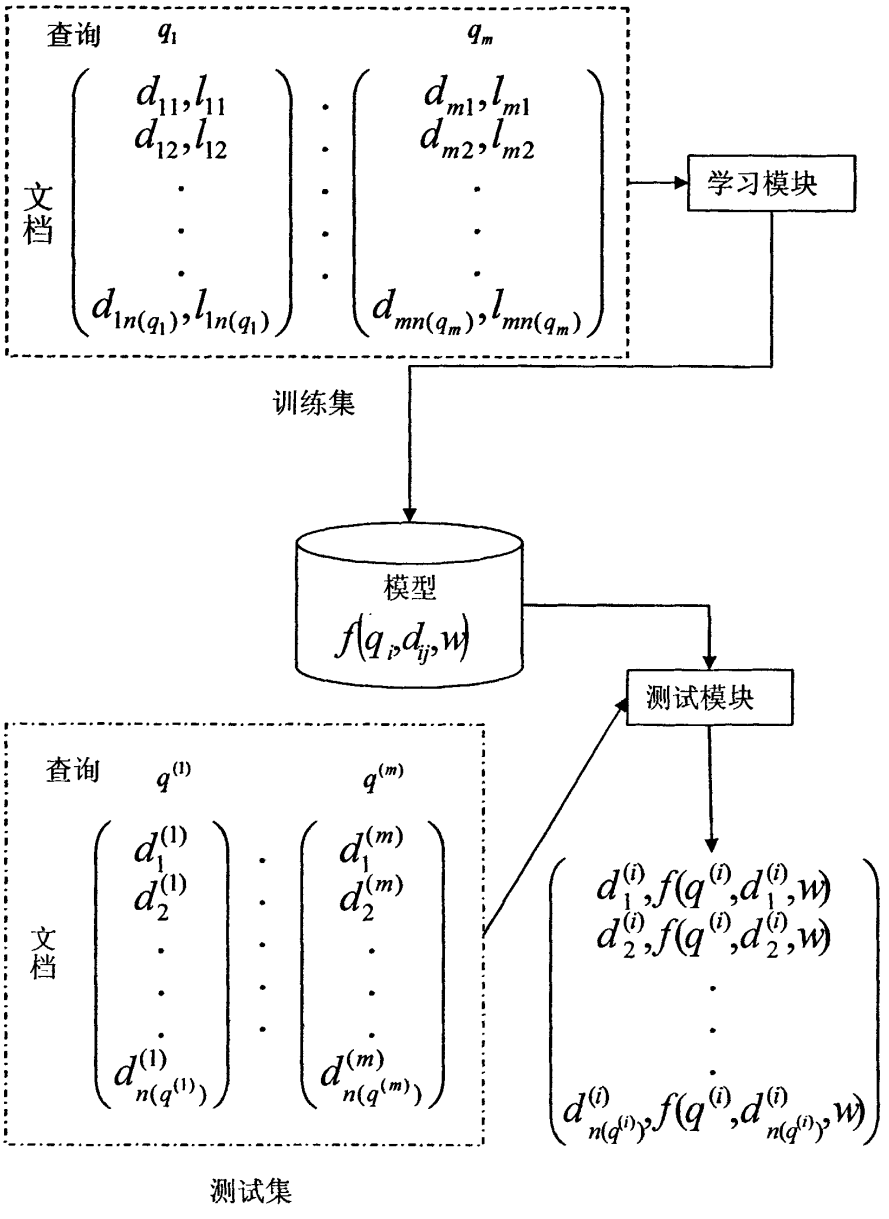


图 2-1 排序函数学习的框架

Cao 等[16, 17]将排序函数学习的方法可以分为三类：（1）Poitwise 方法；（2）Pairwise 方法；（3）Listwise 方法。分别介绍如下：

2.2.1 Pointwise 方法

在 Pointwise 的方法中，每一个训练样本都由一系列用特征向量表示的文档以及文档与查询之间的相关性来表示。学习的过程试图将特征向量映射到排序。这种方法将对文档的排序转换为对单个文档的回归或分类。

Pointwise 方法的代表作有：IR 判别模型（Discriminative models for information retrieval）[18]和 McRank[19]。

IR 判别模型将排序问题看成是分类问题：将与查询关联的文档分为两类：与查询相关的（relevant）和与查询不相关的（non-relevant）。它使用了两个判别模型，分别是最大熵（ME）和支持向量机（SVM）。这两个使用的特征如表 2-1 所示：

表 2-1 IR 判别模型中每个文档抽取的特征

特征		特征	
1	$\sum_{q_i \in q \cap d} \log(c(q_i,d))$	4	$\sum_{q_i \in q \cap d} \left(\log\left(\frac{\lvert C \rvert}{c(q_i,C)} \right) \right)$
2	$\sum_{i=1}^{n} \log\left(1+\frac{c(q_i,d)}{\lvert d \rvert} \right)$	5	$\sum_{i=q}^{n} \log\left(1+\frac{c(q_i,d)}{\lvert d \rvert} idf(q_i) \right)$
3	$\sum_{q_i \in q \cap d} \log(idf(q_i))$	6	$\sum_{i=q}^{n} \log\left(1+\frac{c(q_i,d)}{\lvert d \rvert} \frac{\lvert C \rvert}{c(q_i,C)} \right)$

最大熵原理是根据样本信息对某个未知分布做出推断的一种方法。最大熵对所有已知信息建模，不对未知信息进行任何假设。给定一些事实，应该选取符合这些知识但熵值最大的概率分布的模型。模型最大化后验条件分布 $P(R \mid D)$ 的熵，约束条件是模型对特征函数的预测值与训练集中观察到的经验频率分布一致。可以用以下公式表示：

$$P(R \mid d,q) = \frac{1}{Z(q,d)} \exp\left(\sum_{i=1}^{n} \lambda_{i,R} f_i(d,q) \right) \qquad (2\text{-}2)$$

$P(R \mid d,q)$ 表示与 q 关联的文档 d 属于类别 R 的概率。$Z(q,d)$ 是正则化常量。$f_i(d,q)$ 是文档 d 的特征函数，它的权重是 $\lambda_{i,R}$，n 是特征的数目。权重是从训练集上学习到的。作者使用了后验相似度比例对数（log-likelihood ratio）作为文档的分数进行排序：

$$\log\frac{P(R|d,q)}{P(\bar{R}|d,q)}=\sum_{i=1}^{n}(\lambda_{i,R}-\lambda_{i,\bar{R}})f_i(d,q) \tag{2-3}$$

支持向量机寻找一个最大化间隔的超平面来分开两类训练样本。一般认为，间隔越大，分类器的泛化性能越好。超平面处于一个被称为核空间的高维空间，核函数将特征空间映射到核空间。支持向量机的判别函数如下：

$$g(R|d,q)=w\bullet\phi(f(d,q))+b \tag{2-4}$$

w是支持向量机从训练集中学习到的核空间的权重向量，$\bullet$表示内积，b是一个常量，ϕ是核函数。$g(R|d,q)$表示超平面。在训练过程中，支持向理机使正样本（相关文档）$g(R|d,q)\geq 1$，负样本（不相关的文档）$g(R|d,q)\leq 1$。因此，判别函数的值越大，我们就越确信某个文档是相关文档。

2.2.2 Pairwise 方法

Pairwise 方法不再假定文档绝对的相关性。每一个训练样本由文档对和它们之间的优先关系组成。Pairwise 方法将排序问题归结为文档对之间的分类问题，我们的目标是将这些文档对分到正确和不正确这两种排序类别中。在测试的时候，它的输入空间是文档对，输出空间是优先关系。优先关系的构造如图2-2所示。

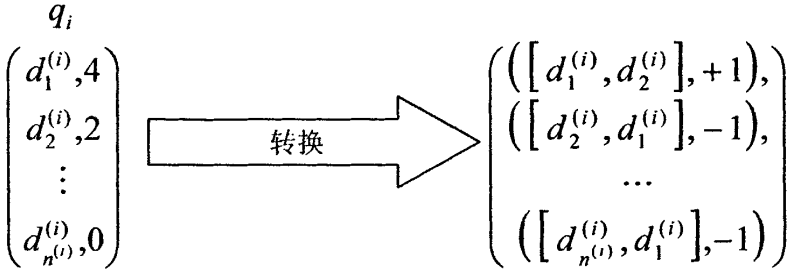


图 2-2 优先关系的构建

Pointwise 方法的代表作有：RankNet[20]，FRank[21]，RankBoost[22]和 Ranking SVM[23]等。

RankNet 算法的输入是集合R^d中的文档对$[A,B]$，还有文档 A 排在文档前

面的目标概率$\overline{P_{AB}}$。在训练的时候，目标概率$\overline{P_{AB}}$的值是已知的，它可以取三个值，$\overline{P_{AB}}=\{0,0.5,1\}$。$\overline{P_{AB}}=1$意味着我们知道$d_A \triangleright d_B$，$\overline{P_{AB}}=0$意味着我们知道$d_A \triangleleft d_B$，$\overline{P_{AB}}=0.5$意味着我们不知道两个文档之间优先关系的任何消息，也就是说这两个文档的相关性是一样的。训练的目的是得到模型$f: R^d \to R$，它使得对测试样本的排序，是由f函数的值确定。也就是对说，如果$f(d_1)>f(d_2)$，意味着模型预测$d_1 \triangleright d_2$。

模型的后验概率$P(d_i \triangleright d_j)$用$P_{ij}$表示。$\overline{P_{ij}}$是这些后验概率的目标值。定义$o_i \equiv f(d_i)$，$o_{ij} \equiv f(d_i)-f(d_j)$，损失函数用交叉熵来表示：

$$C_{ij} \equiv -\overline{P_{ij}} \log P_{ij} - (1-\overline{P_{ij}}) \log(1-P_{ij}) \tag{2-5}$$

输出到概率的映射使用逻辑函数

$$P_{ij} \equiv \frac{e^{o_{ij}}}{1+e^{o_{ij}}} \tag{2-6}$$

这样，C_{ij}变成：

$$C_{ij} \equiv -\overline{P_{ij}} o_{ij} + \log(1+e^{o_{ij}}) \tag{2-7}$$

C_{ij}变得接近线性函数，对于有噪声的样本来说，它比二次损失函数更加鲁棒。

RankNet使用神经网络作为模型，梯度下降作为算法，来优化交差熵的损失。

2.2.3 Listwise方法

Listwise方法直接对排序列表进行操作。与Pointwise方法和Pairwise方法将排序看成回归和分类不同，Listwise方法直接在文档列表上进行训练。它将一个文档排列作为一个训练实例，然后学习一个排序函数用于文档进行排序。通过将与查询关联的文档列表作为训练实例，我们可以自然地得到排序（位置）信息和查询层面的信息，因此我们可以在训练的时候结合更多排序方面的独特

属性。Listwise 方法分为两大类，一类是直接优化 IR 的评价标准，一种是直接对排序定义损失函数。直接优化 IR 评价标准的方法试图优化基于排序评价标准，或者至少与它们相关的评价标准。直接对排序定义损失函数的方法试图最小化定义在排列（排序的文档列表）上的损失函数，损失函数通过考虑 IR 排序的独特属性来设计。

Listwise 方法的目标是学习一个从查询到排序的函数 $h: X \to Y$。X 是查询的集合，Y 是文档集上所有可能的排序。为了衡量预测 $\hat{y} = h(x)$ 的好坏，定义了一个损失函数 $\Delta: Y \times Y \to R$，如果正确的排序是 y，$\Delta(y, \hat{y})$ 衡量预测 $\hat{y}$ 的惩罚。排序函数 h 的目标是使得风险 $R_P^{\Delta}(h) = \int_{X \times Y} \Delta(y, h(x)) dP(x, y)$ 最小化。但是 $P(x, y)$ 是未知的。可以使用训练集上的经验风险 $R_P^{\Delta}(h) = \frac{1}{n} \sum_{i=1}^{n} \Delta(y_i, h(x_i))$ 代替。

直接优化 IR 评价标准的代表作有：SVM-MAP[24]，AdaRankt[25]和 RankGP[26]。直接对排序定义损失函数的有：RankCosine[27]，ListNet[17]和 ListMLE[29]。

Listwise 方法中的损失函数可以结合特定的评价标准，SVM-MAP 采用的优化标准是 MAP，它的损失函数定义为：

$$\Delta_{map}(y, \hat{y}) = 1 - MAP(rank(y), rank(\hat{y})) \tag{2-8}$$

SVM-MAP 采用的方法是学习一个判别函数 $F: X \times Y \to R$。给定一个查询 x，对它进行预测的排序 y 使得判别函数取得最大值：

$$h(x) = \arg\max_{y \in Y} F(x, y) \tag{2-9}$$

SVM-MAP 采用输入线性函数表示 F：

$$F(x, y) = w^T \Psi(x, y) \tag{2-10}$$

排序 y 的分数使用如下公式计算：

$$\Psi(x, y) = \frac{1}{|C^x| \cdot |\bar{C}^x|} \sum_{i: d_i \in C^x} \sum_{j: d_j \in \bar{C}^x} [y_{ij} (\phi(x, d_i) - \phi(x, d_j))] \tag{2-11}$$

ϕ 表示文档的特征向量，如果文档 d_i 排在文档 d_j 前面，$y_{ij} = 1$，如果文档 d_j

排在文档d_i前面，$y_{ij}=-1$。

SVM-MAP 使用 SVM-Struct[30, 31]作为优化的框架，它的目标是：

$$\min \frac{1}{2}\|w\|^2+\frac{C}{N}\sum_{i=1}^{N}\xi_i \qquad (2\text{-}12)$$

约束条件是$\forall i,\forall y\in Y\setminus y_i : w^T\psi(x_i,y_i)\geq w^T\psi(x_i,y)+\Delta(y_i,y)-\xi_i$

SVM-Struct 的训练分为三步：

第一步：在当前工作集上进行优化。

第二步：使用第一步中的模型从指数个约束中找出最大违反约束。

第三步：如果第二步中找到的最大违反约束比工作集中的最大违反约束更违反，添加它到工作集。

因为约束一共有指数个，SVM-Struct 的训练过程要求在每一轮迭代中找到最大违反约束，SVM-MAP 使用以下时间复杂度是$O(n\log n)$的策略：

分别对相关和不相关的文档按照$w^T\phi(x,d)$排序，从最优的排序开始，将不相关的文档按照某种条件依次插入相关文档的列表中，这样得到的排序即为最大违反约束。

