文章编号：1003-0077(2011)02-0038-06

基于流形排序的查询推荐方法

朱小飞[1,2,3]，郭嘉丰[1]，程学旗[1]，杜 攀[1]

(1. 中国科学院 计算技术研究所，北京 100190；
2. 中国科学院 研究生院，北京 100049；3. 重庆理工大学 数理学院，重庆 400054)

摘 要：针对传统查询推荐方法中存在的相关性度量问题和冗余性问题，该文中提出了一种新的基于流形排序的查询推荐方法。该方法利用查询数据内在的全局流形结构来获得查询之间的相关性，可以有效避免传统方法中相关性度量对高维稀疏查询数据处理的不足；同时，该方法通过提升结构上具有代表性的查询来达到减小查询推荐的冗余性。在一个大规模商业搜索引擎查询日志上的实验结果表明：使用流形排序的查询推荐方法要优于传统查询推荐方法和现有的 Hitting-time Ranking 方法。

关键词：查询推荐；流形排序；click-through data
中图分类号：TP391　　**文献标识码**：A

Query Recommendation Based on Manifold Ranking

ZHU Xiaofei[1,2,3], GUO Jiafeng[1], CHENG Xueqi[1], DU Pan[1]

(1. Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China;
2. Graduate University of the Chinese Academy of Sciences, Beijing 100049, China;
3. School of Mathematics and Statistics, Chongqing University of Technology, Chongqing 400054, China)

Abstract: To address problems of both relevance measurement and redundance in traditional query recommendation approaches, in this paper, we propose a novel query recommendation approach based on Manifold Ranking. This approach exploites the intrinsic global manifold structure to capture the relevance among queries, and effectively avoids the deficiency of the relevance measurement in traditional approaches when dealing with high-dimensional query data. Meanwhile, it also reduces the redundance by boosting representative queries in the structure. Empirical experiments on a large scale query log of a commercial search engine show that query recommendation using Manifold Ranking is superior to both the traditional approach and the existing Hitting-time Ranking approach.

Key words: query recommendation; manifold ranking; click-through data

1 引文

目前，互联网上存储了大量的信息，搜索引擎已经成为用户访问这些信息最有效的途径。用户通过输入想要查询的关键字到搜索引擎，然后搜索引擎返回搜索结果供用户使用。然而，构造一个合适的查询并不是一件轻松的事情，一方面因为查询通常比较短[1]（1～2 个词）并且存在歧义性[2]（例如 java，apple 等），另一方面，由于用户对所查询的内容不熟悉，使得自身对要查询什么内容不清楚，例如用户输入 harry potter，他可能是对名字叫 harry potter 的一款游戏感兴趣，或者对同名的电影感兴趣，也可能对同名的小说感兴趣。如何帮助用户更好的找到需要的信息已经成为一个至关重要的问题。

查询推荐通过向用户推荐相关的查询，以帮助用户构造更有效的查询，目前已经成为一个十分重要的研究方向。一些著名的搜索引擎公司（如

收稿日期：2010-08-19　**定稿日期**：2010-11-09
基金项目：国家青年自然科学基金资助项目(61003166)

作者简介：朱小飞(1979—)，男，博士生，主要研究方向为信息检索；郭嘉丰(1980—)，男，博士，助研，主要研究方向为信息检索；程学旗(1971—)，男，博士，研究员，主要研究方向为信息检索、复杂网络分析与社会计算、网络安全。

Google 和 Yahoo)开始为用户提供查询推荐系统，以帮助用户更方便地找到所需要的信息。

传统的查询推荐方法主要关注于向用户推荐相关的查询，通过度量候选查询与用户所提交查询的相关程度，对候选查询进行排序，并推荐最相关的前 k 个查询给用户。这种方法可以有效地帮助用户构造更恰当的查询，但同时也存在一些问题：(1) 相关性度量问题。传统查询推荐方法通常在欧式空间上计算查询之间的相似性，但由于查询日志中的查询具有高维稀疏性，传统的相似度度量方法往往不能很好的反应用户查询之间的相关性。例如，在基于关键词来度量查询之间的相似性时，由于两个相关的查询(如“national basketball association”和“nba”)之间没有相同的查询关键词，从而被认为是不相关；或者在基于用户点击的 URL 来计算查询之间的相似性时，两个相关的查询因为没有点击相同的 URL(因用户结果点击的稀疏性)，会导致其被认为不相关。(2)冗余性问题。由于仅关注候选查询与源查询的相关性，使得传统的推荐方法所推荐的查询存在较大的冗余性(推荐的查询之间的差异性小)，例如当用户输入查询“abc”时，系统可能会同时向用户推荐“abc television”和“abc tv”，而这两个查询其实是表达了相同的意思。

