情 报 学 报 ISSN 1000 - 0135
第 31 卷 第 1 期 47 - 59, 2012 年 1 月
JOURNAL OF THE CHINA SOCIETY FOR SCIENTIFIC AND TECHNICAL INFORMATION ISSN 1000 - 0135
Vol. 31 No. 1, 47 - 59 January 2012

doi:10.3772/j.issn.1000-0135.2012.01.007

基于神经网络的 Listwise 排序学习方法的研究[1)]

林 原 林鸿飞
(大连理工大学计算机科学与技术学院, 大连 116024)

摘要 近年来排序学习方法以其优异的性能成为信息检索领域研究的一个热点。排序学习方法应用机器学习方法训练排序模型用于文档相关性排序,取得了良好的实验结果。在多种排序学习模型中又以 Listwise 方法的效果最为显著,特别是基于神经网络的排序学习算法以其良好的理论基础,灵活的损失函数构造形式,成为排序学习研究的重要手段。本文对基于神经网络的 Listwise 排序学习方法及其改进方法进行综述,并介绍该方面研究的最新进展。

关键词 排序学习 神经网络 Listwise

Research on Listwise Approaches to Learning to Rank Based on Neural Network

Lin Yuan and Lin Hongfei
(*School of Computer Science and Engineering*, *Dalian University of Technology*, *Dalian* 116024)

Abstract Learning to rank approach, with outstanding performance, has become a hot research issue of information retrieval. Its task is to learn a ranking model for sorting documents by relevance according to a query based on machine learning model. The performance of listwise approach is better than other learning to rank approaches, especially, the approach based on neural network is one of most important means to research the learning to rank approaches for its favorable foundations in theory and flexible construction forms of loss function. This paper summarizes the ranking neural network based approaches to introduce the latest research on this issue.

Keywords learning to rank, neural network, Listwise

1 引 言

排序学习[1](learning to rank):是一个信息检索与机器学习相结合的研究领域。它的目标是应用机器学习算法学习排序函数,利用排序函数计算文档和查询的相关性分数,并以此为依据对文档集合进行排序。排序学习研究的核心问题是如何构造一个函数或模型反映文档对于查询的相关度。排序学习在信息检索中所定义的任务内容如下:给定文档的训练集合 D,其中每个文档表示为 $< q,d,r >$ 的三元组的形式,q 为查询;d 为文档特征集合 $\{f_1, f_2, \cdots, f_n\}$,这里的文档特征特是查询和文档的复合特征,一般为普通的检索方法如 BM25, 语言模型检索方法等;r 为文档与查询的相关性判断条件取值一般为 $\{0,1\}$,0 代表不相关,1 代表相关。测试集合用 T 表示,也以三元组 $< q,d,? >$ 形式表示,只有查询和文档特征集合两个元素,而文档的相关性未知。

收稿日期: 2011 年 1 月 12 日

作者简介: **林原**,男,博士生,研究方向:信息检索、排序学习。E-mail:yuanlin@mail.dlut.edu.cn。**林鸿飞**,男,博士,教授,博士生导师,研究方向:搜索引擎、文本挖掘和自然语言理解。

1) 基金项目:国家自然科学基金资助项目(编号:60673039,60973068)和国家 863 计划资助项目(编号:2006AA01Z151);教育部博士点基金(编号:20090041110002)和教育部出国留学人员归国启动基金。

排序模型就是由文档训练集合三元组训练得到,用于预测测试集文档相关性分数,进而计算文档相关性排名。按照训练样本的不同可以将排序学习算法大体分为三类:

Pointwise,将训练集中的每个文档看作一个样本获取 Rank 函数,主要解决办法是把分类问题转化为单个文档的分类和回归问题。应用于 pointwise 的算法主要有 Discriminative model for IR[2], McRank[3]等。

Pairwise,将训练集中有着不同的相关标注的两个文档看作是一个样本,基于文档对的方法的一个主要解决思想就是依靠训练集中的不同相关度的文档对把 rank 问题转化为二值分类问题。主要应用的算法有:Ranking SVM[4],RankBoost[5],RankNet[6]等。

Listwise,则是将整个文档序列看作是一个样本,主要是通过直接优化信息检索的评价方法和定义损失函数两种方法来获取损失函数。主要应用的算法有:AdaRank[7],SVM-MAP[8], ListNet[9],ListMLE[10]等。

三类方法中以 Listwise 方法效果为最好,近年来大多数排序学习方法的研究都集中在此种方法之上,在排序学习公开数据集 Letor[11]集合上取得了优异的排序结果,成为了该领域研究的热点。本文通过介绍这两种基于 Listwise 排序学习方法以及其相关的改进方法,展示排序学习方法的一般过程,以及损失函数构造方法。并通过对这些方法进行实验和研究提出一种新的损失函数的构造方法,继而进一步提高文档集合的相关性排序准确率。Listwise 方法按照优化的对象不同分为两部分:定义 listwise 损失函数方法(define listwise loss functions)以及直接优化信息检索的评价方法(directly optimize IR evaluation measuress),本文对于基于神经网络排序学习方法的介绍也按照这种模式进行介绍。

2 基于定义 Listwise 损失函数的神经网络排序学习方法

2.1 Luce 模型

构造 Listwise 方法损失函数首先要解决的问题就是如何表示文档列表,其中一种办法就是直接将文档列表表示成空间向量模式,每一个文档通过排序函数给出一个相关性分数,以这些分数构成一个相关性向量与文档集合的相关性标注向量计算相似度,选择使相似度最大的备选函数作为最终的排序函数,这种方法就是 RankCosine[12],这是一种最直接的文档序列表示方法,然而实验证明,这种朴素的方法并不是序列表示的最佳方法。ListNet 和 ListMLE 将 Luce[13]模型引入到排序学习方法之中来表示文档序列并取得更好的排序效果。而大多数基于神经网络的排序学习算法都是基于 Luce 模型来表示序列的排序方式的,在介绍这些方法之前首先介绍下 Luce 模型。

通过 Luce 模型,可以将序列的任意一种排列方式表示成一个概率值,假设有以下以数字 $1,2,\cdots,n$ 进行标识的一系列对象,π 为这些对象一种排列方式,而 Ω_n 为所有可能的对象的排列方式。如果有一个排序函数给每个对象指定一个分数则我们可以根据对象集合得到一个分数集合 $s = \{s_1, s_2, \cdots, s_n\}$,这里 s_j 是第 j 个对象的分数。假设 π 为这些对象一种排列方式,那么给定分数的 π 的概率可以定义为:

$$P_s(\pi) = \prod_{j=1}^{n} \frac{\Phi(s_{\pi(j)})}{\sum_{k=j}^{n} \Phi(s_{\pi(k)})} \tag{1}$$