2.3 依赖于查询的网页排序

在信息检索中，不同查询之间可能会有比较大的差异。例如有些查询是关于人名的，对于这种查询，用户想得到往往是这个人的主页，而有的查询是关于当前热点的，如“经济危机”，对于这种查询，用户想得到的是关于这个热点不同方面的消息。

过去的研究者对查询分类进行了深入的研究[5, 6, 7, 8, 9, 10]，对于每一个查询，试图将它分到预先设计好的特定的类别中。Beitzel 等[6, 7, 10]将查询按照它们的主题进行分类，例如分成计算机类，娱乐类和信息类等。Rose 等[5, 8, 9]将查询按照用户的搜索目的，意图和需要进行分类，例如分为导航类，信息类和资源类。在对查询进行分类时，使用了机器学习的方法，如支持向量机（SVM）等。

KDD CUP 2005 曾经举办对过查询进行分类的比赛[33]。它的比赛内容是将800000 个英特网上的用户搜索查询分到预先定义好的 67 个类别中，要求参赛

者在两个月的时间用自动的方法完成分类工作。赛会提供了111已经标注好类别的查询。该比赛并没有提供训练数据，参赛者可以使用任何开放的资源，例如搜索引擎，字典等。用户的查询一般来说比较短，有时只有一个单词，对它们进行分类是一件很困难的事情。尽管很多参赛者通过各种办法获取关于查询的额外信息，例如通过WordNet进行查询扩展，利用搜索引擎进行搜索得到的网页片段和网页标题等，但分类结果还是不甚理想，最好的分类精度仅为42%左右。

对于搜索引擎来说，对不同的查询使用不同的排序模型是有意义的，因为这样可以结构查询的特点来进行网页排序，在排序的时候可以利用查询的特定信息。Kang等[8]将查询分为两类：主题类和主页类。对于主题类的查询，他们使用Lemur作为排序模型，该排序模型使用的是传统的排序函数，如tf-idf和BM25等，这些排序函数只考虑查询与文档之间的相似性，并没有考虑网页重要性等信息，这对于主题查找的多样性是有好处的。对于主页类的查询，他们除了使用Lemur，还加上了URL信息和PageRank值作为排序依据，因为对于主页类的查找，用户想找到的只是一个网页，所以应该利用PageRank值将比较重要的网页排在较前的位置。实验结果表明对不同查询使用不同的模型比对所有查询使用单一模型的排序结果更好，但是改善结果并不十分明显，因为对查询进行分类的准确率并不高。

最近，Chakrabarti等[32]提出了对于不同的查询，应该使用不同的IR评价标准。例如，对于导航型的查询，如“IBM”，还有事实性的查询，如“中国的首都”，用户关心的往往是搜索结果的第一条，所以应该使用MRR作为评价标准。对于检索型的查询，如“经济振兴计划”，用户想得到的是关于这个查询的各方面信息，用户检索的要求是信息的多样性和丰富性，所以应该使用NDCG作为评价标准。

在已有的通过机器学习进行网页排序的模型中，它们大多没有考虑到不同查询之间的差异性。它们用训练集中的所有样本进行训练，得到一个模型，然后将这个模型应用到测试集中的所有样本。在排序函数学习领域，Geng 等[3]提出了一种依赖于查询的排序函数学习方法，他们构造了对查询进行聚类的排序模型。他们将训练集中的查询表示成查询特征空间中的一个点。在对测试集

中的查询q进行文档排序的时候，首先从训练集中找到k个与q最相似的查询，然后用这k个查询训练一个模型，最后用训练得到的模型对q进行关联文档的排序。因为对于每个查询，都要在线训练模型，这对应用来说是不切实际的。作者提出了两个可以离线训练的近似模型，并对这两种近似模型进行理论分析和经典验证。实验结果表明依赖于查询的排序模型比使用单一模型进行排序的效果要好。

第3章 依赖于查询的排序学习

本章节首先从多个角度介绍网页搜索中查询的多样性，然后指出对于不同的查询应该使用不同排序模型。因为对查询进行分类是比较困难的，于是我们提出基于查询聚类的，依赖于查询的排序学习框架。然后介绍对查询进行聚类的方法，包括如何定义查询的特征和如何定义查询的相似性。最后介绍两种减少排序时间的缓存策略。

3.1 信息检索中查询的多样性

网页搜索中，查询是多种多样的，不同的查询之间往往有比较大的差异的，具体表现在以下几方面：用户搜索的目的，用户搜索的主题，查询词的长短，查询的表达形式，相关文档的数量等。

根据用户搜索目的和搜索需要的不同，Lee 等[9]将用户的查询分为以下两类：导航类和信息类。Rose 等[5, 8, 30]在此基础上将查询进一步分为三类，Rose 等[5]将查询分为信息搜索类，主页搜索类和服务搜索类，Broder[30]将查询分为导航类，信息类和事务类。

对于导航型的查询，用户的目的是找到一个网页，这个网页是心目中已经存在的，它可能是用户过去访问过，或者他们确定这个网页是存在的，用户执行这种查询大多数情况下是搜索公司的主页，研究机构的主页或者组织的主页，例如：

用户搜索“中山大学”，他可能想访问：http://www.sysu.edu.cn

用户搜索“微软”，他可能想访问：http://www.microsoft.com

用户搜索“新华社”，他可能想访问：http://www.xinhuanet.com

在 TREC 2001 的 web track[35]中，有一个主页查找的比赛。该比赛包含 145 个查询，在 CSIRO 发布的 WT10g 文档库中搜索。这种主页查找可以看作是导航类查询的搜索。

导航类查询的特点是用户心目中有一个唯一的权威的网页，他假定搜索结果中只有一个是“正确”的，他的兴趣主要集中于那个网页，在返回的搜索结果中，他可能只点击心目中那个网页。用户之所以执行这类搜索，是因为他觉

得通过搜索引擎查找比直接输入URL更方便，或者他不知道URL。Google有一个“手气不错”搜索功能[35]，用户可以通过它告诉搜索引擎他是想搜索一个特定的网页。

在网页搜索中，经常使用链接信息和URL信息作为排序的依据。链接信息是指PageRank等衡量网页权威性的指标，URL信息是指对URL的字符串进行分析，使用URL信息最直接的方法是将查询与URL进行字符串匹配。对于导航类的查找，链接信息和URL信息，还有网页的标题，锚文本等信息都是非常有用的[8]，导航类的查询通常是实体名字或者专有名词，要求检索到的网页包含所有查询词，对于提高排序结果也是有帮助的[8]。评价导航类查询的结果好坏的主要标准是搜索引擎返回的第几个网页是用户心目中的那个网页，用户想要的网页排得越靠前，结果越好。

信息类查询的目的是从一个或者多个网页中获得关于某个主题的相关信息。例如，用户搜索“支持向量机”，他会访问多个提供这个查询背景知识的多个网站或者网页，他预先并没有假设存在一个唯一“正确”的特定网页，他们愿意点击多个结果。

在万维网普及以前，信息检索中的查询可以等价为信息类的查询。在那个时代，可以使用的全文检索数据库都是针对某个特定领域的，如医学，法律等，而用户主要是学生，研究人员，律师等。他们搜索的目的是获取与查询有关的资料。

信息类的查询又可分为“有方向的”，“无方向的”，“建议”，“地点”和“列表”等类型[9]。用户想通过“有方向的”查询学习到关于某个主题特定的知识，这些知识可以是确定性的，也可以是开放性的。用户不一定要以问题的形式表达查询，例如用户想知道美国的首都是哪个城市，他可以在搜索引擎中输入“美国的首都”，他不一定要输入“美国的首都在哪里”。用户也可能输入开放性的问题，例如“金属为什么会发光”。用户想通过“无方向的”查询学习到关于某个主题任意的知识，例如用户输入查询“金融危机”，他的目的是“告诉我一些‘金融危机’的东西”。“无方向”的查询包括科学，医学，历史，新闻，热点话题等等。对于“建议”型的查询，用户想获得一些建议，主张或者指导，例如“如何减肥”。通过“地点”类型的查询，用户想搜索到现实世界中的什

么地方可以从事某种活动，如购物等。用户通过“列表”型的查询，想得到一个可接受的推荐网页列表，列表中的任何一个网页都可能帮助我达到某些底层的未指明的目标，例如查询词“旅游”。

信息类查询的特点是用户心目中没有一个特定的网页，他想通过访问多个网页来了解关于某个主题的信息。信息类的查询通常是与某个概念相关的关键字，或者是用户想知道的东西。但是，不同人对于同一个概念的表达是不一样的，所以对于信息类的查询，在排序中结合链接信息和 URL 信息对于提高排序结果没有太大的帮助，有时甚至会得到反效果[8]，而要求检索到的网页包括部分查询词，而不是所有查询词的排序结果则比较好[8]。信息类查询的返回的结果应该尽量包含多个不同的方面，排在前面的文档应该能从不同角度全面地表现该主题。如果关于某一方面有多个文档，应该只选择一个排在前面。评价信息类查询搜索结果好坏的标准是返回结果的相关性和多样性。因为用户心目中没有特定的网页，他的目的是获取某个主题的相关信息，所以好的搜索结果应该从不同的角度，不同的侧面去帮助用户了解他感兴趣的主题。

用户通过资源（事务）类的查询的目的是从事某些以网页为载体的活动。他们想通过网络获取某种服务或者取得某些资源，而不是想得到信息。这类查询的特点是，用户找到一个网站后，他会有进一步的交互活动。在线购物是事务类搜索的一个典型例子，电子商务网站一般会提供搜索产品的搜索引擎，用户可以搜索各种想要的商品，例如在广州的用户如果想购买鲜花，他可能会搜索“广州 鲜花”。

Gougou 是为了满足用户获得资源需要的一个搜索引擎。用户如果想下载某个软件到本地硬盘进行安装，而他又不知道哪里有得下载，这时他会使用搜索引擎来找到可以下载软件的网页，例如用户想使用 Firefox 浏览器，他可能会在搜索引擎中输入“Firefox”，然后下载安装。用户可以通过搜索引擎找到电子书，小说等，例如用户想阅读《Java 编程思想》，他可能会在搜索引擎中输入书名，然后下载该书的电子版；用户可以通过 Google 的学术搜索检索各种论文，他们可以在计算机上离线阅读，也可以打印出来再阅读。用户取得这些资源不是为了得到某种信息，而是为了使用这种资源本身。下载电影，音乐，观看在线视频等是互联网的主要应用，用户可以通过各种资源类的搜索工具搜

索自己喜欢的娱乐资源，例如用户想搜索某首歌曲，他可以在百度的 MP3 搜索中输入歌曲的名字或者专辑名称等。资源类的搜索还包括电子地图的搜索，出行信息的搜索等，例如 Google，百度等搜索引擎都提供地图搜索功能，用户可以通过输入地名定位到地图上的地点；有一些专门针对生活搜索的搜索引擎会提供列车搜索，航班搜索，公车搜索等服务。

按照查询的主题来划分，查询的种类是非常多的，例如可以是计算机类，购物类，体育类，生活类等等。KDD CUP 2005 的查询分类比赛就将查询按照它们的主题分成 7 大类 67 小类，如表 3-1 所示[33]。

表 3-2 查询按主题划分的类型

粗分类名	细分类名
计算机	硬件，互联网，移动计算，多媒体，网络与通信，安全，软件，其它
娱乐	庆典，游戏，幽默搞笑，电影，音乐，图片，广播，电视，其它
信息	人文艺术，公司行业，科学技术，教育，法律政治，社区宗教，图书资料，其它
生活	图书杂志，汽车，工作职业，约会，家庭小孩，时尚服饰，金融投资，食品烹饪，家具器皿，礼物收藏品，健康，风景园林，宠物，房地产，宗教信仰，用具，旅游度假，其它
在线社区	聊天和即时通讯，论坛，主页，人物搜索，个人服务，其它
购物	竞价拍卖，商店商品，导购，出租租用，折扣，其它
体育	美式足球，赛车，棒球，篮球，曲棍球，新闻与得分，赛程门票，足球，网球，奥运，户外娱乐，其它

不同查询的相关文档数量可能差别很大。对于一些热门的查询，例如“NBA”，它有非常多的相关文档，而对于一些冷门的查询，例如“中山大学 春晖园”，与之相关的文档是比较少的。对于热门的查询，排序时考虑 PageRank 等衡量网页权威性的指标是很重要的，而对于冷门的查询，主要考虑文档与查询的相关性，没必要考虑网页的权威性。

网页搜索中查询的长度也是不一样的，一般而言查询词是比较短的，但是查询也可以很长，长到一个句子或者一段文字。

查询的表达形式也是多种多样的，查询可以是一个字，一个词，几个词语的组合，短语，甚至是一个句子。

3.2 排序模型应该依赖于查询

网络是丰富的各种信息来源。它包含文档内容，网页目录，多媒体数据，用户档案等。互联网中的信息不仅种类繁多的，而且数量庞大，同时搜索引擎的用户行为也是不可预知的。在对搜索的结果进行排序时，可以使用的特征有多种，包括相关性，链接信息和 URL 信息等。如果仅仅只考虑单一的特征，那么搜索的结果在某些情况并不如人意[37]。例如，基于相关性的方法在处理词汇多样性和网页质量方面会遇到困难，而基于链接信息的方法则会受到不完全链接和噪声链接的干扰。过去有研究表明，排序的时候结合多种特征可以克服使用单一特征的缺点，对于提高排序的结果是有帮助的[37, 38]。

此外，用户提交给搜索引擎的查询也是多种多样，无论是语义，搜索目的，搜索主题，查询词的长短，查询的表达形式和相关文档的数量都有很大的不同。在对文档进行排序的时候，不仅应该结合多种特征，同时也应该考虑到查询的多样性，例如：

对于用户查询“TD-SCDMA”，如果我们将链接信息的权值设置得比较大，那么包含“3G”作为索引词的知名网站会排得比较前。

对于用户查询“刘德华歌迷会”，如果我们将相关性的权值设置得太大，那么 yahoo 的目录页面会排得比较靠前，而不是刘德华歌迷会的主页。

上述例子表明，使用单一的模型来处理所有查询是不恰当的，并不是所有情况下都应该结合多种特征进行排序。了解用户查询的意思搜索引擎的核心问题之一。正确地将用户提交的查询映射到特定的类别之中可以改善搜索的结果，尤其是对于那些具有特定领域知识的搜索引擎。过去有研究表明，对于不同的查询应该使用不同的排序模型对于提高排序的结果是有帮助的，Kang 等[8]根据用户搜索的意图将查询分成两类：主题相关搜索和主页搜索，并为这两种类型的搜索分别设计排序的模型：