为了避免传统查询推荐方法中存在的问题，已有研究人员提出一些解决方法，例如 Mei 等人[3]基于随机游走(random walk)提出了通过计算各候选查询到源查询的游走时间(hitting time)来进行查询推荐(文中使用 Hitting-time Ranking 来表示)，该方法可以有效地推荐长尾(long tail)的查询给用户，以降低查询推荐的冗余性。然而通过这种方法获得的长尾查询往往不常见，相关性也无法得到保证，因此会影响到推荐的有效性和实用性。

针对已有查询推荐方法的不足，我们提出了一种基于流形排序的查询推荐方法(文中使用 Manifold Ranking 来表示)。通过把用户查询建模在流形结构上，我们可以利用查询数据内在流形结构来获得查询之间的相关性，从而有效避免传统方法的相关性度量问题。此外，基于查询的流形结构，我们使用流形排序算法来对候选查询进行排序和推荐，由于各查询的排序得分不再仅来自于源查询，同时也来自于其邻居查询，从而使得那些结构上与源查询相近并且具有代表性的查询(那些具有较多邻居结点的查询)获得更大的排序得分。由此可见，和 Hitting-time Ranking 不同，Manifold Ranking 是通过提升那些结构上具有代表性的查询来减小查询推荐的冗余性。

我们使用了一个大规模商业搜索引擎用户查询日志来进行实验，该查询日志包括约 1 500 万条用户查询记录。我们将 Manifold Ranking 与两种基准方法：基于传统的相似性排序算法(Naïve Ranking)和 Hitting-time Ranking 进行比较，并通过一个自动评价机制来对各方法所推荐查询的相关性和差异性进行评价。实验结果表明，Manifold Ranking 方法在保持较高相关性的条件下，明显地提高了查询推荐的差异性，在整体上要优于 Naïve Ranking 和 Hitting-time Ranking 方法。

本文的组织结构如下：第 2 节介绍相关的研究工作，基于 Manifold Ranking 的查询推荐方法将在第 3 节进行讨论，第 4 节给出了实验结果及其评价，结论将在最后一节给出。

2　相关工作

查询日志作为一种用户产生的数据，是众多用户在使用搜索引擎进行查询操作时的日志记录，其包含大量有价值的信息，被称为“大众智慧”(wisdom of crowds)[4]。近年来，许多研究工作开始使用查询日志中的 click-through data 来挖掘查询之间的语义相关关系，这种相关关系可以被用来进行查询推荐。例如，Beeferman 等人[5]通过对 query-URL 二部图上使用凝聚聚类算法来发现相关查询；Wen 等人[6]同时考虑使用 click-through data 和查询及文档的内容信息来确定相似查询。Li 等人[7]提出了一个两阶段的查询推荐方法：发现阶段和排序阶段。在发现阶段，使用基于 query-URL 向量来计算查询之间的相似度；在排序阶段，使用层次凝聚聚类算法来对相似查询进行排序。Baeza-Yates 等人[8]基于 query-URL 二部图定义了 3 种连边类型：identical cover，strict complete cover 和 partial cover，并以此计算查询间的语义关系。以上研究工作共同的特点是仅关注于推荐相关的查询。Mei 等人[3]在 query-URL 二部图上使用随机游走方法，依据各候选查询到源查询的游走时间来对查询进行排序，可以有效的推荐长尾的查询，但是其缺点在于：由于推荐长尾的查询可能与源查询相关程度不高，降低了查询推荐的相关性。

Manifold Ranking 是由 Zhou 等人[9-10]首先提出，与传统的欧式空间上直接计算查询之间的相似

性不同，该方法通过利用大规模数据内在的全局流形结构来计算排序得分。直观的解释为：首先构造一个带权网络，并且分配给源节点一个正的得分，其他待排序节点的得分设为0，然后每一个节点将其自身得分传播到邻居节点直到整个网络达到平衡状态。除源节点外的所有节点按照最终得分大小进行排序（得分越大，排序越靠前）。目前 Manifold Ranking 已经被用来对图像或文档摘要进行排序，并取得很好的效果。例如，He 等人[11]基于 Manifold Ranking 进行图像排序，其利用所有数据之间的关系，来测量输入查询与所有图片的相关性。Wan[12]等人使用 Manifold Ranking 来进行主题相关的多文档摘要，通过利用文档中所有句子相关性以及句子与主题的相关性，并基于贪心算法来实现有差异的文档摘要。