这里我们定义 $s_{\pi(j)}$ 为在序列 π 中 j 位置对象的分数。Φ 是一个严格递增的正函数。我们假设一个实例中有三个对象{1,2,3},分数集合为 $s = \{s_1, s_2, s_3\}$。有两种排列方式 $\pi_1 = <1,2,3>$ 和 $\pi_2 = <3,2,1>$ 则可以根据式(1)用以下方式进行计算:

$$P_s(\pi_1) = \frac{\Phi(s_1)}{\Phi(s_1)+\Phi(s_2)+\Phi(s_3)} \cdot \frac{\Phi(s_2)}{\Phi(s_2)+\Phi(s_3)} \cdot \frac{\Phi(s_3)}{\Phi(s_3)} \tag{2}$$

$$P_s(\pi_2) = \frac{\Phi(s_3)}{\Phi(s_1)+\Phi(s_2)+\Phi(s_3)} \cdot \frac{\Phi(s_2)}{\Phi(s_2)+\Phi(s_1)} \cdot \frac{\Phi(s_1)}{\Phi(s_1)} \tag{3}$$

式(2)、式(3)中 $P_s(\pi_1)$、$P_s(\pi_1)$ 分别表示 π_1、π_2 的概率值。应用 Luce 模型使我们能够将一个序列的排序方式表示成一个单一的概率值。Luce 模型奠定了上述两种以排序列表为研究对象排序学习方法的基础,用 Luce 模型将原来的排序难以比较的序列排列方式量化,使其能以简单的方式进行比较计算。Φ 一个严格递增的正函数,实际应用过程中一般用 exp()函数来表示。Luce 模型能够将有序序列有效的表示成一个概率值,是因为它有以下优秀的性质:所有可能的排序方式的概率值的和等于1;基于排序函数的任意排序列表,若交换较高得分的对象与较低得分的对象的位置,将得到一个较低的概率值;对于一个给定的排序函数,基于该函数有

最高的排列概率值,而逆序的排序列表则为最低的概率值。

不仅很多 Listwise 方法都是以 Luce 模型为基础的,之前发表的一些 Pairwise 方法中构造偏序对的模型其实也是 Luce 模型,只是没有明确指出而已,如 RankNet[6],Frank[14],以及 Zhou[15]等发表的 Ties 对文章中,就是用 Luce 模型表示相关性不同的文档对(Zha 等同时应用该模型表示相关性相等的文档对),以 Luce 模型表示的概率文档对作为其研究和实验基础,构造所谓的"Pairs",因为有序的文档对也可以看做是一个仅有两个元素的 Luce 模型的实例,ListNet[9]和 ListMLE[10]则是完全显式地将 Luce 模型引入到排序学习的损失函数构造当中。

2.2 ListNet 排序学习方法

ListNet[9]方法最早将 Luce 模型引入到 Listwise 方法当中,它所采用的机器学习方法为神经网络模型。ListNet 方法就是一种应用概率模型来构造损失函数的排序学习方法,从 Letor3.0[11]的多个数据集上的测试来看,该方法的性能是比较稳定的,并取得了较好的结果。ListNet 方法开创了排序学习 Listwise 方法的先河。继该方法面世以后 Listwise 方法逐渐丰富起来,大体看来该方法主要分为两类:即 Listwise 损失函数方法和直接优化信息检索评价函数的方法如等。对于一组排序对象我们可以根据对象的评分函数应用 Luce 模型得到所有排列方式的概率值,从而得到每种评分函数的排列概率分布。这里的评分函数有两种,一种是相关性判断条件,另一种是排序函数,我们可以用 Luce 模型对这两种评分函数分别获得一个序列的概率分布,应用交叉熵来衡量这两个概率分布的相似性进而构造损失函数。该思想就是 ListNet 方法实现的基础。其损失函数可以表示为:

$$L(y^q,z^q) = -\sum_{q\in Q}\sum_{\pi\in\Omega}\left(\prod_{t=1}^{n}\frac{\Phi(y_{\pi(t)})}{\sum_{u=t}^{n}\Phi(y_{\pi(u)})}\right)\log\left(\prod_{t=1}^{n}\frac{\Phi(z_{\pi(t)})}{\sum_{u=t}^{n}\Phi(z_{\pi(u)})}\right) \quad (4)$$

式中,Q 为训练集中所有查询,Ω 为查询所对应的文档列表的所有排列方式,y 为相关性判断条件,z 为排序函数,$\pi(i)$ 为序列 π 中 i 位置的元素。该损失函数的特点是连续可微,并且是凸函数,误差损失在 0~1 之间。尽管 Listwise 损失函数有以上优点,但是计算量成为该算法的致命缺点,按照上述损失函数进行迭代计算算法的整体时间复杂度则为 $O(n*n!)$,而 n 动辄以千计,而 1000! 的计算量是难以想象的所以需要采取办法简化计算,近似优化算法。参考文献[9]中提出了一种行之有效的办法 Top-K 概率来解决这个问题。

Top-K 概率的核心思想简言以蔽之可以解释为用随机序列的前 k 项的排列概率来近似替代原有的整个序列的概率,虽然牺牲了一定的准确度,但是有效地节省了运算时间。前 k 个对象 $(j_1,j_2,\cdots,j_k)$ 概率如式(5)所示:

$$P_s(g_k(j_1,j_2,\cdots,j_k)) = \sum_{\pi\in g_k(j_1,j_2,\cdots,j_k)}P_s(\pi) = \prod_{t=1}^{k}\frac{\Phi(s_{j_t})}{\sum_{l=t}^{n}\Phi(s_{jl})} \quad (5)$$

式(5)的含义为,Top-K 概率为前具有相同的 k 项的序列的概率和,如果我们用 Luce 模型来估计 Top-K 概率,并将正项函数带入式(5)中间项,我们则可以得到式(5)末端项用以进行计算实际概率值。虽然我们通过 Top-K 概率优化了计算时间但是我们仍要计算 $n!/(n-k)!$ 次,为了近一步提高计算效率我们将 k 值设为 1 则可以更大幅度减少该算法的计算量。此时,算法的时间复杂度则可以降低到 $O(n)$。我们用指数函数替代正项函数重写式(5)可得:

$$P_s(g_k(j_1,j_2,\cdots,j_k)) = \prod_{t=1}^{k}\frac{\exp(s_{j_t})}{\sum_{l=t}^{n}\exp(s_{jl})} \quad (6)$$