对于主题相关的搜索，在排序时主要考虑相关性的特征，如 tf，idf，BM25 和 LMIR 等，加入链接信息和 URL 信息不仅不能提高排序结果，反而会有负面作用。

对于主页类的搜索，不仅考虑相关性的特征，还加入了链接信息和 URL 信息。他们还通过实验表明，加入锚文本信息有助于提高主页类搜索的排序结

果。

3.3 查询分类的困难性

依赖于查询的排序，最直接的方法就是对查询进行分类：将查询划分到预先定义好的类别，对每个类别都训练一个排序模型。Rong 等[39]在参加 TREC 2004 的搜索比赛时，曾经尝试将查询分成主题查找和主页（命名页面）查找两类，并为每类查询分别调整排序模型的参数，但他们发现查询分类并不容易，这种做法相比使用单一的排序模型，只有很少的改善。查询分类还可以应用在个性化搜索和有目的地发布广告等应用上，因此过去有很多学者对查询分类进行研究，包括按搜索目的分类和按搜索主题分类。过去的研究表明，对查询进行分类是一件非常困难的事情，因为：

一、自然语言的多义性。例如，对于查询词“apple”，有的人想搜索的是与水果苹果相关的信息，而有的人则希望搜索与美国苹果公司相关的产品信息和新闻等，而另外一些人则将“apple”看成是服装品牌，他们可能希望搜索与“苹果服装”有关的信息。如果按照主题分类，在没有用户提供的进一步信息的情况下，我们很难确定将它分在水果类，计算机类还是服装类。又例如，对于查询“CS”，我们不知道将其分在计算机类还是游戏类。因为有的人认为它是“Computer Science”的缩写，他们想搜索与计算机科学相关的信息，而有的人则认为它是“Counter-Strike”的缩写，他们可能想搜索与游戏“反恐精英”相关的信息。KDD CUP 2005 的查询主题分类比赛中，一个查询是可以属于多个类别的[33]。

二、查询词通常比较短。很多时候，查询只包含一个词，我们从查询词上可以获取的分类特征是很少的，从词本身也很难知道它的意义。与文本分类不同，因为文本比较长，可以提取的特征相对比较多。参加 KDD CUP 2005 查询主题分类比赛的参赛者们使用多种方法来收集关于查询的额外信息[33]，例如使用 WordNet 和 WikiPedia 来进行查询扩展和类别描述，利用搜索引擎进行搜索得到的网页，网页片段和网页标题，词袋，搜索引擎的目录等。Kang 等[8]在对查询按照搜索目的进行分类时，使用了以下额外信息：查询词分布，交互信息（共同出现），查询在锚文本中的使用率，词性分析（POS）信息。尽管使用了

各种各样的额外信息，但分类的结果还是不能令人满意。Lee 等[9]在根据搜索意图对查询进行分类时，使用了查询日志中的“用户点击行为”和“锚链接分布”等信息。但是，查询日志中的“用户点击行为”是在用户执行搜索后才得到的信息，如果我们想在用户提交查询时就判断他搜索的目的，这些数据是不能得到的。

三、查询是主观的，不同的人对于同一个查询有不同的理解，尤其是对于人名类的查询和软件名称类的查询[9]。如果一个查询是人名，有的人认为这是导航类的查询，他们搜索的目的是访问那个人的主页，去了解一下那个人的基本信息，最新动向等。另外一些人则认为这是信息类的查询，他们除了想访问那个人的主页外，还想通过其他相关网页去了解那个人的生平，还有相关新闻等等。例如，如果一个用户搜索“姚期智”，他可能想去访问姚期智教授的个人主页，也可能想通过多个相关网页去了解姚期智教授的学术生涯，与获得“图灵奖”有关的研究工作，以及回国后的贡献和成就等。如果一个查询是软件名称，如“Firefox”，“极品飞车”，“卡巴斯基”有的人认为这些是导航类的查询，因为他们想访问由软件的开发者所维护的官方主页，他们认为通过官方主页下载最新的软件和补丁更加安全便捷。而有的则认为人这是信息类的查询，因为他们想通过搜索从多个网站而不是官方网站得到其他用户对该软件的评论，使用建议等。另外还有一些认为这是资源类的查询，因为他们认为只要一个网站能提供该软件的下载服务就足够了。

四、查询词的类型是可能发生变化的。在实际应用中，服务提供者可能根据他们的需要来解释不同查询的类型，而且随着网络中内容分布的变化，查询词的类型也会发生改变。例如，“躲猫猫”，“俯卧撑”，“打酱油”等词汇，它们在网络中的意思已经不是它们本身的意思了。

Kang 等[8]使用查询词分布（Dist.），交互信息（MI），查询在锚文本中的使用率（Anchor），词性分析（POS）和以上信息的结合，根据搜索目的对查询词进行分类。分类结果如表 3-2 所示：

表 3-2 查询根据搜索目的分类的结果，引自[8]

查询	训练集		测试集	
评价标准	准确率	召回率	准确率	召回率
Dist.	77.3%	38.7%	82.1%	28.2%
MI	90.9%	20.0%	78.2%	29.9%
Anchor	73.6%	35.3%	82.4%	35.9%
POS	100%	9.3%	96.4%	13.8%
所有	81.1%	57.3%	91.7%	61.5%

训练集由 TREC 2000 中主题相关任务的 50 个查询和 100 个随机选择的主页查询构成。测试集由 TREC 2001 中主题相关任务的 50 个查询和 145 个主页查询构成。作者在训练集上确定 4 种方法结合时，各种方法的权重。从上表的结果可以看到，尽管只对查询分成两类，但是分类的结果并不十分理想，尤其是召回率比较低。

KDD CUP 2005 的查询主题分类比赛中，比赛的主办者从 800000 个查询中随机选择 800 个作为测试集，并让 3 个人手工对这 800 个查询的类别进行标注，假设他们标注的结果分别为 L1，L2 和 L3。每个查询最多可以有 5 个有序的类别。表 3-3 展示了每个标注者以其他两个标注者的标注为标准进行分类的平均准确率和平均 F1 分数。这三个标注者的平均得分是 0.5 左右，这说明对查询按照主题进行分类，对人来说也是一件非常困难的事情[10]。

表 3-3 三个标注者按主题对查询分类，每个标注者以其他两个标注者为标准的分类准确率和 F1 分数，引自[10]

	L1	L2	L3	平均
F1	0.538	0.477	0.512	0.509
准确率	0.501	0.613	0.463	0.526

KDD CUP 2005 的查询主题分类比赛的冠军由 Shen[10]等夺得。他们利用搜索引擎对查询和类别同时进行扩展，以搜索引擎返回的相关网页，网页片段，网页标题来表示查询和类别。他们首先从“开放目录项目（Open Directory Project，http://dmoz.org）”得到查询词的间接类别，然后结合以下几种方法得到目标类别：使用 SVM 作为分类器，对间接类别和直接类别进行直接匹配，使用概率的方法将查询从间接类别映射到目标类别，分类的准确率和 F1 分数也仅仅是 0.465 和 0.461[10]。

TREC 2004 搜索比赛的数据集中总共有 225 个查询，这些查询被手工分成三类：主题提取（topic distillation），命名网页查找（named page finding）和主页查找（homepage finding）。这些查询还附有关联的文档和这些文档的标注。Geng 等[3] 将 TREC 2004 中的查询，按照 Song 等[39]的方法定义 8 类共 27 个特征，然后将查询表示成 27 维空间的点，最后使用主要成分分析（Principal component Analysis）将其压缩在 2 维空间中，如图 3-1 所示。

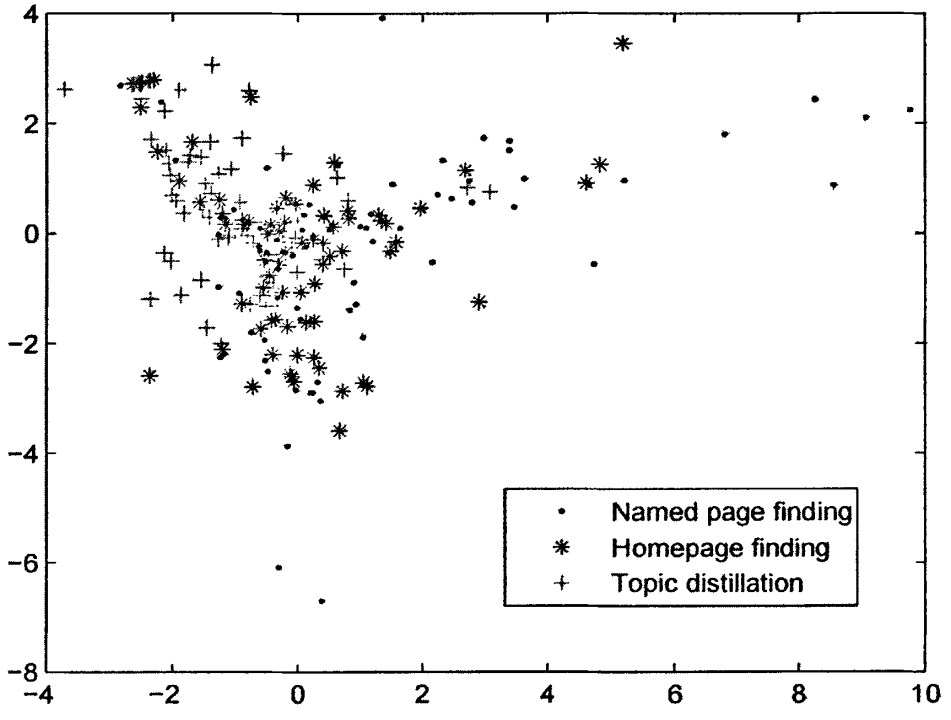


图 3-1 TREC 2004 中查询的分布，引自[3]

从图中可以看到，不同类别的查询混合在一起，我们很难在不同类型的查询之间划清界线。

3.4 依赖于查询的排序学习算法框架

在排序学习领域过去的研究中，大部分研究都是采用单一的模型来对所有查询进行排序，除了 Geng 等[3]使用 KNN 算法构建依赖于查询的排序模型。在 Geng 等[3]工作的基础上，本文提出一个更为通用的依赖于查询的排序学习算法框架。算法描述如图 3-2 所示。

预定义：

（1）定义查询特征的模型

（2）查询特征空间中查询相似性的度量模型

（3）选择相似查询的标准

（4）排序学习算法 *LEARN*

输入：

（1）训练集 $S=\{(q_i,\mathbf{d}_i,\mathbf{l}_i)\}_{i=1}^{m}$

（2）测试集 $T=\{(q_j,\mathbf{d}_j)\}_{j=1}^{n}$

输出：

排序的结果 $T'=\{(q_j,\boldsymbol{\pi}_j)\}_{j=1}^{n}$

1　定义训练集中查询的特征，将其表示成查询特征空间中的点，得到 $S'=\{\varphi(q_i)\}_{i=1}^{m}$

2　For j = 1, 2..., j

（1）　将测试查询 q_j 表示成查询特征空间的点 $\varphi(q_j)$

（2）　For i = 1, 2, ... , m

在查询特征空间中计算 q_j 与 q_i 的相似性 $Sim(q_j,q_i)$

End For

（3）　根据上一步计算中得到的相似性，以 C 为标准从训练集中选择与查询 q_j 相似的查询，组成集合 S_j

（4）以 S_j 作为训练集，使用排序学习算法 *LEARN* 进行训练，得到一个模型 h_j

（5）将 h_j 应用于与查询 q_j 关联的文档 $\mathbf{d}_j$ 得到文档的排列 $\boldsymbol{\pi}_j$

End For

图 3-2 依赖于查询的排序学习算法框架

依赖于查询的排序学习算法的核心思想是：针对对于查询进行“硬分类”

的排序模型的不足，提出对查询进行“软分类”的方法，也是聚类的方法：对训练集中的每一个查询定义特征向量，将其表示成查询特征空间的一个点，这一步我们可以在对数据进行预处理的时候完成。对于一个测试查询，本文使用同样的方法将其表示成查询特征空间的点，然后根据查询特征空间中预先定义的查询相似性计算方法和相似查询的标准，从训练集中选出与测试查询相似的查询，作为依赖于测试查询的训练集。与每个查询对应的训练集都是动态生成的，它们相互不同，都是整个训练集的一个子集。利用动态生成的训练集，我们可以使用任何排序学习的算法学习一个排序模型，例如 Pointwise 方法：IR 判别模型 [18]和 McRank[19]; Pairwise 方法：RankNet[20], FRank[21], RankBoost[22] 和 Ranking SVM[23]; Listwise 方法：SVM-MAP[24], AdaRankt[25], RankCosine[27], ListNet[28]和 ListMLE[29]等。最后用训练得到的模型应用于测试查询，得到与测试查询相关联的文档的排序。这种基于聚类的方法既利用了相似查询的有用信息，也避免了不相似查询的负面影响。

训练集 S 中的每一个查询 q_i 都有与之关联的文档列表 $\mathbf{d}_i=\{d_{i1},d_{i2},...,d_{in(q_i)}\}$ 和一个相关性标注的列表 $\mathbf{l}_i=\{l_{i1},l_{i2},...,l_{in(q_i)}\}$，$n(q_i)$ 表示列表 $\mathbf{d}_i$ 和 $\mathbf{l}_i$ 的大小，d_{ij} 表示 $\mathbf{d}_i$ 的第 j 个文档，l_{ij} 表示文档 d_{ij} 的相关性标注。与训练集不同，测试集 T 中的查询 q_j 只有关联的文档列表 $\mathbf{d}_j=\{d_{j1},d_{j2},...,d_{jn(q_j)}\}$，我们并不知道每个文档的相关性标注。我们进行排序的目的，就是要对与 q_j 关联的文档进行排序，得到 $\mathbf{d}_j$ 的一个排序 $\boldsymbol{\pi}_j$。

查询的聚类是算法的关键，它包括：定义查询特征的模型 φ，查询特征空间中查询相似性的度量模型 Sim 和选择相似查询的标准 C。如何定义查询的特征，过去有比较多的研究[5, 8, 9, 10, 39]。Song 等[39]首先提出使用与查询关联的文档的特征来表示查询的特征，他们使用查询词长度，反向文档频率，以及关联文档的 BM25 分数，URL 信息等 8 类特征来定义查询的特征向量。Geng 等[3]使用以下启发式方法来定义查询的特征向量：以 BM25 分数为标准，在与查询关联的文档中选出前 T 个，对于文档的每一个特征，取这 T 个文档该特征的平均值作为查询的一个特征值。例如，如果文档的一个特征是 tf-idf，那么查询的