3 我们的工作

在这一部分的内容中，我们详细描述了如何使用 click-through data 来构造查询的流形结构，然后在此基础上使用 Manifold Ranking 方法来进行查询推荐的过程。

3.1 基本概念

给定一组查询 $\chi=\{x_0,x_1,\cdots,x_n\}\subset\mathbb{R}^m$，第一个查询 x_0 表示源查询，其他查询 $x_i(1\leqslant i\leqslant n)$ 为候选查询。令 $d:\chi\times\chi\to\mathbb{R}$ 表示 χ 上的一个度量，其中 $d(x_i,x_j)$ 表示查询 x_i 和 x_j 的距离（如欧式距离）。$f:\chi\to\mathbb{R}$ 为排序函数，其为每一个查询 $x_i(0\leqslant i\leqslant n)$ 计算一个排序得分 f_i，这里我们可以将 f 看成是一个向量 $f=[f_0,f_1,\cdots,f_n]^T$。同时，定义向量 $y=[y_0,y_1,\cdots,y_n]^T$，其中 $y_0=1$（x_0 是源查询），$y_i=0$ $(1\leqslant i\leqslant n)$。

3.2 构造查询的流形结构

我们首先使用 click-through data 构建一个 query-URL 二部图（如图1所示），$G=\langle V,E\rangle$，其中 $V=V_1\times V_2$，V_1 表示所有的查询节点，V_2 表示所有 URL 节点，若用户使用查询 x_i 进行检索，并且点击了检索结果中 URL j，则存在一条边 $e_{ij}\in E$。对于每一个查询，我们可以用一个 query-URL 向量来表示。与传统的 TF-IDF 方法类似，我们定义 $e_{ij}=tf_{ij}\times\log(N/df_j)$，其中，$tf_{ij}$ 表示用户使用查询 x_i 并点击 URL j 的次数，df_j 表示用户点击 URL j 所使用

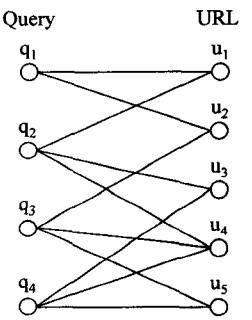

图1 query-URL 二部图

的不同查询的数目，N 表示所有查询的数目。我们使用带权网络表示流形结构，网络中每一个节点对应于一个查询，相似查询之间有一条边相连，边的权重表示两个查询的相似程度。对于查询 $x_i(0\leqslant i\leqslant n)$，我们先对其对应的 query-URL 向量进行规范化，则其第 k 元素 x_i^k 为：

$$x_i^k=\begin{cases}norm(e_{ik}) & if\ \text{edge}\ e_{ik}\ \text{exist}\\ 0 & otherwise\end{cases}\tag{1}$$

其中 $norm(e_{ik})=e_{ik}/\sqrt{\sum_{u=1}^{|U|}e_{iu}^2}$，$|U|$ 表示 URL 集合的大小。我们使用欧式距离来计算查询 x_i 和 x_j 之间的距离 $d(x_i,x_j)=\sqrt{\sum_{u=1}^{|U|}(x_i^u-x_j^u)^2}$。当带权网络中 x_i 与 x_j 存在连边时，边的权重定义为：

$$w_{ij}=e^{-d(x_i,x_j)^2/2\sigma^2}\tag{2}$$

令 $w_{ii}=0$。此外，若带权网络中边所对应的两个查询不是互为 k 近邻（$k=50$），则删除此边，以保持带权网络的稀疏性。至此，我们得到了一个查询带权网络（即查询流形结构），其可以用一个仿射矩阵 $W=[w_{ij}]$ 来描述。Manifold Ranking 算法如下：

1. Compute the pair-wise similarity values w_{ij} between queries i and j, and then construct a weighted network. The affinity matrix W is defined by $W=[w_{ij}]$.
2. Symmetrically normalize W by $S=D^{-1/2}WD^{-1/2}$, in which D is the diagonal matrix with (i,i)-element equal to the sum of the i-th row of W.
3. Iterate $f^{(t+1)}=\alpha Sf^{(t)}+(1-\alpha)y$ until convergence, where α is a parameter in $[0,1)$, started with $f^{(0)}=0$.
4. Let f_i^* denote the limit of the sequence $\{f_i(t)\}$. Rank each query x_i according to its ranking scores f_i^* (largest ranked first).