对于一个查询 $q^{(i)}$,排序函数 f_ω 可以产生一个得分列表 $z^{(i)}$,那么 Top-K 概率我们可以按照以下公式进行计算

$$P_{z^{(i)}(f_\omega)}(g_k(j_1,j_2,\cdots,j_k)) = \prod_{t=1}^{k}\frac{\exp(f_\omega(x_{j_t}^{(i)}))}{\sum_{l=t}^{n(i)}\exp(f_\omega(x_{jl}^{(i)}))} \quad (7)$$

应用 Top-K 概率估计序列概率,并使用交叉熵表示损失函数,则 $q^{(i)}$ 的损失函数可以表示为:

$$L(y^{(i)},z^{(i)}(f_\omega)) = -\sum_{\forall g\in g_k}P_{y^{(i)}}(g)\log(P_{z^{(i)}(f_\omega)}(g)) \quad (8)$$

ListNet 方法算法流程为:

ListNet 采用梯度下降神经网络算法,损失函数相对于 ω 的梯度可以进行如式(9)计算:

$$\Delta\omega = \frac{\partial L(y^{(i)},z^{(i)}(f_\omega))}{\partial\omega} = -\sum_{\forall g\in g_k}\frac{\partial P_{z^{(i)}(f_\omega)}(g)}{\partial\omega}\frac{P_{y^{(i)}}(g)}{P_{z^{(i)}(f_\omega)}(g)} \quad (9)$$

ListNet:
Input: training data $\{(x_{(1)}, y_{(1)}), (x_{(2)}, y_{(2)}), ..., (x_{(m)}, y_{(m)})\}$
Parameter: number of iterations T and learning rate η .
Initialize parameter ω .
for t = 1 to T do
  for i = 1 to m do
    Input $x_{(i)}$ to Neural Network and compute score list $z^{(i)}(f_\omega)$ with current ω
    Compute gradient $\Delta\omega$
    Update $\omega = \omega - \eta \times \Delta\omega$
  end for
end for
Output Neural Network model

这里如果 $n^{(i)}$ 等于 2 的话,则 ListNet 就等同于一层神经网络的 RankNet。进一步如果我们将 $k=1$ 带入上式,则可以得到式(10)

$$\Delta\omega = -\sum_{j=1}^{n(i)} P_{y(i)}(x_j^{(i)}) \frac{\partial f_\omega(x_j^{(i)})}{\partial\omega} + \frac{1}{\sum_{j=1}^{n(i)} \exp(f_\omega(x_j^{(i)}))} \sum_{j=1}^{n(i)} \exp(f_\omega(x_j^{(i)})) \frac{\partial f_\omega(x_j^{(i)})}{\partial\omega} \quad (10)$$

这里采用一个不考虑 b 线性神经网络模型:

$$f_\omega(x_j^{(i)}) = \langle \omega, x_j^{(i)} \rangle \quad (11)$$

以上为 ListNet 算法的介绍,从与先前的算法进行对比的结果来看,ListNet 较大限度地提高了排序准确率,充分地体现了 Listwise 方法的有效性,实际上即使对于后续出现的算法,ListNet 方法仍是一种很有竞争力的对比方法。以 ListNet 方法为代表的 Listwise 方法的出现为排序学习方法的研究开辟了一个崭新的视野。

2.3 ListMLE 排序学习方法

ListMLE[10] 算法是另一种基于 Luce 模型的排序学习算法。虽然 ListMLE 算法是后于 ListNet 算法出现,但是从性能上看并不优于 ListNet, 仅仅是在 Letor 的 OHSUMED 数据集上,ListMLE 方法才要好于 ListNet。虽然如此,但这并不能影响 ListMLE 的许多优秀性质和可扩展性,有不少文章就是以 ListMLE 为基础对该算法进行改进的并取得不错的效果。

ListMLE 的损失函数的构造仍是以 Luce 序列概率模型为基础,但是与 ListNet 不同的是所采用的损失函数为极大似然损失函数。其具体表示形式为:

$$\phi(g(x), y) = -\log P(y \mid x; g) \quad where, P(y \mid x; g) = \prod_{i=1}^{n} \frac{\exp(g(x_{y(i)}))}{\sum_{k=i}^{n} \exp(g(x_{y(k)}))} \quad (12)$$

这里的 y 是按照文档的相关性判断条件降序排列的文档序列,我们对所有给定预测序列的结果采用参数化的指数概率分布,定义真实的排序列表为负的对数似然估计。似然损失函数有以下优点:首先似然损失 Listwise 损失是一致的(consistent),该损失函数是位置敏感的,其次似然损失函数是合理的。简单起见,假设有两个对象需要被排序。排序函数给定两个对象的分数为 g_1 和 g_2。这里实际上 g_1 应该排在 g_2 后面。那么图 1(a) 中 $g_2 = g_1$ 左边的部分应为正确排序区域,而右边的部分则是错误的排序区域。根据似然损失定义所有在直线 $g_2 = g_1 + d$ 上的点都具有同样的损失。因此我们认为似然损失主要依赖于 d, 图 b 则显示了损失函数与 d 之间的关系。我们可以看到损失函数在 d 增加的情况下,连续递减。函数对负值 d 的惩罚要大于对正值的惩罚。这样可以使学习算法更加重视避免不正确的排序。由此可见,该损失函数是一个较好的 0~1 序列损失的近似。

这里似然损失是连续的,可微的,凸函数的。进一步,损失函数在线性时间内可以进行有效的计算。有了这些优良的性质,这个学习算法能够有效的优化损失似然损失函数并且有能力创造一种排序函数。

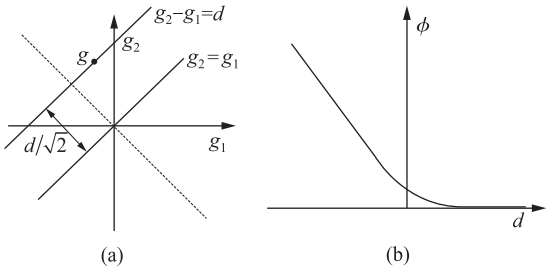

图 1　ListMLE 损失函数

ListMLE 算法就是一种基于似然损失函数的 Listwise 排序学习方法。在训练数据集上,我们最大化训练查询的似然的和。ListMLE 选择梯度下降算法作为构造最小化损失函数算法。我们采用线性神经网络作为排序模型。ListMLE 算法流程表示为:

上面我们对 ListMLE 方法进行了介绍,该方法基于神经网络算法,提出了一种新的损失函数构造模式,而且该算法的训练时间复杂度不用优化直接就是 $O(n)$, 在 Letor 的 OHSUMED 的数据集上取得了最好的排序结果。由此可见,ListMLE 对于先前的排序学习方法还是有一定的优势的。虽然如此,但是在其他数据集上 ListMLE 的表现情况不尽人意,这也给后续研究提供了进一步改进的空间,近年来的文献有不少都是对其进行改进的,并取得了很好的结果。

ListMLE:
Input: training data $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ..., (x^{(m)}, y^{(m)})\}$
Parameter: learning rate η , tolerance rate ε , Initialize parameter ω .
Repeat
　for i = 1 to m do
　　Input ($x^{(i)}$, $y^{(i)}$)to Neural Network and compute gradient $\Delta\omega$ with current ω
　　Update $\omega = \omega - \eta \times \Delta\omega$
　end for
　calculate likelihood loss on the training set
until change of likelihood loss is below ε
Output Neural Network model ω

2.4　基于 Listwise 损失函数改进方法

近年来,对 Listwise 损失函数排序学习方法研究[16,17]主要集中于两个方面,一种是从 Listwise 损失函数本身出发,寻找函数的不足和缺陷,对此进行改进,进而提高排序准确率。另一种是寻找损失函数与排序评价方法如 MAP[18],NDCG[19,20]等方法的联系,力求优化该损失的时候能够直接优化这些评价方法,以此得到更好的排序结果。

2.4.1　Top-K 框架下的 Listwise 方法

文献[17]也是以 ListMLE 为实例对 Listwise 方法进行研究的,它提出了一种基于 Top-K 框架的 Listwise 改进方法,Top-K 框架原本是为改善排序模型对前 K 项文档的排序准确率而设计实现的,但是从实验结果上来看这种思路不但能够有效的改善前 K 项文档的排序准确率,对于文档列表整体的排序准确率有能进行有效的提高。为了更好的介绍 Top-K 框架,首先定义列表的前 Top-K 项排序文档的损失,可表示为:

$$l_k(h(x),y) = \begin{cases} 0, if\ \hat{y}(i) = y(i)\ \forall i \in \{1,\cdots,k\}, \\ \quad where\ \hat{y} = h(x). \\ 1, \quad otherwise. \end{cases} \tag{13}$$

真实的 k 值是由实际应用所决定的,当 K 值等于整个排序序列的长度的时候,Top-K 损失函数就直接等于 Listwise 损失函数,从这个角度来看,Listwise 是 Top-K 方法的一个特殊情况。其期望风险可以表示为:

$$R_k(h) = \int_{X\times Y} l_k(h(x),y)\,\mathrm{d}P(x,y) \tag{14}$$

选择使使 Top-K 子序列的排列概率达到最高的中间结果作为最终的排序函数:

$$h_k^*(x) \in \mathrm{argmax}_{G_k(j_1,j_2,\cdots,j_k)\in G_k} P(G_k(j_1,j_2,\cdots,j_k)\mid x) \tag{15}$$

表1 Top-K ListMLE方法在OHSUMED数据集上的排序结果

	Map	NDCG@1	NDCG@3	NDCG@10	P@1	P@3	P@10
$K=1$	0.4290	0.4332	0.4082	0.3999	0.5164	0.5054	0.4739
$K=3$	0.4405	0.4588	0.4443	0.4252	0.5550	0.5601	0.4887
$K=10$	0.4441	0.5158	0.4772	0.4360	0.6304	0.5977	0.4945
ListMLE	0.4326	0.4196	0.4188	0.4191	0.5342	0.5540	0.4958

$G_k(j_1,\cdots,j_k)$ 表示Top-K子序列,这里指含有相同的前 K 项的序列。G_k 指集合中所有的序列。文献[17]给出了ListMLE方法的Top-K损失的替代函数获取的证明。根据上述参数设计,我们可以得到Top-K ListMLE损失函数:

$$P(y\mid x;g)=\prod_{i=1}^{k}\frac{\exp(g(x_{y(i)}))}{\sum_{k=i}^{n}\exp(g(x_{y(k)}))} \quad (16)$$

Top-K方法的核心思想是使损失函数对于训练集中的排在前 K 项的文档分数变化敏感,而对于排在第 K 项以后的文档分数变化迟钝,用这种办法训练出来的排序模型的对于测试集文档中的前 K 项排序准确率的预测能力会大大提高,文献[17]的实验结果表明这样改进ListMLE方法的损失函数的确有助于最终的前 K 项排序结果的提高。

本文在Letor3.0上的OHSUMED数据集合上再现了该方法,得到了Top-K ListMLE在该数据集上的实验结果如表1所示。

以上数据为三个数据集合的实验结果,可以看出,Top-K ListMLE与ListMLE相比随着 K 的增加,Top-K损失函数的性能越来越好,尤其是 $K=10$ 的时候取得了最好的效果。用Top-10 ListMLE与ListMLE,该方法在大多数情况下都取得了较好的效果,特别是对于N@1和P@1两个指标,提高幅度最大。从中可以看出这种根据实际应用意义对损失函数进行改造的方法对于排序学习方法排序准确率的提高是十分有意义的。虽然如此,但是Top-K矿框架仍有一些问题没有解释清楚,如为什么Top-K框架在 $K=10$ 的时候会取得较好的结果,K 从1~10变化的过程中排序函数的各项指标都在相应增加,而不是 $K=1$ 时,NDCG@1取得最好结果,$K=3$ 时NDCG@3取得最好结果,都是同时在 $K=10$ 时取到最好结果的,而继续增加到 $K=n$,n 为查询所对应的文档列表的长度,Top-K框架的结果就会和普通的Listwise方法的结果一致,即性能会再次变差,这些问题都是本文以下部分研究的重点,研究的目标是通过对这些问题的研究,进一步改进已有Listwise方法,取得更好的排序性能。

2.4.2 基于相关性敏感的Listwise方法

我们在Top-K框架的基础之上对Listwise进行研究,寻找Top-K方法提高排序准确率的原因,借此进一步提高排序准确率。该文提出了一种基于相关性敏感的思想来优化似然损失函数,解决ListMLE以及Top-K框架下的ListMLE方法所存在的问题,进而提高了排序准确率。