相应特征是排在前面的T个文档的tf-idf的平均值。Geng等[3]使用的是查询特征空间中的欧氏距离来表示查询的相似性，而相似查询的标准则选择了K近邻（KNN）算法，也就是说，他们从欧氏空间中选择K个与测试查询最近邻的训练查询构建训练集。

3.5 查询聚类

查询的聚类是算法的关键，本节详细介绍了本文提出的查询聚类方法。

3.5.1 定义查询特征

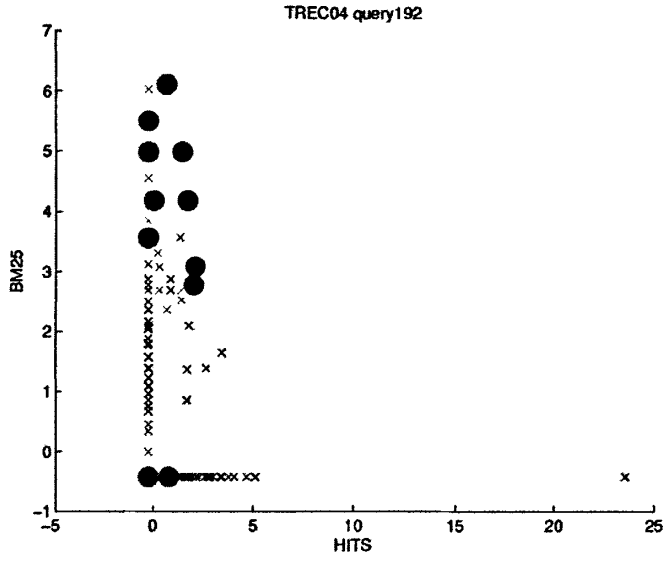
对于训练集和测试集中的查询，我们都知道与之关联的文档列表。我们使用这些文档的特征来表示查询的特征。每一个查询-文档对(q_i, d_{ij})可以用一个特征向量$\phi(q_i, d_{ij})$来表示，向量的每一维表示一个特征（例如tf，idf，BM25，PageRank等）。每个查询都有多个关联的文档，每个文档都可以表示成一个向量，这样一个查询可以表示成一个矩阵，矩阵的行表示文档，列表示特征，如图3-3所示。

查询q

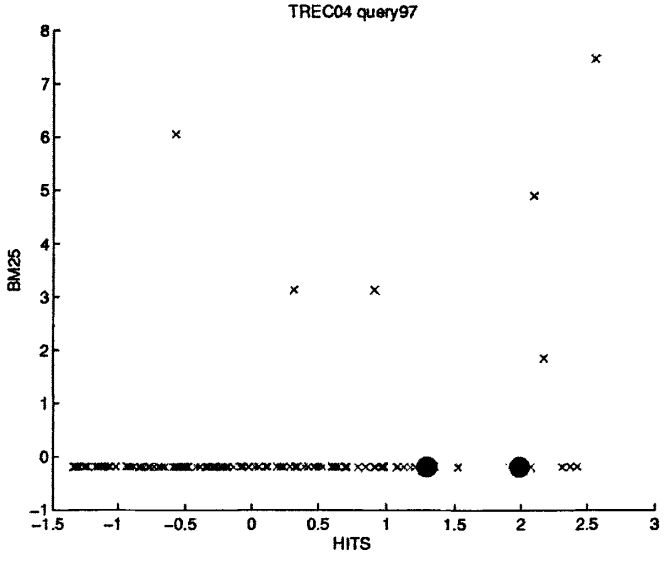
文档	特征1	特征2	…… ……	特征n
d_1	25.614140	29.000000	…… ……	23.231527
d_2	25.941042	257.000000	…… ……	24.116287
d_3	26.470301	137.000000	…… ……	29.489431
…	……	……	…… ……	……
…	……	……	…… ……	……
…	……	……	…… ……	……
d_n	19.148941	1139.00000	…… ……	0.000000

图 3-3 查询的矩阵表示

这样，每个查询都有多个关联的特征。Duh等[4]的研究发现，不同的特征对于不同查询的区分度是不一样的。如图3-4所示。



(a)



(b)

图 3-4 不同的特征对于不同查询的区分度不一样，引自[4]

图3-4表示了TREC 2004中的两个查询。图中的点表示与查询关联的文档，x轴表示HITS Hub分数，y轴表示标题的BM25分数，这两个特征都是网页排序的重要特征。与查询相关的文档用红色的点表示，不相关的文档用蓝色的叉表示。从图中可以看到，对于图 a 表示的查询，y 轴上数据的变化比较大，x

轴上数据的变化比较小，也就是说，特征BM25的区分度比较大；对于图b表示的查询，x轴上数据的变化比较大，y轴上数据的变化比较小，也就是说，特征HITS Hub的区分度比较大。

因为不同的特征对于不同查询的区分度不一样，所以本文用特征的区分度来表示查询。与查询关联的文档可以表示成一个矩阵，对于矩阵的每一列，本文采用近似的方法，用该列的标准差表示这一列的区分度，也就是这一列对应的特征的区分度，标准差的计算如下：

$$std=\sqrt{\frac{\sum_{i=1}^{n}\left(s_i-\bar{s}\right)^2}{n}} \tag{3-1}$$

std是标准差，s_i是第i个文档的特征值，s_i是所有文档该特征的平均值。这样，每个查询都可以用一个特征向量表示，向量的每一维是一个特征的标准差，查询q表示成：

$$q=(std(f1),std(f2),\cdots,std(fn)) \tag{3-2}$$

$f1,f2,\cdots,fn$表示特征，$std()$表示标准差。

本文在计算标准差的时候，并不是在原始数据上计算。从图3-3可以看出，如果某一列的特征值普遍比较大，那么该列的标准差可能会比较大，即使这一列的区分度并不大，所以在原始数据上直接计算标准差并不是表示区分度一种很合适的表示方法。在计算标准差时，先对数据进行预处理，对特征矩阵进行正则化（normalization），使每个特征的特征值都在0到1之间，正则化的方法如下：

假设X是一个查询原始的特征矩阵，$X(i,j)$是这个查询第i个关联文档的第j个特征。

假设Y是正则化后的特征矩阵：

$$Y(i,j)=[X(i,j)-\min\_\{i\}\{X(i,j)\}]/[\max\_\{i\}\{X(i,j)\}-\min\_\{i\}\{X(i,j)\}]$$

如果$\max_\{i\}\{X(i,j)\}=\min_\{i\}\{X(i,j)\}$，那么$Y(i,j)=0$

$\max_\{i\}\{X(i,j)\}$和$\min_\{i\}\{X(i,j)\}$分别表示矩阵第j列的最大值和最小

值。

特征值经过正则化处理后计算得到标准差，能够比较恰当地表示特征的区分度。图 3-3 中矩阵经过正则化后，如图 3-5 所示，这里假设特征 1，特征 2 和特征 n 的最大，最小值都在图中出现。

查询 q

文档	特征 1	特征 2	…… ……	特征 n
d_1	0.883060	0.000000	…… ……	0.787792
d_2	0.927710	0.205405	…… ……	0.817794
d_3	1.000000	0.097297	…… ……	1.000000
…	……	……	…… ……	……
…	……	……	…… ……	……
…	……	……	…… ……	……
d_n	0.000000	1.000000	…… ……	0.000000

图 3-5 查询的矩阵表示（正则化后）

3.5.2 查询的相似性

用查询特征的区分度来表示查询以后，可以使用欧氏距离来表示两个查询的差异：

$$Dis(q_i,q_j)=\sqrt{\sum_{k=1}^{n}(f_k(q_i)-f_k(q_j))^2} \qquad (3\text{-}3)$$

查询的相似性用$1-Dis(q_i,q_j)$表示，$Dis(q_i,q_j)$表示q_i和q_j的距离，$f_k(q_i)$表示查询q_i第k个特征的值，n是表示查询的特征个数。

除了利用欧氏距离来表示两个查询的相似性以外，本文提出一种新颖的方法来度量查询之间的相似性：

1、对于任意一个查询，按照它各个特征的特征值大小进行排序，得到一个特征的排序（排序中的元素为特征的序号）。

2、对于任意两个查询，用它们对应的特征排序的相似性来表示这两个查询的相似性。

例如，有两个查询q_i和q_j：

$q_i=(0.3,1.2,0.7)$

$q_j=(1.2,0.8,1.6)$

对于查询q_i，因为第2个特征值最大，第3个特征值次之，第1个特征值最小，对特征值排序后得到的特征排序为$<2,3,1>$。同理，对查询q_j的特征值排序后得到的特征排序为$<3,1,2>$。

将查询表示为特征排序后，本文用特征排序的相似性来表示两个查询的相似性。衡量两个排序相似性的方法有很多种[40]，例如Spearman的footrule F，correlation R，还有Kendall的τ [41, 42]。本文选择Kendall的τ来表示排序的相似性，任意两个查询q_i和q_j的相似性可以如下计算：

$$Sim(q_i,q_j)=Sim(r1,r2)=\frac{\#\{(f_s,f_t)\in F \mid f_s \prec_{r1} f_t \ and \ f_s \prec_{r2} f_t\}}{\#\{(f_s,f_t)\in F\}} \qquad (3\text{-}4)$$

公式中#表示集合中元素的个数。$r1$和$r2$和表示查询q_i和q_j对应的特征排序。F表示查询所有特征中任意两个特征组成的特征对的集合，假设查询有n个特征，那么$\#\{(f_s,f_t)\in F\}=n(n-1)/2$。$f_s \prec_{r1} f_t$表示在排序$r1$中，特征$f_s$排在特征$f_t$后面，$f_s \prec_{r2} f_t$表示在排序$r2$中，特征$f_s$排在特征$f_t$后面。$\#\{(f_s,f_t)\in F \mid f_s \prec_{r1} f_t \ and \ f_s \prec_{r2} f_t\}$表示$r1$和$r2$公共的特征对数目。我们可以看出，$Sim(q_i,q_j)=Sim(q_j,q_i)$。

对于上面的例子，q_i和q_j对应的特征排序中，共同的特征对只有$(1,3)$，特征对的总数为$3\times(3-2)/2=3$，所以$Sim(q_i,q_j)=1/3$。

3.5.3 选择相似查询的标准

对于一个待测试的查询，本文从训练集中找出与之相似的查询作为训练集，来训练相应的模型。在计算待测试查询与训练集中所有查询之间的相似性

以后，选择相似查询的标准可以有多种。Geng 等[3]使用 KNN 算法来选择相似的查询，也就是说，从训练集中选出固定的 k 个最接近的邻居来构造训练集，如图 3-6 所示。

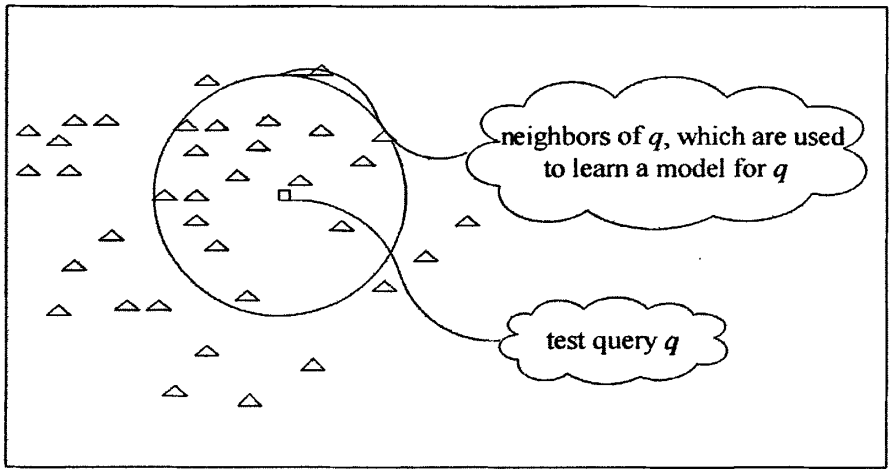


图 3-6 选择相似的查询构建训练集，引自[3]

图中正方形表示测试查询 q，三角形表示训练集中的查询，以测试查询 q 为圆心画一个圆，被圆圈包围的查询共有 k 个，它们是与测试查询最相似的 k 个查询，用这 k 个查询构建依赖于 q 的的训练集，然后用来训练模型，对 q 进行测试。

除了可以像 KNN 算法那样固定查询的个数之外，我们还可以对测试查询与训练查询的距离设定一个域值，小于或等于域值的训练查询我们认为是相似的，将其添加到依赖于测试查询的动态训练集中去。也就是说，图 3-6 中圆的半径是固定的，算法如图 3-7 所示。

对于 KNN 算法，在训练集中选择的查询个数是固定的，图 3-6 中圆的半径是不确定的，对于不同的查询半径是不同的，如果与测试查询接近的查询比较多，也就是说，如果测试查询落在训练集中查询比较稠密的地方，则圆的半径比较小，否则半径会比较大。对于固定范围的算法，图 3-6 中圆的半径是固定的，在训练集中选择的查询个数是不确定的。对于不同的查询，动态生成的训练集的大小是不同的，如果与测试查询相似的查询比较多，则动态生成的训练集包含的查询比较多，否则包含的查询比较少。

输入：

（1） 训练集 $S=\{q_i\}_{i=1}^{m}$

（2） 测试查询 q_j

（3） 距离的域值 u

输出：依赖于查询 q_j 的训练集 S_j

初始化： $S_j=\phi$

For i = 1, 2, … , m

计算 q_j 与 q_i 的相似性 $Sim(\varphi(q_j),\varphi(q_i))$

If($Sim(q_j,q_i)$ <= u)

$S_j \cup = q_i$

End If

End For

Return S_j

图 3-7 固定范围的相似查询选择算法

假设训练集有 m 个查询，KNN 算法在计算测试查询与训练集查询的相似性时，可以动态维护一个有 k 个元素的小顶堆，堆中保存已经计算的训练查询中，与测试查询最相似的 k 个查询，这样 KNN 算法的时间复杂度是 $O(m\log m)$。对于固定范围的算法，每计算一个训练查询与测试查询的相似性后，我们就可以判断是否将它添加到依赖于测试查询的训练集，算法的时间复杂度是 $O(m)$。