3.3 Manifold Ranking 方法

基于以上定义的查询流形结构，我们使用流形

排序算法来进行查询排序和推荐。首先，我们对 W 进行对称规范化：$S=D^{-1/2}WD^{-1/2}$，其中 D 为对角矩阵，其对角元素分别为 W 各行元素之和。然后，按照下面的迭代公式：

$$f^{(t+1)}=\alpha Sf^{(t)}+(1-\alpha)y \tag{3}$$

迭代至 f 收敛。其中，α 用来控制来自于先验的得分和来自于结构上相邻的结点得分对最终排序得分的贡献，α 值越大表示来自于相邻的节点得分贡献所占比例越大($\alpha\in[0,1)$)。

Manifold Ranking 的过程可以形象地描述为：首先构造一个带权网络，网络中每一个节点对应于一个查询；初始化时，给源查询赋以一个正的排序得分，其余候选查询的排序得分为 0；网络中所有的节点将其得分传播给与其相邻的节点，当网络达到全局平衡状态时，传播过程停止；所有候选查询的得分将被用来对候选查询进行排序，并将排序最前的 k 个候选查询推荐给用户。Manifold Ranking 算法的详细过程，文献[9]中证明了 f 最终会收敛到：

$$f^*=\beta(I-\alpha S)^{-1}y \tag{4}$$

其中 $\beta=1-\alpha$，虽然可以直接使用(4)式来计算 f^*，但是由于涉及到计算逆矩阵 $(I-\alpha S)^{-1}$，当数据规模较大时，需要的计算开销很大，因此与大多数研究工作[10-11]一样，我们这里选择使用迭代方式来计算 f^*。

此外，为了减小计算开销同时又不影响算法的有效性，我们从源查询出发使用广度优先搜索构造一个子图，直到子图中的查询节点达到预先指定的数目。子图的每一个节点(除源查询节点外)可以看成是源查询的一个候选查询，那些不在子图中的节点，由于离源查询节点足够远，以至于不可能被推荐给用户，将这部分节点去除，不会影响实际查询推荐的效果，但可以有效的减小算法计算开销。

4　实验结果及评价

4.1　数据集

我们使用一个商业搜索引擎的查询日志作为数据集，该查询日志记录了 2006 年 5 月 1 日至 2006 年 5 月 31 日期间的 1 500 万条查询。对于每一个查询，日志中记录其相应的信息，包括查询 ID，查询词本身，所点击的 URL，URL 的位置信息，时间戳，用户 Session ID，URL 在结果页面中的排序，以及返回的结果数目。我们对查询日志进行预处理：只保留英文查询，并用空格替代查询中的标点符号，不执行词干还原和去除停用词的操作。为了减小噪音数据带来的影响，对于每个查询要求用户的点击次数不能小于 3 次。表 1 是所获得的 query-URL 二部图的统计信息。

表 1　query-URL 二部图的统计信息

	Size
# Query Nodes	191 585
# URL Nodes	251 427
# Edges	318 947

4.2　评价指标

评价查询推荐是一件十分困难的事情，因为对于一个查询来说并不存在某种标准的推荐结果。为了使评价结果更为客观合理，这里我们采用了一种自动评价的策略。该策略使用开放式分类目录搜索系统 ODP (the Open Directory Project)和 Google 的检索结果作为评价数据资源，对查询推荐结果的相关性和差异性进行评价。ODP 是目前最大的人工编制的分类检索系统，已经被一些研究人员用来作为评价数据，例如 Baykan 等人[13]使用 ODP 作为 URL 主题类别信息来评价分类结果，Baeza-Yates 等人[8]使用其评价查询的语义相关性。

4.2.1　相关性指标

对于 ODP 中的两个类别 c_i 和 c_j，我们定义其相关度为类别的相同前缀长度与类别最大长度的商，形式化的表示为 $s(c_i,c_j)=|l(c_i,c_j)|/\max\{|c_i|,|c_j|\}$，其中 $|l(c_i,c_j)|$ 为类别 c_i 和 c_j 相同前缀长度，$|c|$ 为类别 c 的长度。举例来讲，两个目录“Arts/Television/News”和“Arts/Television/Stations/North_America/United_States”的相关度为 2/5。因为其相同前缀为“Arts/Television/”，长度为 2，而最大类别为“Arts/Television/Stations/North_America/United_States”，长度为 5。