首先,通过一个例子了解ListMLE排序函数的选取过程,从而找出ListMLE方法的不足之处。迭代过程中产生的两个排序函数如表2所示。

表2 排序函数评分样例

Doc No.	1	2	3	4	5	6
Label	1	1	1	0	0	0
$\exp(f_1(x))$	0.3	0.2	0.1	0.1	0.2	0.1
$\exp(f_2(x))$	0.3	0.2	0.1	0.2	0.2	0.1

如表2所示,对于一个查询有6个文档,Label代表文档与查询的相关性,文章的索引顺序刚好是按照文档相关性进行排序的一种最佳排列方式,f_1,f_2 是ListMLE在迭代过程中产生的排序函数。f_1 在 f_2 之前产生,通过ListMLE损失函数进行计算可知 f_1 的似然损失为5.8579,f_2 的损失为5.7791。这里出现了一个严重的问题,排序函数 f_2 损失相对于 f_1 降低了。f_2 使不相关文档4的分值升高了,整个损失函数因为不相关文档4的分数的提高而降低,作为主程序选择排序函数的机制,因为损失函数降低会自动选择 f_2 作为当前的排序函数,如果是迭代的最后一步,那么 f_2 将成为最终的排序函数,而我们知道实际上 f_2 的排序效果并不如 f_1,所以由此可见,在ListMLE迭代产生排序函数的过程中还是存在一定问题的,主要是因为不相关文档分数的提高也可能

使损失函数降低,这就是 ListMLE 方法所存在的主要问题。

从上述 Top-K ListMLE 的实验不难看出,Top-K 框架可以对 ListMLE 方法进行进一步改进。如表 1 所示,这种改进不仅仅体现在 P@ n 和 NDCG@ n 这些评价列表前几位文档排列顺序的评价指标的改进,同时对于序列整体的排序顺序也有一定的改进。表 1 中所记录的实验为 Letor3.0 中的数据集。如该表所示,随着 K 值由 1 ~ 10 不断的变化,排序评价指标都在增加,尤其是序列的整体排序准确率也得到了有效的改善。可以 Top-K 框架可以有效的改善 ListMLE 的排序准确率,可以认为是 ListMLE 的一种有效的改进方法,同时这组数据也暗示,Top-K 框架也同时存在两个问题,首先 Top-K 框架提出伊始是以提高前 K 项文档排序准确率为基础的然而从实验结果上我们不难看出,表中所示各项指标都是在 $K=10$ 的时候获得了最好的表现情况,而并非是 $K=1$ 时 N@1 结果最好,$K=3$ 时 N@3 结果最好,而增长趋势显示 K 值越大排序结果会越好。这就引出了第二个问题,K 值多大的时候才能获得最好的表现情况,因为按照上述对 Top-K ListMLE 方法的说明我们可以了解到 K 值选取的极限情况是 K 等于文档列表长度的情况,这时的 Top-K ListMLE 是等同于 ListMLE 方法,但是这时的算法表现情况是很差的。我们由上述问题可以得出结论,取得一定大小的 K 能够使算法表现情况变好,但是 K 大到一定程度就是表现情况变差。所以我们再来思考问为什么 Top-K 框架会优化 ListMLE。和 ListMLE 相比 Top-K 框架唯一的不同之处就是 K 变小了,再从 ListMLE 本身来看,它的目标序列恰好是训练集中的排序列表按照正确的顺序进行排序的一种排列方式,而如果我们取前 1 项,3 项和 10 项,这些位置之前的文档大多会是相关文档,这样就会在一定程度上降低上文所说的不相关文档对于损失函数的负面影响,所以有此可见 Top-K ListMLE 在一定程度上克服 ListMLE 方法所存在的问题。

所以 K 值应该与对应每个查询的相关文档数有关,K 值的最佳选择应为每个查询所对应的相关文档个数,以此本文提出一个相关性文档敏感算法用于改善最终的排序效果。算法主要思想是,将每个用于训练的查询所对应的文档序列构造似然损失函数的时候 K 值取序列中相关文档的长度,这样似然损失函数只会因为相关文档的分数增加而降低,而不相关文档分数增加的时候,损失函数会相应的降低。但是此时仍然存在问题,如果文档集合的标注不只是(0,1),而是多种标注情况如(0,1,2),这种情况如果只采用两种标签相关 1 或不相关 0 作为相关文档则会很大程度上浪费标签相对信息。针对此种情况,设计出一种新的文档序列样本组样本:一组相关性高的文档和一组相关性较低的文档。例如,如果相关性标注为(0,1,2),那么我们就能构造出三种样本(Group(2),Group(1)),(Group(2),Group(0)),(Group(1),Group(0)),这里的 Group()代表与一个查询相关的等相关性的所有文档。通过构建这种文档我们就能够充分的利用多标签信息改善算法结果,对于这种样本我们构造应用 Top-K 框架对损失函数进行优化,这里 K 值选取为相关性较高的文档的个数,这样可以使损失函数对相关性较高的文档分数变化更为敏感,从而达到提高排序结果的目的。这样可以使损失函数对相关性较高的文档分数变化更为敏感,从而达到提高排序结果的目的。我们称这种方法为相关性敏感算法(R-sensitive),同时这种以组为训练样本的方式也将是我们以后研究的一个重要方向。重写后的基于相关性敏感似然损失损失函数可以表示为下式:

$$L(f;x^g,y^g)=\sum_{s=1}^{r}\left(-f(x^g_{y^g(s)})+\ln\left(\sum_{i=s}^{n}\exp(f(x^g_{y^g(i)}))\right)\right) \quad (17)$$

这里 x^g 表示组样本,而 y^g 为按照相关性判断条件的最佳排序方式,而 $x^g_{y^g(i)}$ 则表示 y 序列中 i 位置所对应的文档。该方法在 Letor3.0 的 OHSUMED 数据集和 TD2004 数据集上的实验结果如表 3 所示,实验集构造方法与文献[17]一致。

从实验结果来看,该方法(R-sensitive)相对于现存的 ListMLE 和 Top-10 ListMLE 排序结果有显著的提高。所采用的四个评价指标 MAP,NDCG@1,NDCG@3 和 NDCG@10 在 Group-group 方法在两个数据集上都取得了较好的结果,从中我们可以看到 R-sensitive p 方法不但改进了前 K 项排序结果,对序列整体的排序准确率的改善也起着很大的作用,该实验证明了 R-sensitive 方法的有效性。表 3 还将该算法与其他排序学习方法进行比较,对比方法包括 Listwise 方法 SVMMAP[8], ListNet[9], AdarankMAP[21], AdarankNDCG[21],以及 Pairwise 方法 RankSVM[4],和典型的 Pointwise 方法 Regression,其中 ListNet 方法是 Letor3.0 数据集所提供的 Baseline 方法中性能最好,在各个数据集上表现最稳定的方法。这两个