3.6 依赖于查询的排序学习算法实现

根据图 3-2 中“依赖于查询的排序学习算法框架”以及上一节的查询聚类方法，本文实现了两种依赖于查询的排序学习算法，分别称为“特征排序+RADIUS 和“特征排序+KNN”，介绍如下：

对于训练集和测试集的每一个查询，本文使用 3.5.1 小节的方法，将查询表示成一个特征区分度的向量。然后对这些特征的区分度进行排序，得到一个特征的排序，这一步称为“特征排序”。对于每一个测试查询，我们在训练集中寻找与之相似的查询，查询的相似性用 3.5.2 小节中的方法计算。然后，在训练集中，本文分别使用了 KNN 算法和固定距离的算法，选择与测试查询相似的查询建立训练集。确定训练集以后，本文使用 SVM-MAP[24]学习一个排序模型。最后将学习得到的模型应用于测试查询，得到与测试查询相关联的文档的排序。“特征排序+RADIUS”算法如图 3-8 所示。

输入：

（1）训练集 $S=\{(q_i,\mathbf{d}_i,\mathbf{l}_i)\}_{i=1}^{m}$

（2）测试查询 q 和与之关联的文档

（3）距离的域值 r

输出：

查询 q 的文档排序结果 $\boldsymbol{\pi}$

预处理：

对于训练集中的每个查询 q_i，计算与之关联的文档的每个特征的标准差，用标准差近似作为特征的区分度来表示查询

算法过程：

（1）计算与 q 关联的文档的每个特征的标准差，用标准差近似作为特征的区分度来表示查询 q

（2）将 q 和训练集中的所有查询表示成特征的排序，计算 q 的特征排序与训练集中所有查询的特征排序的相似度，并找出与 q 的距离小于 r 的查询，组成训练集 S_q

（3）以 S_q 为训练集，使用 SVM-MAP 训练一个模型 h

（4）将 h 应用于与查询 q 关联的文档，得到文档的排列 $\boldsymbol{\pi}$

图 3-8 “特征排序+RADIUS”算法

“特征排序+KNN”算法与“特征排序+RADIUS”算法类似，只需要在图3-8的算法中进行如下修改：

将输入（3）修改为“选择相似查询的个数k”。

将算法过程的第（2）步修改为“计算q的特征排序与训练集中所有查询的特征排序的相似度，并找出与q最相似的k个查询，组成训练集S_q”。

3.7 缓存策略

对于依赖于查询的排序学习，给定一个查询之后，我们要实时训练一个模型，然后用于测试。这对于实际应用是不能接受的，因为训练的时候一般会比较长。对于使用KNN作为动态构建训练集的算法，Geng等[3]提出了两种缓存策略，将学习的过程放到离线进行。我们稍作推广，也可以将其应用于使用固定半径选择测试查询的算法。

3.7.1 离线算法1

在训练的时候，对于训练集中的每一个查询q_i，我们使用KNN算法或者固定范围的算法找出与之相似的查询，构成集合S_i。然后我们使用S_i预先离线地训练模型h_i。

在测试的时候，对于测试查询q_j，我们用训练时使用的相似查询选择标准，找出与之相似的查询，构成集合S_j。然后我们将S_j和每一个$S_i, i=1,\ldots,m$比较，从中找出与S_j最相似的集合S_i。接着使用h_i来对测试查询进行排序，如图3-9所示。

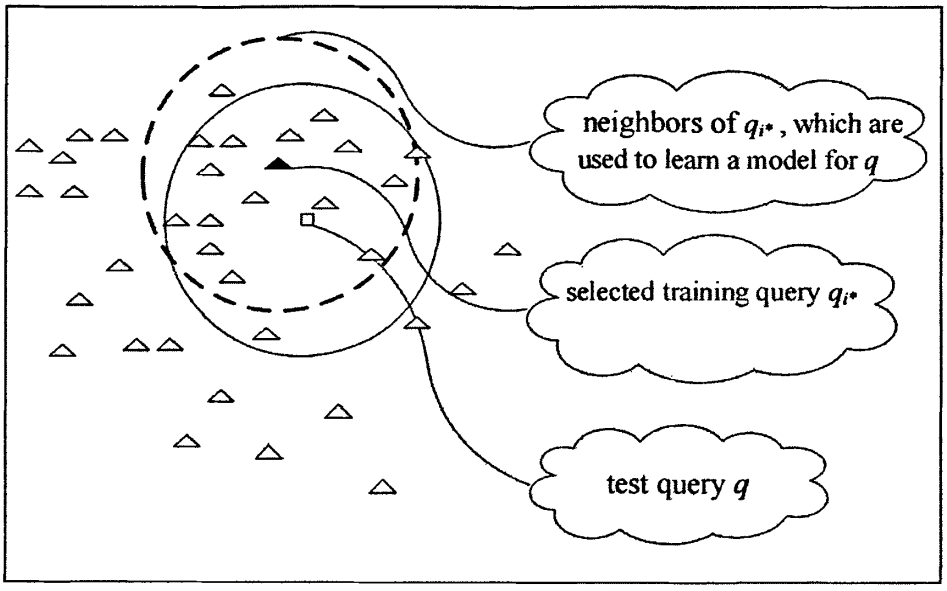


图 3-9 离线算法 1，引自[3]

比较两个集合相似性的方法有多种[43]。如果使用 KNN 算法选择相似查询，因为不同集合的元素个数都是 k 个，我们只要选择共同元素最多的集合就可以了。如果使用固定范围的算法选择相似查询，那么任意两个集合的大小可能是不同的，我们使用以下公式来计算两个集合的相似性：

$$sim(S_i, S_j) = 2|S_i \cap S_j| / (|S_i| + |S_j|) \qquad (3\text{-}5)$$

图 3-8 中的正方形表示测试查询 q，三角形表示训练查询。实线的圆圈包围的是和测试查询相似的查询，实心三角形表示被选中的查询 q_{i^*}，虚线的圆圈包围的是和 q_{i^*} 相似的查询。用虚线中的查询训练得到的模型被用于对测试查询的排序。

3.7.2 离线算法 2

对于离线算法 1，我们需要寻找“最相似”的训练集，这一步的时间复杂度是 $O(m)$。另外，我们也需要找出测试查询的邻居，对于 KNN 算法，时间复杂度是 $O(m \log m)$；对于固定范围的算法，时间复杂度是 $O(m)$。在实际应用中，

响应时间是非常重要的，Geng 等[3]提了更高效的离线算法 2。它的思想是：我们只找出与测试查询最相似的查询，而不是 k 个相似的查询或者与之距离小于某一域值的所有查询，如图 3-10 所示。

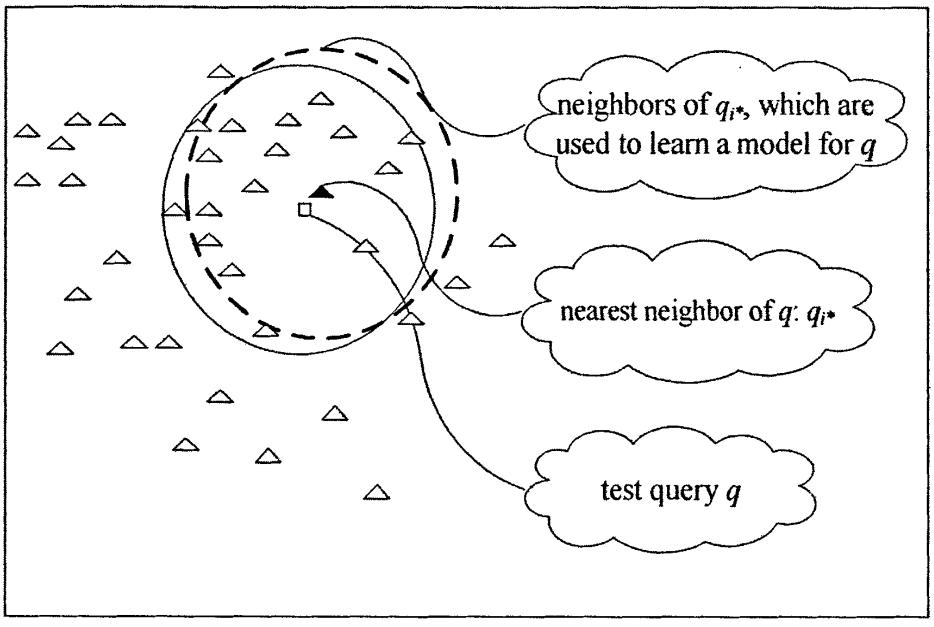


图 3-10 离线算法 2，引自[3]

与离线算法 1 一样，在训练的时候，对于训练集中的每一个查询 q_i，我们都找出与之相似的查询，然后利用这些查询预先离线地训练模型 h_i。

在测试的时候，对于测试查询 q_j，我们从训练集中找出与之最相似的查询，假设是 q_{i*}，则我们直接使用模型 h_{i*} 对 q_j 进行排序。

第4章 实验结果及分析

本章通过实验数据表明本文提出的依赖于查询的排序学习的有效性。本章首先介绍实验的数据集，然后介绍实验结果的评价标准，包括MAP和NDCG。在上一章中，本文提出了查询相似性的度量方法和两种选择相似查询的标准，本章分别以这两种选择相似查询的标准进行实验，并与使用单一模型进行排序的结果进行对比以及和Geng等[3]使用KNN算法进行查询聚类的方法比较。本章还讨论算法的有效性。

4.1 实验数据集

本文在LETOR数据集[44]上进行实验。LETOR是排序学习研究领域的基准数据集，它由微软亚洲研究院发布，包括3个文档检索的数据子集：TD2003，TD2004和OHSUMED。LETOR还提供了评价工具和一些作为基线的评价结果（如Ranking SVM[23]，RankBoost[22]，Frank[21]，ListNet[28]，AdaRank[25]等）。每一个子集都包含一个集合的查询，查询-文档对的特征和相应的相关性标注。每一个查询-文档对都表示成一个特征向量和它们的相关性标注。

4.1.1 OHSUMED数据集

OHSUMED数据集是MEDLINE的子集，MEDLINE是一个医学出版物的数据库。该数据集包括来自于1987至1991年间270本医学杂志的348566条记录。每一条记录包括标题，摘要，作者，来源，出版物类型等。

OHSUMED共有106个查询，对于每个查询，平均有152.3个关联的文档，其中相关的文档平均有28个。一个查询是一个关于医学方面的检索需要，因此它与患者信息和主题信息有关。文档与查询的相似性由人工判断，共有三个级别：完全相关（definitely relevant），可能相关（possibly relevant）和不相关（not relevant）。相关的文档对共有16140个。

OHSUMED数据集总共抽取了25个特征（10个来自标题，10个来自摘要，5个来自“标题+摘要”），包括低级特征和高级特征。低级特征包括词频（tf），反向文档频率（idf），文档长度（dl）和它们的结合。高级特征包括BM25和

LMIR 分数等。

4.1.2 TD2003 和 TD2004 数据集

在 TREC 2003 和 TREC 2004 中，有一个专门针对信息检索的网页比赛。这个比赛的目的是：研究在一个像万维网那样的庞大的超链接结构文档库中搜索时，人们的检索行为。这个网页比赛使用“.gov”数据集，数据集中总共有 1053110 个 html 文档，加上 11164829 个超链接。

主题提取是网页比赛的其中一个任务，它的目的是找到与主题相关的网页入口点列表。比赛关注的是返回好网站的入口页面，而不是包含相关信息的页面本身。因为入口页面提供了数据集中覆盖主题的一个更好的纵览。

LETOR 中的 TD2003 和 TD2004 数据集来自 TREC 2003 和 TREC 2004 的主题提取任务。TD2003 有 50 个查询，TD2004 有 75 个查询。TD2003 中，每个查询平均有 983.4 个关联的文档，其中相关的文档平均有 1 个；TD2004 中，每个查询平均有 988.9 个关联的文档，其中相关的文档平均有 0.6 个。每一个查询-文档对都有一个手工的二元相关性标注：相关（relevant）和不相关（irrelevant）。

TD2003 和 TD2004 数据集总共抽取了 44 个特征，这些特征可以分为四类：

（1）低级内容特征。这类特征包括词频（tf），反向文档频率（idf），文档长度（dl）和它们的结合（例如 tf*idf）。对于一个网页，这些特征中的每一个都有 4 个值，分别与网页的主体，标题，锚和 URL 对应。

（2）高级内容特征。这类特征包括 BM25 和 LMIR 算法的输出。整个文档，锚文本，标题和抽取的标题都有对应的 BM25 特征。对于 LMIR，使用了不同的平滑方法（DIR，JM，ABS）和三个域（锚，标题，抽取的标题），总共有 9 个 LMIR 的特征。

（3）超链接特征。这类特征包括 PageRank，HITS 和它们的变种（HostRank，主题 PageRank 和主题 HITS）。HITS 和主题 HITS 都有权威性和中心两个分数，所以总共有 7 个超链接特征。

（4）混合特征。这类特征包含内容和超链接信息，有“基于超链接的相关性繁殖”和“基于网站地图的相关性繁殖”两种。

4.1.3 交叉验证

为了避免过拟合问题，LETOR 采用五折交叉验证（five-fold cross validation）。每个数据集都被平均分成五部分，实验一共进行五组，对于每一组，使用其中三部分做训练集，一部分做校验集，剩下的一部分作测试集。训练集用于学习排序模型，校验集用于调整参数（例如 RankBoost 的迭代次数），测试集用于评价排序模型的表现。对于机器学习，测试集上数据的分布是未知的，我们不能在测试集上直接选择参数，因为这样会导致过拟合，影响模型的可扩展性。为了使实验结果更有说服力，我们用检验集上数据的分布来模拟测试集上数据的分布，训练得到的模型首先在校验集上进行测试，在校验集上取得最优结果的那组参数，再用于测试集进行测试。最终的实验结果是五个测试集上平均的结果。

4.2 评价尺度

在信息检索中，广泛使用的文档与查询之间相关性的判断标准有三种[46]：

（1）Pointwise。评价单个文档的相关性。每个文档的相关性可以是二元的：相关（Relevant），不相关（Irrelevant）。也可以是多元的：完全相关（Perfect），很相关（Excellent），相关（Good），比较相关（Fair），不相关（Bad）。