对于每个查询，在抓取 Google Directory 中前 k 个 ODP 目录作为该查询对应的类别集合，设 C 和 C' 分别为查询 q 和 q' 所对应的 ODP 类别集合，类似于文献[8]，我们用类别集合 C 与 C' 之间最相似的两个类别的相关度来衡量查询 q 和 q' 相关性。形式化表示为：

$$r(q,q')=\max_{c_i\in C,c_j\in C'}s(c_i,c_j) \tag{5}$$

因此，对于给定的源查询 q，其推荐查询结果的

相关性指标定义为：所有推荐查询与 q 的平均相关性，形式化表示为：

$$rel(q)=\frac{1}{|S|}\sum_{q'\in S} r(q,q') \tag{6}$$

其中 S 为源查询 q 的推荐查询集合，$|S|$ 为推荐查询集合中查询数目。

4.2.2 差异性指标

对于每个查询 q，我们抓取 Google 前 k 个搜索结果的 URL，作为查询 q 对应的 URL 集合。查询 q 和 q' 的差异度可以定义为：查询 q 和 q' 的 URL 集合中相异的 URL 的数目所占的比例，形式化表示为：$d(q,q')=1-|o(q,q')|/k$，其中 $|o(q,q')|$ 是查询 q 和 q' 的 URL 集合中相同 URL 的数目。因此，对于给定的源查询 q，其推荐查询结果的差异性指标定义为：所有推荐查询两两之间的平均差异度，形式化表示为：

$$div(q)=\sqrt{\frac{\sum_{q\in S}\sum_{q'\in S} d(q,q')}{|S|(|S|-1)}} \tag{7}$$

其中 S 为源查询 q 的推荐查询集合，$|S|$ 为推荐查询集合中查询数目。

4.3 实验结果及分析

我们随机抽取了 150 个查询作为测试样本，类似于文献[14]，我们要求这些查询的各自总的 URL 点击次数介于 700 至 15 000 次之间，这样做的目的是为了避免选择那些过于频繁出现的查询（例如 www，car，book 等查询，因为对其进行推荐没有实质意义）和过于不频繁出现的查询（查询日志中不存在可以推荐的查询）。其他参数设置为：算法迭代的次数设置为 30，$\alpha=0.99$，$\sigma=1.25$。

我们将基于 Manifold Ranking 的查询推荐方法，与另外两种查询推荐方法（Naïve Ranking，Hitting-time Ranking）进行了实验比较，实验结果分别显示在表 2 和表 3[①] 中。表 2 是对三种不同方法的相关性指标进行比较，从表 2 中我们可以看到 Manifold Ranking 与 Naïve Ranking 的查询推荐结果在相关性指标方面接近相同（Naïve Ranking 的平均相关性指标为 0.891，Manifold Ranking 平均相关性指标为 0.898）；而 Hitting-time Ranking 的查询推荐结果在相关性指标方面要明显低于前两种方法（Hitting-time 的平均相关性指标只有 0.859），这主要是因为 Hitting-time Ranking 方法过于强调推荐长尾的查询，其计算候选查询到源查询的游走时间来作为推荐依据，而忽略了这样一个事实：候选查询到源查询的游走时间的值小，并不意味着源查询到候选查询的游走时间的值也小，这就造成推荐的查询与源查询相关性不高。表 3 是对三种不同方法的差异性指标（所推荐查询的冗余性越大，其差异性指标越低）进行比较，从表 3 中我们可以看到 Manifold Ranking 与 Hitting-time Ranking 的查询推荐结果在差异性指标方面要显著高于 Naïve Ranking 方法

表 2 三种查询推荐方法（Naïve Ranking，Hitting-time Ranking，Manifold Ranking）在不同推荐数目下相关性指标的比较

Num. of Rec.	Naïve	Hitting-time	Manifold
1	0.895 357	0.865 029	0.929 037
2	0.889 213	0.867 908	0.921 127
3	0.887 114	0.855 341	0.914 037
4	0.889 551	0.858 329	0.907 529
5	0.894 822	0.859 017	0.897 493
6	0.889 221	0.861 678	0.892 139
7	0.890 248	0.856 508	0.888 523
8	0.890 754	0.858 719	0.881 784
9	0.889 303	0.854 848	0.875 094
10	0.890 152	0.851 882	0.872 619
Average	0.890 574	0.858 926	0.897 938