表3 排序方法结果比较

Methods	OHSUMED				TD2004			
	MAP	NDCG@1	NDCG@3	NDCG@10	MAP	NDCG@1	NDCG@3	NDCG@10
Regression	0.4220	0.4456	0.4426	0.4110	0.2078	0.3600	0.3352	0.3031
RankSVM	0.4334	0.4958	0.4207	0.4140	0.2237	0.4133	0.3467	0.3078
ListNet	0.4457	0.5326	0.4732	0.4410	0.2231	0.3600	0.3573	0.3175
SVMMAP	0.4453	0.5229	0.4663	0.4319	0.2049	0.2933	0.3035	0.2907
AdrankMAP	0.4487	0.5388	0.4682	0.4429	0.2189	0.4133	0.3757	0.3285
AdrankNDCG	0.4498	0.5330	0.4790	0.4496	0.1936	0.4267	0.3688	0.3163
ListMLE	0.4326	0.4196	0.4188	0.4190	0.1485	0.2222	0.2193	0.2362
Top-10	0.4441	0.5156	0.4772	0.4360	0.2214	0.4133	0.3414	0.3135
R-sensitive	0.4517	0.5589	0.4888	0.4572	0.2386	0.4267	0.3424	0.3175

数据集都是比较有代表性,OHSUMED 的标注方法是(2,1,0);而 TD2004 则是(1,0)相关度逐渐减低。与现有的排序学习方法进行比较的可以发现,R-sensitive 算法在不同数据集上都取得了较好的结果。这两个数据集上的各种方法的实验结果表明 R-sensitive 算法对于改善相关文档排序准确率是十分有效的。同时实验结果显示,对不同相关性标注数据集的改进效果不同,对于多标注相关数据集,该算法对于改进部分序列(如 Top-K)的排序准确率更为有效,而对于仅有两种标注:相关和不相关的数据集则对全序列排序准确率的改善效果更为显著。

从上面的介绍我们可以看到,对于已有方法进行改进的研究还是十分有效的,在解决了已有算法所存在的问题的同时,可以显著地提高排序学习方法的实验结果,进而促进信息检索准确率的提高。

2.4.3 直接优化损失函数的方法

Chen 等[17]从损失函数和评价方法的关系上对排序学习方法进行改进,尤其是对于 ListMLE 方法有了更大的提高和改进。他们的主要思想是如果能将排序学习方法的损失函数直接与排序评价指标建立关系,最小化损失函数就等于最大化这些评价指标,这样的构造新的损失函数会使排序学习方法的学习更有针对性。他们通过引入了一个中间变量来沟通评价指标和损失函数。通过一个简单例子介绍这个中间变量。

如图2所示,假设这里有三个对象 A,B,C 和一个正确的排序顺序 $y=\{A,B,C\}$。如果排序函数的

$$\begin{pmatrix} y \\ A \\ B \\ C \end{pmatrix} \quad \begin{pmatrix} \pi \\ B \\ A \\ C \end{pmatrix} \xrightarrow[remove\ A]{incorrect} \begin{pmatrix} y \\ B \\ C \end{pmatrix} \quad \begin{pmatrix} \pi \\ B \\ C \end{pmatrix} \xrightarrow[remove\ B]{correct} \begin{pmatrix} y \\ C \end{pmatrix}$$

图2 样例序列分析

输出顺序为{2,3,1},因此预测排序序列为 $\pi=\{B,A,C\}$。在第一个分解阶段排序函数预测对象 B 应该在列表顶上,可是,实际上 A 应该在列表第一位,因此产生了预测错误,在第二阶段,移除 A,这时排序函数仍然预测 B 为列表第一位对象,这时根据正确的排列顺序 y 这次没有预测错误。之后我们移除对象 B,很容易证明第三步没有预测错误。整体来看,排序函数在分类任务中出了一个错误。我们制定一个非负权重 $\beta(s)(s=1,\cdots,n-1)$ 给每步的分类任务,代表它在整个序列的重要性。我们计算每次分类任务的权重和,计算公式如下:

$$L_\beta(f;x,y) \triangleq \sum_{s=1}^{n-1}\beta(s)\left(1-\prod_{i=s+1}^{n} I(f(x_{y(s)})>f(x_{y(i)}))\right) \tag{18}$$

这里,I 为指示函数,如满足不等关系则为1,不满足则为0,从式(18)我们可以得出结论,只要有一对文档偏序错误,则会产生分类错误,定义该损失的最小值为序列的必要损失函数,得到了必要损失函数,可以得到 MAP 和 NDCG 与必要损失函数的关系[16]:

$$1-MAP(f;x,\zeta) \leqslant \frac{1}{\sum_{i=k}^{K} n_i} L_\beta(f;x,\zeta), where \quad \beta(s)=1 \tag{19}$$

$$1 - NDCG(f;x,\zeta) \leqslant \frac{1}{N_n} L_\beta(f;x,\zeta), where$$

$$\beta(s) = G(l(y(s)))D(s), \forall y \in Y_\zeta \quad (20)$$

式(19)中，$L_\beta(f;x,\zeta)$ 为必要损失函数，$\sum_{i=k}^{K} n_i$ 为相关文档总数，式(20)中 N_n 为 NDCG 标准化因子，G 为 NDCG 的增益函数，D 则为其位置折扣函数。这样我们就建立起了必要损失函数必要损失函数和评价方法之间的联系。

同时我们也可以得到必要损失函数与 Listwise 损失函数之间的关系[16]为：

$$L_\beta(f;x,\zeta) \leqslant \frac{1}{\ln 2}\left(\max_{1\leqslant s\leqslant n-1}\beta(s)\right)L^l(f;x,y), \quad \forall y \in Y_\zeta \quad (21)$$

式中，$L^l(f;x,y)$ 为 Listwise 损失函数，这样就可以以必要损失函数为纽带建立评价函数和 Listwise 损失函数的关系，可表示为：

$$1 - MAP(f;x,\zeta) \leqslant \frac{1}{\ln 2 \sum_{i=k}^{K} n_i} L^l(f;x,y), \forall y \in Y_\zeta \quad (22)$$

$$1 - NDCG(f;x,\zeta) \leqslant \frac{G(K-1)D(1)}{N_n \ln 2} L^l(f;x,y), \forall y \in Y_\zeta \quad (23)$$