（2）Pairwise。使用文档对的偏好，而不是每个文档绝对的相关性。例如，我们可以说对于查询 q，文档 A 比文档 B 更相关。

（3）Listwise。文档之间的偏序或者全序关系。可以从用户的点击日志中挖掘。

评价一个排序好坏的尺度有多种，在信息检索中广泛使用的有 MAP 和 NDCG，分别介绍如下：

4.2.1 MAP

对于一个查询，位置 n 的精度 P@n 度量一个排序结果中，前 n 个文档的相关性，它的定义如下：

$$P@n=\frac{\#\{\text{前}n\text{个文档中相关的文档}\}}{n} \tag{4-1}$$

例如，如果一个排序结果的前 8 个文档分别是{相关，不相关，相关，不相关，相关，不相关，相关，不相关}，那么 P@1 到 P@8 的值为{1，1/2，2/3，2/4，3/5，3/6，4/7，4/8}。

对于多个查询，通过计算所有查询 P@n 的平均值可以得到平均的 P@n 值。P@n 是针对二元的相关性判断标准，对于 OHSUMED 数据集，我们可以简单认为“完全相关”是文档是相关的，其它的是不相关的。

对于一个查询而言，平均精度（AP）定义为所有相关文档对应位置的 P@n 值的平均值：

$$AP=\frac{\sum_{n=1}^{N}(P@n*rel(n))}{\#\{\text{相关的文档}\}} \tag{4-2}$$

N 是检索到的文档的总数，$rel(n)$是一个二元函数，它判断第 n 个位置的文档的相关性：

$$rel(n)=\begin{cases}1,\text{如果第}n\text{个文档是相关的}\\0,\text{否则}\end{cases}$$

MAP 是指多个查询 AP 的平均值。

假设有两个查询，它们对应的排序结果分别是{相关，不相关，相关，不相关，相关}，{相关，不相关，不相关，相关}。对于第一个查询，$AP=(1+2/3+3/5)/3\approx 0.76$；对于第二个查询，$AP=(1+2/4)/2=0.75$。这两个查询组成的集合的 $MAP=(0.76+0.75)/2=0.755$。

4.2.2 NDCG

MAP 只能处理二元相关性：“相关”或者“不相关”的情况。最后，有人提出新的评价标准：NDCG（Normalized Discount Cumulative Gain）[28, 47]，它可以处理多元相关性的情况。它遵循两条基本原则：

（1）相关性高的文档比相关性低的文档更有价值。

（2）相关的文档排得越后，它对用户而言价值越低，因为用户访问它的

可能比较低。

对于多元的相关性判断标准，每个文档都有它的相关性分数，第 i 个文档的相关性分数可以如下定义：

$$r(i)=\begin{cases}5, & \text{第i个文档与查询完全相关}(Perfect)\\ 4, & \text{第i个文档与查询很相关}(Excellent)\\ 3, & \text{第i个文档与查询相关}(Good)\\ 2, & \text{第i个文档与查询一般相关}(Fair)\\ 0, & \text{第i个文档与查询不相关}(Bad)\end{cases}$$

在一个排序中，每个文档的相关性分数可以用来衡量它所在位置的增益（Gain），增益一般用 $2^{r(i)}-1$ 计算，这些增益可以从位置 1 一直累加到位置 n，得到累计增益（Cumulated Gain）。如果位置 i 的增益用 CG[i]表示，那么累计增益可以递归地定义为：

$$CG[i]=\begin{cases}G[1], & if\ i=1\\ CG[i-1]+G[i], & otherwise\end{cases} \tag{4-3}$$

用户关注的是排名靠前的文档，因此排名靠后的文档对累计增益的贡献应该比排名靠前的文档要小。NDCG 中的 D（Discount）表示位置的折扣因子，位置越靠后，折扣因子越大，它的计算公式是：$1/\log(1+position)$。

在累计增益中，每个文档的增益都乘以折扣因子后，得到折扣的累计增益（DCG）：

$$DCG[i]=\begin{cases}G[1], & if\ i=1\\ DCG[i-1]+G[i]/\log(1+i), & otherwise\end{cases} \tag{4-4}$$

随着位置的递增，文档增益对累计增益的贡献越来越小。

NDCG 中的 N（Normalized）表示正则化因子 Z_n，它是一个完美的排序的 DCG 值，也就是在位置 n，DCG 的最大值。

综上所述，NDCG 可以用以下公式表示：

$$NDCG@n=\frac{\sum_{i=1}^{n}(2^{r(i)}-1)/\log(1+i)}{Z_n} \tag{4-5}$$

与 MAP 一样，对于多个查询组成的测试集，它的 NDCG@n 值为所有查

询 NDCG@n 的平均值。

如果一个排序结果的前 4 个文档分别是{完全相关，一般相关，非常相关，非常相关}，那么每个文档的增益为{2^5-1=31，2^2-1=3，2^4-1=15，2^4-1=15}，每个位置的累计增益依次为{31，31+3=34，34+15=49，49+15=64}，折扣的累计增益为{31*1=31，31+3*0.63=32.9，32.9+15*0.5=40.4，40.4+15*0.43=46.9}。按照文档的相关性进行排序，得到完美的排序为{完全相关，非常相关，非常相关，一般相关}，每个位置的正则化因子为{31，31+15*0.63=40.5，40.5+15*0.5=48，48+3*0.43=49.3}。因此，NDCG@n 的值分别为{31/31=1，32.9/40.5=0.81，40.4/48=0.84，46.9/49.3=0.95}。

对于 TD2003 和 TD2004 数据集，相关性分数{0，1}对应“不相关文档”和“相关文档”。对于 OHSUMED 数据集，相关性分数{0，1，2}对应“不相关文档”，“部分相关文档”和“相关文档”[44]。

4.3 实验结果与分析

本文对 3.6 小节中提出的两种算法：“特征排序+RADIUS”和“特征排序+KNN”，在 LETOR 上进行了实验。在实验中，本文使用“单一模型”作为基线（baseline），“单一模型”是指利用训练集中的全体查询训练一个模型，然后用于测试集上的所有测试。此外，作为比较，本文还实现了 Geng 等[3]的算法，称为“特征平均+KNN”，他们使用了与查询关联文档的特征的平均值来表示查询，然后使用欧氏距离来表示查询的相似性，最后使用 KNN 算法来选择训练集。

4.3.1 实验设置

一般情况下，特征 idf 仅与文档库中包含查询词的文件数目有关，对于所有与查询关联的文档，它的值都是一样的，所以这个特征的区分度为 0。但是，在TD2003 和 TD2004 数据集中，有些查询通过查询扩展的办法，找到一些与查询相关的文档，这些文档并没有包含查询词，所以这些文档的 idf 值与其它包含查询词的文档的 idf 值是不一样的，这就造成了这个查询的 idf 特征的区分

度不为 0 的情况，这种情况在 OUSHUMED 数据集中并没有出现。本文在 TD2003 和 TD2004 数据集上进行特征排序时，忽略了所有与 idf 相关的特征（第 9 至 12 列）。因为 OHUSMED 数据集并没有出现 idf 值不为 0 的情况，所以本文在特征排序时没有忽略这些特征，如果忽略这些特征，实验结果也不会发生变化。

在对查询的特征区分度进行排序之前，本文对这些特征的区分度进行正则化。我们首先将一个查询表示成一个特征的向量，然后训练集的所有查询和测试可以构成一个矩阵。我们对矩阵的列进行正规化，使得该列的最大值为 1，最小值为 0，其它值在 0 和 1 之间。正规化的方法与表示查询的特征矩阵的正则化一样。

在实验中，对于所有四种算法，本文都使用 SVM-MAP[24]作为基本的训练算法。本文使用 SVM-MAP 的理由是：它是一种 Listwise 的方法，与 Pointwise 和 Pairwise 方法不同，它是将一个查询看成一个训练样本，所以更能体现“依赖于查询”的特征。SVM-MAP 只有一个参数 C，表示最大间隔和错误率的折中。对于所有四种算法，本文都将 C 值设置为 10。

“特征排序+KNN”和“特征平均+KNN”都有一个参数 k，表示在构建依赖于测试查询的训练集时，选择的相似查询的个数。“特征排序+RADIUS”有一个参数 r，表示选择相似查询的域值，与测试查询的距离小于等于 r 的查询构成依赖于测试查询的训练集。本文按照 LETOR 的指引，使用 5 折交叉校验，在校验集上选择参数。因为 SVM-MAP 是直接优化评价尺度 MAP 的，所以本文选择参数的标准是：使得校验集的测试结果中，MAP 值最大的那组参数。

“特征排序+KNN”和“特征平均+KNN”算法中，本文在校验集上选择 k 的范围是 $0.3N$~$0.8N$，每次递增 $0.1N$，N 是训练集的大小。对于 TD2003 数据集，k 的选择范围是 9~24，间隔为 3；对于 TD2004 数据集，k 的选择范围是 15~35，间隔为 5；对于 OHSUMED 数据集，k 的选择范围是 18~48，间隔为 6。本文这样选择 k 的理由是：如果选择的查询太少，则训练集提供给学习算法的信息是不足够的；如果选择的查询太多，则和使用单一模型进行训练差别不大，没必要使用依赖于查询的排序学习。在后面的实验中，本文会用不同的 k 值在测试集上进行实验，该实验验证了本文这样选择参数的理由。

“特征排序+RADIUS”算法中，参数 r 的选择比较困难，因为对于不同的数据集，r 的变化范围差别比较大。本文根据 r 在不同数据集上的变化情况，确定参数的选择范围。对于 TD2003 数据集，r 的选择范围是 0.4~0.55，间隔为 0.05；对于 TD2004 数据集，r 的选择范围是 0.3~0.5，间隔为 0.05；对于 OHSUMED 数据集，r 的选择范围是 0.2~0.4，间隔为 0.05。

4.3.2 与基线比较的实验结果

表 4-1，表 4-2，表 4-3 分别列出了四种算法（单一模型，特征平均+KNN，特征排序+RADIUS，特征排序+KNN）在 TD2003，TD2004 和 OHSUMED 数据集上的实验结果。表中括号内的数字表示依赖于查询的算法比使用单一模型算法的改进百分比。

表 4-1 四种算法（单一模型，特征平均+KNN，特征排序+RADIUS，特征排序+KNN）在 TD2003 数据集中的实验结果（MAP 和 NDCG@n）

	MAP	NDCG@1	NDCG@3	NDCG@5	NDCG@10
单一模型	0.114	0.280	0.180	0.174	0.175
特征平均+KNN	0.119 (4.39%)	0.220 (-21.40%)	0.172 (-4.44%)	0.171 (-1.72%)	0.184 (5.14%)
特征排序 +RADIUS	0.163 (42.98%)	0.300 (7.14%)	0.261 (45.00%)	0.241 (38.51%)	0.233 (33.14%)
特征排序+KNN	0.145 (27.19%)	0.280 (0.00%)	0.212 (17.78%)	0.2 (14.92%)	0.199 (13.71%)

表 4-2 四种算法（单一模型，特征平均+KNN，特征排序+RADIUS，特征排序+KNN）在 TD2004 数据集中的实验结果（MAP 和 NDCG@n）

	MAP	NDCG@1	NDCG@3	NDCG@5	NDCG@10
单一模型	0.312	0.387	0.345	0.348	0.382
特征平均+KNN	0.319 (2.24%)	0.373 (-3.62%)	0.362 (4.93%)	0.361 (3.74%)	0.391 (2.36%)
特征排序 +RADIUS	0.321 (2.88%)	0.387 (0.00%)	0.344 (-0.29%)	0.343 (-1.44%)	0.388 (1.57%)
特征排序+KNN	0.321 (2.88%)	0.400 (3.36%)	0.375 (8.70%)	0.375 (7.76%)	0.397 (3.93%)

表 4-3 四种算法（单一模型，特征平均+KNN，特征排序+RADIUS，特征排序+KNN）在 OHSUMED 数据集中的实验结果（MAP 和 NDCG@n）

	MAP	NDCG@1	NDCG@3	NDCG@5	NDCG@10
单一模型	0.414	0.452	0.415	0.385	0.369
特征平均+KNN	0.425 (2.66%)	0.476 (5.31%)	0.413 (-0.48%)	0.399 (3.63%)	0.387 (4.88%)
特征排序 +RADIUS	0.421 (1.69%)	0.47 (3.98%)	0.423 (1.93%)	0.397 (3.12%)	0.384 (4.07%)
特征排序+KNN	0.419 (1.21%)	0.472 (4.42%)	0.431 (3.86%)	0.411 (6.75%)	0.389 (5.42%)

从表中我们可以看到：

（1）在 TD2003 数据集中，本文提出的两种依赖于查询的排序学习算法比使用单一模型有明显的改善，“特征排序+RADIUS”算法的改善效果最为明显，提高的幅度在 7.14%（NDCG@1）到 45.00%（NDCG@3）之间。“特征排序+RADIUS”算法也达到比较理想的结果，平均改进在 15%左右。“特征平均+KNN”的方法的实验结果并不十分理想，有些评价指标比使用单一模型要差，这说明使用的启发式方法表示查询，并不是在所有数据集都可行。

（2）在 TD2004 数据集中，两种使用 KNN 来选择相似查询的算法获得比较好的结果。“特征排序+KNN”的所有评价指标都是四种算法中最高的。“特征排序+RADIUS”没有取得预期的效果，可能是由于参数 r 选择的困难性导致。

（3）在 OHSUMED 数据集中，“特征排序+RADIUS”和“特征排序+KNN”的所有评价指标都优于基线，“特征平均+KNN”也仅有 NDCG@3 略低于基线，但是它在 MAP 和 NDCG@1 都取得了最优的结果，而另外三个评价指标的最优结果被“特征排序+KNN”获得。

图 4-1，图 4-2，图 4-3 分别给出了四种算法在 TD2003，TD2004 和 OHSUMED 数据集上的主要评价指标（MAP，NDCG@1，NDCG@3，NDCG@5，NDCG@10）的实验结果比较图。