表 3 三种查询推荐方法（Naïve Ranking，Hitting-time Ranking，Manifold Ranking）在不同推荐数目下差异性指标的比较

Num. of Rec.	Naïve	Hitting-time	Manifold
2	0.720 000	0.794 000	0.740 000
3	0.723 778	0.787 556	0.756 667
4	0.735 222	0.793 000	0.762 000
5	0.744 622	0.797 600	0.777 822
6	0.759 644	0.805 956	0.784 978
7	0.772 462	0.812 794	0.791 326
8	0.782 974	0.820 810	0.795 979
9	0.792 264	0.829 370	0.801 718
10	0.801 247	0.835 559	0.809 108
Average	0.759 135	0.808 516	0.779 955

① 注：我们没有列出推荐数目为 1 的差异性指标，因为差异性指标是计算推荐查询两两之间的平均差异度，当推荐数目为 1 时，计算出来的差异性指标没有评价意义。

(Hitting-time Ranking 与 Manifold Ranking 的平均差异性指标比 Naïve Ranking 方法分别提高了 4.9 和 2.1 个百分点)。从差异性指标比较的结果来讲，Hitting-time Ranking 要优于 Manifold Ranking 的推荐结果。但是如前面分析的结果所示，Hitting-time Ranking 的平均相关性指标比 Naïve Ranking 低了 3.9 个百分点，而 Manifold Ranking 平均相关性指标和 Naïve Ranking 基本相同。从查询推荐的角度来讲，我们希望所推荐的查询与源查询尽可能相关，同时所推荐的查询之间又尽可能保持差异性。考虑查询推荐的相关性与差异性，我们可以得到以下结论：Manifold Ranking 其在不损失相关性的条件下，明显的提高了查询推荐的差异性，整体上要优于 Naïve Ranking 和 Hitting-time Ranking。

5　结论

本文中，我们提出了一种基于流形排序的查询推荐方法，该方法通过利用查询数据内在全局的流形结构来进行查询推荐，避免了传统查询推荐方法在相关性度量方面和冗余性方面的不足。与现有 Hitting-time Ranking 相比，Manifold Ranking 通过提升结构上具有代表性的查询，从而使得所推荐的查询与源查询差异性得到的提高，同时保持较高的相关性。在一个大规模商业搜索引擎查询日志上的实验结果表明，Manifold Ranking 要明显优于两种基准方法(Naïve Ranking 和 Hitting-time Ranking)。

参考文献

[1] H. Cui, J.-R. Wen, J.-Y. Nie, and et al. Probabilistic query expansion using query logs[C]//WWW '02, 2002: 325-332.

[2] J.-R. Wen, J.-Y. Nie, and H.-J. Zhang. Clustering user queries of a search engine[C]//WWW '01, 2001: 162-168.

[3] Q. Mei, D. Zhou, and K. Church. Query suggestion using hitting time[C]//CIKM '08, 2008: 469-478.

[4] J. Surowiecki. The wisdom of crowds: why the many are smarter than the few and how collective wisdom shapes business [C]//Doubleday, Reading, MA, 2004.

[5] D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log [C]//KDD '00, 2000: 407-416.

[6] J.-R. Wen, J.-Y. Nie, and H.-J. Zhang. Clustering user queries of a search engine[C]//WWW '01, 2001: 162-168.

[7] L. Li, Z. Yang, and et al. Query-url bipartite based approach to personalized query recommendation[C]//AAAI'08, 2008: 1189-1194.

[8] R. Baeza-Yates and A. Tiberi. Extracting semantic relations from query logs[C]//KDD '07, 2007: 76-85.

[9] D. Zhou, O. Bousquet, T. N. Lal, and et al. Learning with local and global consistency [C]//NIPS '03, 2003.

[10] D. Zhou, J. Weston, A. Gretton, and et al. Ranking on data manifolds[C]//NIPS'03, 2003.

[11] J. He, M. Li, H.-J. Zhang, and et al. Manifold-ranking based image retrieval[C]//MULTIMEDIA'04, 2004: 9-16.

[12] X. Wan, J. Yang, and J. Xiao. Manifold-ranking based topic-focused multi-document summarization [C]//IJCAI'07, 2007: 2903-2908.

[13] E. Baykan, M. Henzinger, L. Marian, and I. Weber. Purely url-based topic classification[C]//WWW '09, 2009: 1109-1110.

[14] P. Boldi, F. Bonchi, C. Castillo, and et al. Query suggestions using query-flow graphs [C]//WSCD '09, 2009: 56-63.