这样，我们就表示出评价方法和 Listwise 损失函数之间的关系，通过式(22)和式(23)我们可以看到最小化 Listwise 损失函数就相当于最大化排序评价指标，与普通的 Listwise 方法相比，新的 Listwise 损失函数多了项查询依赖的变量。这样通过建立二者的关系优化了 Listwise 损失函数，对应的新损失函数越小，评价指标 MAP，NDCG 的值越高。文献[16]以此方法对 ListMLE 方法进行改进，在 OHSUMED 数据使该方法的排序性能的得到了进一步的提高。

实验证明这种建立 Listwise 损失函数与评价排序方法指标关系的方法重新构造损失函数的方法的确有助于 ListMLE 排序性能的提高，这也是一种对 Listwise 损失函数构造的合理性的证明，这个尝试对以后相关问题的研究奠定了一定的基础。同时，该方法也在一定程度上沟通了原来两种看似没有联系的 Listwise 方法：定义损失函数方法以及直接优化信息检索的评价方法。使我们认识到这两种方法还是有内在联系的，按照描述序列的状态之间的相似性的方法定义损失函数的方法：如交叉熵损失函数和似然损失函数，在一定程度上都对信息检索评价方法进行了优化。

上面我们通过介绍两种经典的定义 Listwise 损失函数的排序学习方法 ListNet，ListMLE 以及一系列与它们相关的改进方法，这些改进算法不仅仅他们在算法上有依存关系，而且在训练过程中都采用了神经网络方法作为模型训练的基础，并且可以看出这些算法充分的利用了神经网络方法构造损失函数的灵活性，构造了新的损失函数并且取得了较好的实验结果。

3 基于直接优化信息检索评价方法的神经网络排序学习方法

在 2.4.3 节我们介绍了实际上 Listwise 方法的两个研究分支是有内在联系的，进而引出了 Listwise 方法研究的另一分支，直接优化信息检索评价方法。这是一个很直观的想法，之前定义的损失函数主要是为了计算预测的查询文档序列和目标序列的相似性，两个序列越相似那么损失函数的值越小，进而排序模型的预测能力越高，但是实际上一个排序算法的好坏一般不是通过计算两个序列的相似性来判断的。信息检索领域有自己的完善的评价系统，如 MAP，NDCG 和 P@ N 等，这些指标能够最直观的体现排序算法性能的好坏。因此，如果对果能构造损失函数直接优化这些评价标准，可能会取得事半功倍的效果，出于这种考虑，出现了对直接优化评价函数的研究。基于神经网络机器学习方法除了能够实现 Listwise 方法中的定义 Listwise 损失函数方法之外，也可以很容易的实现该方法的另一分支直接优化信息检索评价函数方法，比较有代表性的方法就是 SoftRank[22]，LambdaRank[23]。本节我们将对基于直接优化信息检索评价方法的神经网络排序学习方法进行介绍。

3.1 SoftRank 排序学习方法

SoftRank[22]用随机分布表示文档排序分数，这样可以突破以往只将排序分数看作室单一数值的限制，而是将分数转化为一个概率分布，如此排序分数大小的比较就变成了对分数偏序关系的概率估计。以此为基础对评价方法 NDCG 进行近似期望估计，进而将不平滑不可微的评价方法转化为可以用梯度下降方法进行优化的损失函数。

首先，SoftRank 定义了概率分布，给定一组与查询 q 相关的文档 $x = \{x_1, x_2, \cdots, x_j\}$。文档相对于查询的相关性分数不再是一个数值而是一个随机变量。这个随机变量时由高斯分布定义的，该分布的方

差为 σ_s,均值为 $f(x_j)$,$f()$ 为文档的相关性判断分值。这样我们就可以用高斯分布表示文档的相关性分值,具体表达形式如下:

$$p(s_j) = N(s_j \mid f(w,x_j),\sigma_s^2) \quad (24)$$

这里高斯分布采用以下形式:

$$N(x \mid \mu,\sigma) = (2\pi\sigma^2)^{-0.5}\exp[-(x-\mu)^2/2\sigma^2] \quad (25)$$

以往我们获取排序序列的方法是得到每个文档的相关性判断分数,并按照大小关系对其进行排序,分值越大的分数他的排名越靠前,反之则越靠后。然而,对于每个文档所对应的高斯分布我们是不能用这种方法计算排序序列的。对于一个文档 x_j 我们考虑另外一个文档 x_i 排在它之前的概率进而计算它的排名。我们用 S_j 表示 $p(s_j)$,我们用 $Pr(S_i - S_j > 0)$ 表示 x_i 排在 x_j 之前的概率。这里定义这个概率为 S_i 和 S_j 两个高斯分布差值的积分[24],用 π_{ij} 这样表示文档 x_i 排在 x_j 之前的概率:

$$\pi_{ij} = Pr(S_i - S_j > 0) = \int_0^{\infty} N(s \mid f(x_i) - f(x_j), 2\sigma_s^2)\,ds \quad (26)$$

这样我们就得到了两个文档之间的概率分布偏序关系,极端的情况,$\pi_{ij} = 1$ 或 $\pi_{ij} = 0$,这是由原始的 $f(x_i)$ 和 $f(x_j)$ 接近程度决定的,相对的 $f(x_i)$ 越大则 π_{ij} 值越大,反之则越小。SoftRank 就是通过这种方法"软化"了文档间的偏序关系,进而软化了每个文档的位置。以此为基础可以获得文档 x_j 的位置概率分布。可以认为排序列表的中的文档是逐次加入到序列当中的,假设文档 x_j 的是排序列表中的最初文档,那么新加入文档 x_i 后,如果文档 x_i 的相关性分数于 x_j 的,则 x_j 的文档位置加1,反之,x_j 的文档位置不变。用数学表达式表示在加入文档 x_i 后 x_j 在位置 r 的概率为:

$$p_j^{(i)}(r) = p_j^{(i-1)}(r-1)\pi_{ij} + p_j^{(i-1)}(r)(1-\pi_{ij}) \quad (27)$$

当文档列表中的最后一个文档加入到迭代过程以后就得到了文档 x_j 被排在位置 r 的概率最终概率 $p_j(r)$,有了这个概率就可以对 NDCG 进行进一步重写近似替代了。