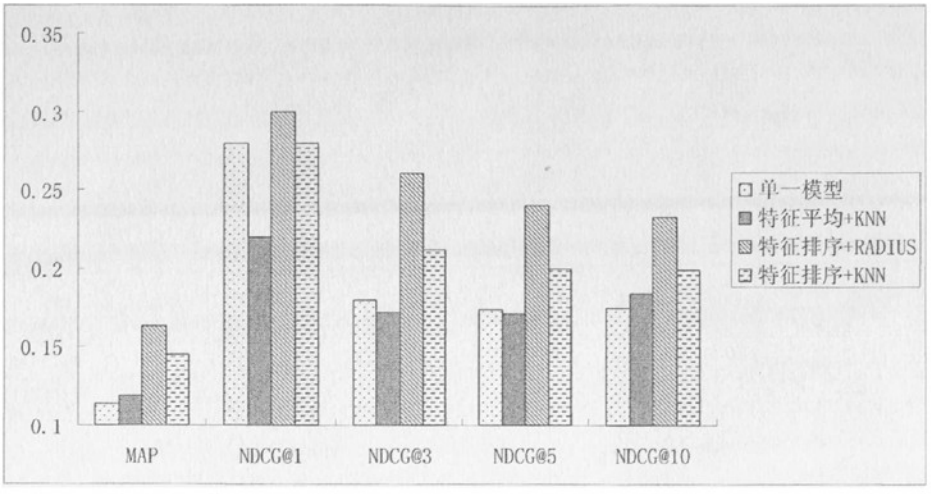


图 4-1 四种算法在 TD2003 数据集上主要评价指标（MAP，NDCG@1，NDCG@3，NDCG@5，NDCG@10）实验结果的比较图

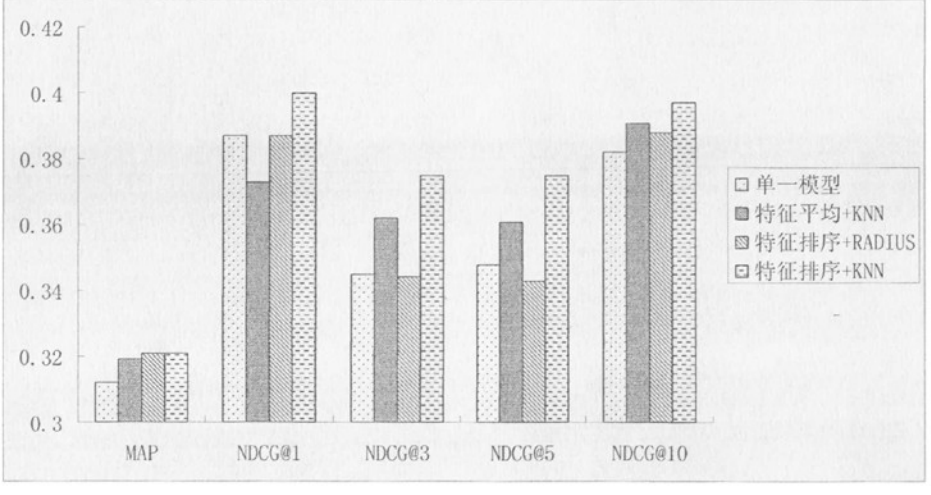


图 4-2 四种算法在 TD2004 数据集上主要评价指标（MAP，NDCG@1，NDCG@3，NDCG@5，NDCG@10）实验结果的比较图

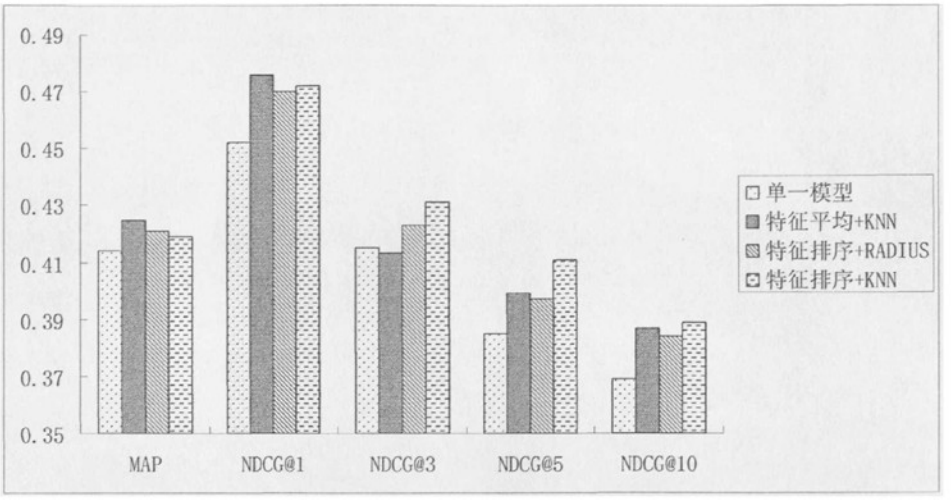


图 4-3 四种算法在 OHSUMED 数据集上主要评价指标（MAP，NDCG@1，NDCG@3，NDCG@5，NDCG@10）实验结果的比较图

从图表中我们可以观察到：

（1）对于大多数评价指标，三种依赖于查询的排序学习算法（“特征排序+RADIUS”，“特征排序+KNN”和“特征平均+KNN”）都比使用单一模型的算法要好。因此，依赖于查询的排序学习对于排序结果的改进确实是有帮助的。

（2）“特征排序+RADIUS”在 TD2003 数据集中获得最好的实验结果，“特征排序+KNN”在 TD2004 和 OHSUMED 数据集中表现最好。因此，本文提出的两种基于特征排序的算法都优于“特征平均+KNN”中用特征平均值表示查询的算法。这说明本文提出的用特征的区分度来表示查询，用特征排序的相似性表示查询的相似性是合理的，优于采用的用特征的平均值来表示查询，用欧氏距离来度量查询的相似性。

（3）“特征排序+RADIUS”和“特征排序+KNN”在各项评价指标中互有胜负，使用固定距离来选择相似查询和选择固定个数的相似性查询可以达到相似的结果。KNN 算法中的参数 k 是一个整数，选择的范围比较小，也比较容易确定，而固定距离的“距离”是比较难把握的，本文在校验集中选择距离的时候，选择的范围是比较粗糙的，作者认为，如果选择距离的时候选择得更加细致，可以达到更好的结果。

4.3.3 不同 k 值和 r 值的实验结果

对于“特征排序+KNN”和“特征排序+RADIUS”算法，本文直接在测试集上选择不同的参数值进行实验，以显示不同参数对实验结果的影响，与前面的实验一样，实验结果也是在5个不同的组（fold）上进行，然后取平均值。

图 4-4 和图 4-5 显示了“特征排序+KNN”算法在 TD2004 数据集中 NDCG@5 和 NDCG@10 随着选择相似查询 k 的个数而变化曲线。

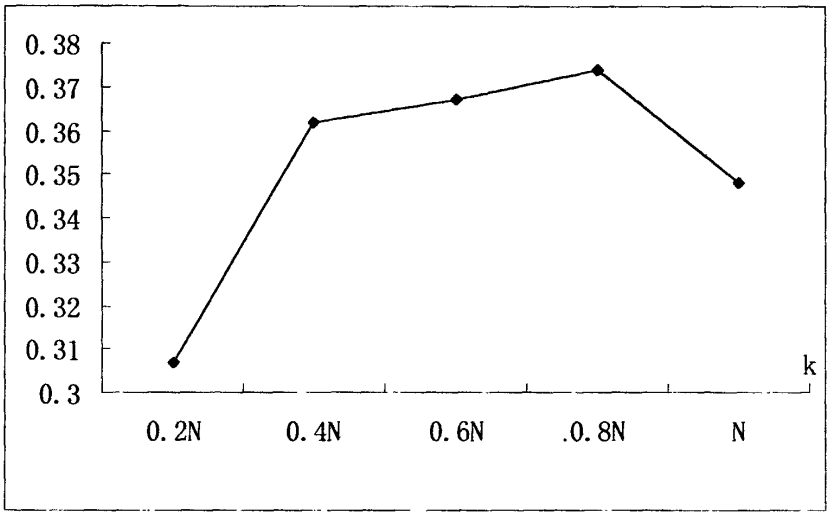


图 4-4 “特征排序+KNN”算法在 TD2004 数据集中 NDCG@5 随 k 的变化曲线

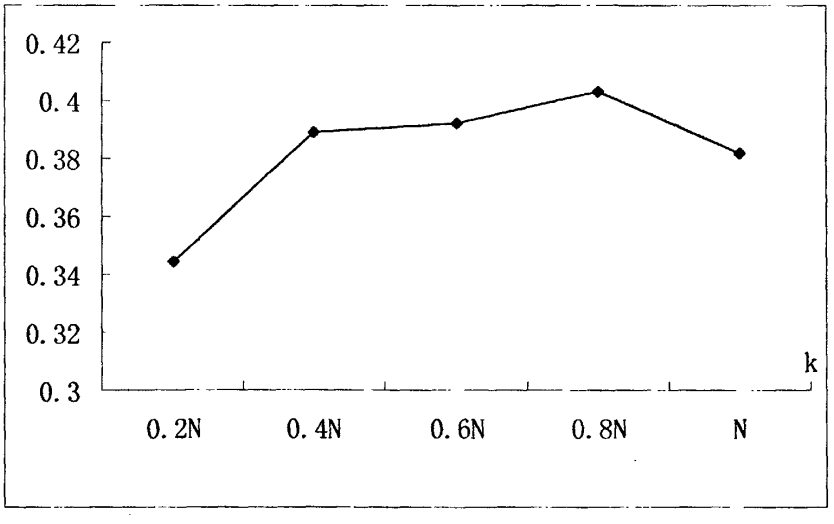


图 4-5 “特征排序+KNN”算法在 TD2004 数据集中 NDCG@10 随 k 的变化曲线

图中的横坐标表示选择的查询个数，当$k=n$时，即为使用单一的模型。从图中我们可以看到，随着k的增加，排序的效果先增加后减小。

图 4-6 和图 4-7 显示了“特征排序+RADIUS”算法在 TD2003 数据集中 NDCG@5 和 NDCG@10 随着距离域值r的变化曲线。

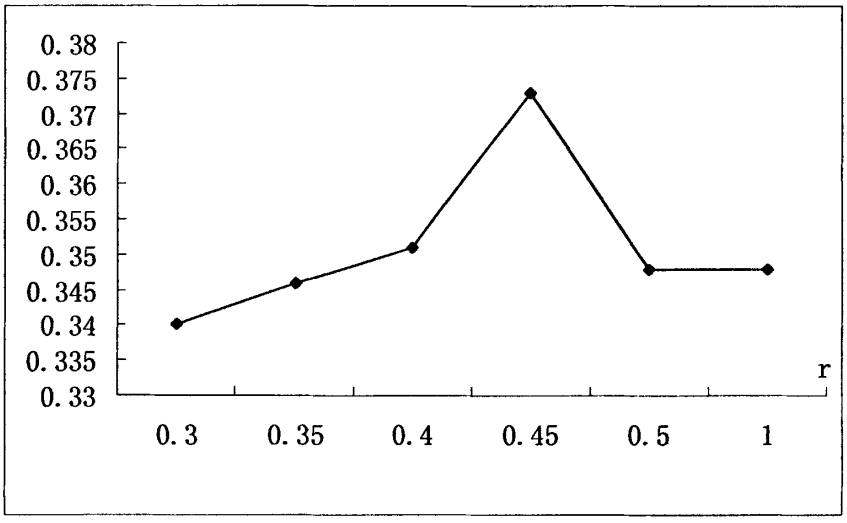


图 4-6 “特征排序+RADIUS”算法在 TD2004 数据集中 NDCG@5 随 r 的变化曲线

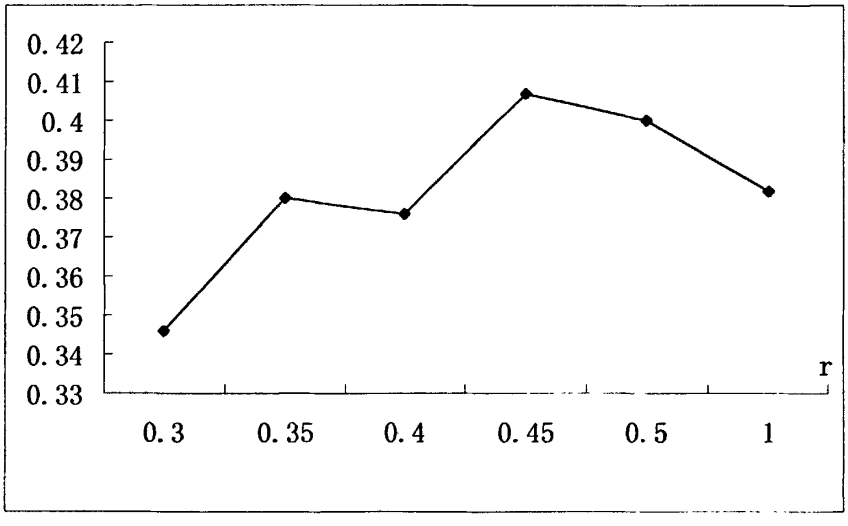


图 4-7 “特征排序+RADIUS”算法在 TD2004 数据集中 NDCG@10 随 r 的变化曲线

图中的横坐标表示选择的查询的距离域值，当$r=1$时，即为使用单一的模

型。从图中我们可以看到，随着 r 的增加，排序的效果先增加后减小。

在前面四幅图里，曲线的变化趋势与本文的预想是一致的，从这些曲线中我们可以观察到：

（1）当 k 值或 r 值比较小的时候，训练集比较小，因为训练样本的不足，排序的结果并不是很好。

（2）随着 k 值或 r 值的增大，更多的相似查询添加到训练集，排序的结果随着可利用的训练信息的增加而不断改进。

（3）当 k 值或 r 值继续增大的时候，与测试查询不相似的训练样本也添加到了训练集，因为有噪声的加入，排序的结果逐渐变差。

这说明依赖于查询的排序模型对于改善排序的结果确实是有帮助的，因为它既利用了与测试查询相似的查询的有用信息，也排除了与测试查询不相似的查询的噪声的干扰。

第5章 总结和展望

5.1 依赖于查询的排序学习算法总结

网页排序是搜索引擎的重要部件，过去主要使用统计学的办法进行网页排序。近年来，有人提出使用机器学习的算法来进行排序学习，并取得了比较好的效果。目前大部分的研究都是从训练集中学习一个模型，然后应用于测试集的所有查询。但是在信息检索中，不同查询之间可能会有比较大的差异。用同一个模型对测试集上的所有查询进行测试，往往不能达到最优的结果。

本文在 Geng 等[3]的工作的基础上，提出了依赖于查询的排序学习框架，该算法以查询聚类为基础。对于测试集中的每一个查询，本文使用查询聚类的算法，从训练集中找出与之相似的查询，组成依赖于查询的训练集，然后使用基本的排序学习算法训练一个依赖于查询的排序模型，最后用于测试。在对查询进行聚类时，如何定义查询的相似性是非常关键的。如果我们可以从训练集中选择出与测试查询相似的查询，构建依赖于测试查询的训练集，那么我们既可以利用相似查询的有用信息，也排除了不相似查询的噪声干扰。

本文提出了一种新颖的，用特征的区分度表示查询的方法，并在此基础上定义查询的相似性。在选择相似查询构建依赖于查询的训练集时，本文使用了两种算法：KNN 算法和固定距离的算法。

实验结果表明，无论使用 KNN 算法还是固定距离的算法选择相似的查询，依赖于查询的排序比使用单一模型的排序结果要好。

5.2 依赖于查询的排序学习算法展望

在算法展望方面，本文认为有以下几点：

（1）对于依赖于查询的排序算法而言，如何比较合理地表示查询以及查询的相似性，是非常关键的。本文从特征区分度的角度来表示查询以及查询的相似性，在进一步的工作中，本文认为可以从其它角度来表示查询和查询的相似性。

（2）本文使用与查询关联文档的特征的标准差来表示特征的区分度，这只

是一种近似的表示方法，它会受到一些噪声点的干扰，在进一步的工作，我们可以考虑如何通过数据挖掘的办法来消除这些噪声点。

（3）本文使用特征的区分度来表示查询。但是，有一些特征是与文档无关，只与查询本身相关的（假如 idf），如果使用本文的表示方法，这些特征的区分度是 0，也就是说，这些特征在特征排序时，并没有起到作用。在下一步工作中，我们可以考虑如何将查询本身的特征结合与特征排序相结合，以更好地表示查询。

（4）在对特征进行排序之前，我们可以考虑采用 Geng 等[40]中的方法，先进行特征选择，只选择一部分特征进行排序。

（5）一个查询可以用一个矩阵表示，比较查询的相似性，其实就是比较两个矩阵的相似性。在比较两个查询的相似性时，我们可以采用矩阵压缩的办法，将两个大矩阵压缩成两个小矩阵，再进行相似性的比较。

（6）本文在构建依赖于查询的训练集时，只选择了与测试查询相似的查询，把不相似的查询从训练集中去掉。我们还可以考虑保留所有查询，只是在训练的时候，降低不相似查询的权重。

（7）在对测试查询的特征区分度进行排序后，我们知道哪些特征对测试查询是比较具有区分度的。因此，在构建依赖于查询的训练集时，我们可以保留原来训练集的所有样本，不过这些样本我们只保留对测试查询比较有区分度的特征，去掉其它特征。

（8）对于训练集和测试集中的查询，我们可以按照特征的排序效果进行聚类。如果两个特征有相似的排序效果，那么它们是同一类的。这样，在比较两个查询的相似性时，我们可以比较两个聚类结果的相似性。

（9）Geng 等[3]提出了两种依赖于查询的排序学习的缓存策略，在进一步的工作中，我们可以研究新的缓存策略。

参考文献

[1].郑实福, 刘挺, 秦兵, 李生. 自动问答综述. 中文信息学报. 2002, 16(6): 46-52.

[2].汤庸, 林鹭贤, 罗烨敏, 潘炎. 基于自动问答系统的信息检索技术研究进展. 计算机应用, 2008, 28(11): 2745-2748.

[3].Xiubo Geng, Tie-Yan Liu, Tao Qin, Andrew Arnold, Hang Li, and Heung-YeungShum. Query Dependent Ranking Using K-Nearest Neighbor. In: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval. Singapore, 2008, 115-122.

[4].Kevin Duh and Katrin Kirchhoff. Learning to Rank with Partially-Labeled Data. In: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval. Singapore, 2008, 251-258.

[5].Daniel E. Rose and Danny Levinson. Understanding user goals in web search. In: Proceedings of the 13th international conference on World Wide Web. New York, NY, USA, 2004, 13–19.

[6].Steven M. Beitzel, Eric C. Jensen, Abdur Chowdhury, and Ophir Frieder. Varying approaches to topical web query classification. In: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. Amsterdam, The Netherland, 2007, 783-784.

[7].Steven M. Beitzel, Eric C. Jensen, Ophir Frieder, David Grossman, David D. Lewis, Abdur Chowdhury, and Aleksandr Kolcz. Automatic web query classification using labeled and unlabeled training data. In: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval. Salvador, Brazi, 2005, 581-582.

[8].In-Ho Kang and GilChang Kim. Query type classification for web document retrieval. In: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval. Toronto, Canada, 2003, 64-71.

[9]. Uichin Lee, zhenyu Liu, and Junghoo Cho. Automatic identification of user goals in web search. In: Proceedings of the 14th international conference on World Wide Web. Chiba, Japan, 2005, 391–400.

[10].Dou Shen, Jian-Tao Sun, Qiang Yang, and Zheng Chen. Building bridges for web query classification. In: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval. Seattle, Washington, USA, 2006, 131–138.

[11].刘铁岩，徐君，李航，马维英. 为搜索引擎学习最优的排序模型. 中国计算机学会通讯, 2007, 3(10): 41-45.

[12].Jay M. Ponte and W. Bruce Croft. A language model approach to information retrieval. In: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval. Melbourne, Australia, 1998, 275-281.

[13].John Lafferty and Chengxiang Zhai. Document Language Models, Query Models, and Risk Minimization for Information Retrieval. In: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval. New Orleans, Louisiana, United States, 2001, 111-119.

[14].Larry Page, Sergey Brin, R Motwani, and T. Winograd. The PageRank citation ranking: bringing order to the Web, Technical report, Stanford University, 1998.

[15].Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. Journal of the ACM, 1999, 46(5): 604-622.

[16].Yunbo Cao, Jun Xu, Tie-Yan Liu, Hang Li, alou Huang and Hsiao-Wuen Hon. Adapting ranking SVM to document retrieval. In: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval. Seattle, Washington, USA, 2006, 186–193.

[17].Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In: Proceedings of the 24th international conference on Machine learning. Corvalis, Oregon, 2007, 129-136.

[18].Ramesh Nallapati. Discriminative models for information retrieval. In: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval. Sheffield, United Kingdom, 2004, 64-71.

[19].Ping Li, Christopher J.C. Burges, and Qiang Wu. McRank: Learning to Rank Using Classification and Gradient Boosting. In: Proceedings 21st Proceedings of

Advances in Neural Information Processing Systems. Vancouver, BC, Canada, 2007.

[20].Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to Rank using Gradient Descent. In: Proceedings of the 22nd international conference on Machine learning. Bonn, Germany, 2005, 89-96.

[21].Ming-Feng Tsai, Tie-Yan Liu, Tao Qin, PHsin-Hsi Chen, and Wei-Ying Ma.. FRank: A Ranking Method with Fidelity Loss. In: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. New York, NY, USA, 2007, 383-390.

[22].Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. The Journal of Machine Learning Research, 2003, 4: 933-969.

[23].Ralf Herbrich, Thore Graepel, and Klaus Obermayer . Support Vector Learning for Ordinal Regression. In: Proceedings of the 9th International Conference on Artificial Neural Networks. Edinburgh, UK, 1999, 97-102.

[24].Yisong Yue, Thomas Finley, Filip Radlinski, and Thorsten Joachims. A Support Vector Method for Optimizing Average Precision. In: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. Amsterdam, The Netherland, 2007, 271-278.

[25].Jun Xu and Hang Li. AdaRank: A Boosting Algorithm for Information Retrieval. In: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. Amsterdam, The Netherland, 2007, 391-398.

[26].Jen-Yuan Yeh, Jung-Yi Lin, Hao-Ren Ke, and Wei-Pang Yang. Learning to Rank for Information Retrieval Using Genetic Programming. In: Proceedings of SIGIR 2007 Workshop on Learning to Rank for Information Retrieval. Amsterdam, The Netherland, 2007, 41-48.

[27].Tao Qin, Xu-Dong Zhang, Ming-Feng Tsai, De-Sheng Wang, Tie-Yan Liu, and Hang Li. Query-level loss functions for information retrieval. Information Processing and Management: an International Journal, 2008, 44(2): 838-855.

[28].Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. ACM Transactions on Information Systems, 2002, 20(4): 422-446.

[29].Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. Listwise Approach to Learning to Rank: Theory and Algorithm. In: Proceedings of the 25th international conference on Machine learning. Helsinki, Finland, 2008, 1192-1199.

[30].Andrei Broder. A taxonomy of web search. ACM SIGIR Forum, 2002, 36(2): 3-10.

[31].Thorsten Joachims. A support vector method for multivariate performance measures. In: Proceedings of the 22nd international conference on Machine learning. Bonn, Germany, 2005, 377-384.

[32].Soumen Chakrabarti, Rajiv Khanna, and Chiru Bhattacharyya. Structured learning for non-smooth ranking losses. In: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. Las Vegas, Nevada, USA, 2008, 88-96.

[33].Ying Li, Zijian Zheng, and Honghua(Kathy) Dai. KDD CUP-2005 Report: Facing a Great Challenge. ACM SIGKDD Explorations Newsletter, 2005, 7(2): 91-99.

[34].Paul Ogilvie and Jamie Callan. Experiments Using the Lemur Toolkit. In: Proceedings of the Tenth Text Retrieval Conference (TREC-10). Gaithersburg, MD, 2001, 103–108.

[35].David Hawking and Nick Craswell. Overview of the trec-2001 web track. In: Proceedings of the Tenth Text Retrieval Conference (TREC-10). Gaithersburg, MD, 2001, 61–67.

[36].Google, Description of "I'm Feeling Lucky" feature, http://www.google.com/help/features.html#lucky.

[37].W. Bruce Croft. Combining approaches to information retrieval. In Advances in Information Retrieval: Recent Research from the Center for Intelligent Information Retrieval, 2000, 7(1): 1–36.

[38].Kiduk Yang. Combining text- and link-based retrieval methods for WEB IR. In: Proceedings of the Tenth Text Retrieval Conference (TREC-10). Gaithersburg, MD,

2001, 609–618.

[39].Ruihua Song, Ji-Rong Wen, Shuming Shi, Guomao Xin, Tie-Yan Liu, Tao Qin, Xin Zheng, Jiyu Zhang, Guirong Xue, and Wei-Ying Ma. Microsoft research asia at web track and terabyte track of trec 2004. In: Proceedings of the Thirteenth Text REtrieval Conference Proceedings (TREC-2004). Gaithersburg, MD, 2004.

[40].Xiubo Geng, Tie-Yan Liu, Tao Qin, and Hang Li. Feature Selection for Ranking. In: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. Amsterdam, The Netherland, 2007, 407-414.

[41].Maurice G. Kendall. Rank correlation methods. Oxford University Press, 1990.

[42].Albert M. Liebetrau. Measures of association, volume 32 of Quantitative Applications in the Social Sciences. Sage Publications, Inc., 1983.

[43].Moses S. Charikar. In Proceedings of the 34th ACM STOC Symposium on Theory Of Computing. Montreal, Quebec, Canada, 2002, 380–388.

[44].Tie-Yan Liu, Jun Xu, Tao Qin, Wenying Xiong, and Hang Li. LETOR: Benchmark dataset for research on learning to rank for information retrieval. In: Proceedings of SIGIR 2007 Workshop on Learning to Rank for Information Retrieval. Amsterdam, The Netherland, 2007, 3-10.

[45].Tie-Yan Liu. Learning to Rank for Information Retrieval. In WWW 2008 Tutorial:http://research.microsoft.com/en-us/people/tyliu/learning_to_rank_tutorial_-_www_-_2008.pdf

[46].Tie-Yan Liu. Learning to Rank for Information Retrieval. In SIGIR 2008 Tutorial: http://research.microsoft.com/en-us/people/tyliu/letor-tutorial-sigir08.pdf

[47].Kalervo Järvelin and Jaana Kekäläinen. IR evaluation methods for retrieving highly relevant documents. In: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval. Athens, Greece, 2000, 41-48.

[48].Stephen Robertson. Overview of the okapi projects. Journal of Documentation, 1997, 53(1): 3-7.

[49].Yuting Liu, Bin Gao, Tie-Yan Liu, Zhiming Ma, Shuyuan He, and Hang Li.

BrowseRank: Letting Web Users Vote for Page Importance. In: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval. Singapore, 2008, 451-458.

附录一 研究生阶段发表论文列表

以下为本人研究生阶段发表的论文列表：

1. Yan Pan, Yong Tang, Luxian Lin, and **Yemin Luo**. Question classification with semantic tree kernel. Proceedings in 31st ACM SIGIR.

2. 汤庸，林鹭贤，罗烨敏，潘炎. 基于自动问答系统的信息检索技术研究进展. 《计算机应用》, 2008 年第 28 卷第 11 期.

3. CHEN Guohua, TANG Yong, **LUO Yemin**, and ZHAO Chunfang. The Design and Implementation of the OPC based Multi-campus Oriented Solar Energy Monitoring System. 第三届国际计算机新科技与教育(ICCSE2008)学术会议.

致谢

首先衷心地感谢我的导师汤庸教授。汤老师渊博的学识、深邃的洞察力、出色的教学科研能力、不畏辛劳的工作作风、平易近人的性格、对学生认真负责的态度使我受益非浅。所有这些无疑是我二年研究生生活中最为宝贵的收获，这种收获对于我今后的工作、学习和生活必将产生深远的影响。

我要向中山大学潘炎老师致以最真挚的谢意！跟潘老师探讨问题是一件非常愉快的事情。他是我学习、工作和生活的良师益友。从潘老师的身上，我可以看到中国的青年科学工作者对学术孜孜不倦的追求。他严肃的科学态度，严谨的治学精神，精益求精的工作作风，深深地感染和激励着我。

实验室是一个温暖的大家庭，这里既有浓厚的学术交流，又充满了同学朋友之间的关心。感谢信息检索研究小组的成员们——余峰师兄、陈国华师兄、戚洪睿师弟、蔡奕伦师弟、罗海霞师妹，还有同年级的林鹭贤、吴桂宾、彭泽武等。因为有你们，使我二年的研究生生活变得丰富多彩。

感谢我的父母，你们是我身后最坚实的支撑力量，你们无私地给予我的一切，我终生都难以偿还。感谢我的妹妹和所有家人，没有你们的关爱和支持，就不会有今天的我。这篇论文是我硕士学业的总结，更是我十八年求学生涯的总结，把它献给你们。

最后，衷心感谢所有关心、帮助过我的人。

依赖于查询的排序学习算法研究

作者：罗烨敏
学位授予单位：中山大学