最后,通过排序分布 SoftRank 可以计算出 $NDCG@m$ 的数学期望,这里 m 为与查询相关的文档的个数。用这个数学期望作为排序学习方法的目标函数。实际上就是用 1-SoftNDCG 作为 SoftRank 方法的损失函数,最小化 1-SoftNDCG 就相当于最大化 NDCG 的值,这里 SoftRank 表示为:

$$SoftNDCG = Z_m \sum_{j=1}^{m} (2^{y_i} - 1) \sum_{r=0}^{m-1} D(r) p_j(r) \quad (28)$$

其中 Z_m 为 $NDCG$ 标准化因子,$D(r)$ 为位置折扣因子。为了学习排序模型最大化 Soft -NDCG,SoftRank 采用神经网络机器学习方法,利用梯度下降作为优化算法。

该方法在 TREC 的.GOV 等数据集上取得了较好的效果。但是仍存在一些问题,如为什么采用高斯分布作为随机变量,文章中并没有详细的解释和证明,同时,作为高斯分布的重要参数 σ,文章中只是一个预先定义的常量,而不是通过训练学习的到的。尽管如此该文也提供了一种如何构造直接优化评价方法新的思路,同时也让我们意识到该方法的两个重要的研究问题即如何构造一个可微的替代函数来近似信息检索评价方法,以及如何对这个函数中的参数进行估计,这两个因素对于排序模型的性能起着决定因素。

SoftRank 是一种直接通过寻找近似函数来替代 NDCG 方法排序学习方法,下面我们将介绍一种,通过对已有的排序学习方法进行变换,进而得到直接优化评价方法目标函数的排序学习方法 LambdaRank。

3.2 LambdaRank 排序学习方法

LambdaRank[23] 是一种 Pairwise 方法 RankNet 的 Listwise 改进版本,通过重新定义 RankNet 损失函数的梯度方式对 NDCG 评价指标进行优化。交换排序列表中的两个不同相关性的文档会对最终的排序效果产生影响,假设相关性较高的文档排在相关性较低的文档后面,如果我们将相关性较高的文档提到相关性较低的文档前面,我们就会提高排序准确率,进而优化信息检索评价方法,LambdaRank 就是基于这种考虑完成对 RankNet 损失函数的梯度重写的,进而提高排序准确率的。

一个 LambdaRank 的梯度,λ_j,是对在排序列表中 j 位置的文档分数的目标损失梯度的一种平滑近似。对于一个查询中的两个相关不同的文档,他们的 λ-梯度是等值但是方向是相反的。正方向的梯度可以将文档推向排序列表顶端,而负方向梯度则会将文档推向排序列表底端。选择一个合适的 λ-梯度,目标评价方法的梯度可以对一个给定文档进行平滑近似。LambdaRank 选择对 NDCG 方法进行优化构造 λ-梯度。

若对于 RankNet 中的每个文档偏序对的损失为 $C(s_i,s_j)$,则目标总损失为:$C_T = \sum_{\{i,j\} \in P} C(s_i,s_j)$,则目标损失对于网络权重的微分为:

$$\frac{\partial C_T}{\partial w_k} = \sum_{\{i,j\} \in P} \frac{\partial C(s_i, s_j)}{\partial s_i} \frac{\partial s_i}{\partial w_k} + \frac{\partial C(s_i, s_j)}{\partial s_j} \frac{\partial s_j}{\partial w_k} \quad (29)$$

这里 P 为所有有效偏序对 $\{i, j\}$ 的集合，设 P_i 为所有可以与文档 i 构成有效偏序对的文档 j 的集合，进一步我们可以将式(29)重写为：

$$\frac{\partial C_T}{\partial w_k} = \sum_{i \in D} \frac{\partial s_i}{\partial w_k} \sum_{j \in P_i} \frac{\partial C(s_i, s_j)}{\partial s_i} \quad (30)$$

此时我们用 λ_i 代替 $\sum_{j \in P_i} \frac{\partial C(s_i, s_j)}{\partial s_i}$，则可以用 λ-梯度重写网络权重的微分。

为了定义进一步 λ-梯度我们首先给出 RankNet 损失函数的定义：

$$C_{ij} = C(o_{ij}) = -S_{ij} o_{ij} + \log(1 + e^{S_{ij} o_{ij}}) \quad (31)$$

RankNet 用交叉熵损失函数定义两个文档对偏序损失，其中 $o_{ij} = s_i - s_j$ 代表两个文档的分数差异，而 S_{ij} 则定义为：

$$S_{ij} = \begin{cases} +1, & l_i > l_j \\ -1, & l_i < l_j \end{cases} \quad (32)$$

其中 l_i 为文档 i 的相关性判断条件。由此可以得到 RankNet 损失函数对与文档相关性得分的微分：

$$\partial C_{ij} / \partial o_{ij} = \partial C_{ij} / \partial s_i = -S_{ij} / (1 + e^{S_{ij} o_{ij}}) \quad (33)$$

此时我们对上式的梯度用 λ-梯度近似重写，使梯度向着整体 NDCG 值升高的趋势变化：

$$\lambda_{ij} = S_{ij} \left| \Delta NDCG \times \frac{\partial C_{ij}}{\partial o_{ij}} \right| \quad (34)$$

进一步可以得到：

$$\lambda_{ij} = S_{ij} \left| N(2^{l_i} - 2^{l_j}) \left(\frac{1}{\log(1 + r_i)} - \frac{1}{\log(1 + r_j)} \right) \left(\frac{1}{1 + e^{S_{ij} o_{ij}}} \right) \right| \quad (35)$$

这里 N 是 DCG 的标准化因子。r_i 和 r_j 分别代表文档 i，j 在排序列表中所处的位置。而 S_{ij} 则则只依赖于文档 i，j 的相关性标注，而不依赖于他们的位置，若 $l_i > l_j$，那么文档 i 比文档 j 更加相关，文档 i 应该排的更靠前以减小损失，所以 $S_{ij} = 1$，λ-梯度对于文档 i 是正的。则上述的 $\lambda_i = \sum_{j \in P} \lambda_{ij}$。这样就完成了 λ-梯度的定义，由于 LambdaRank 在测试数据集上[23]取得了良好的实验结果，所以相关学者对于 λ-梯度进行了进一步的研究，文献[25]给出了 λ-梯度对于其它信息检索评价方法的定义。文献[25]分别给出了 MAP 及 MRR 的 λ-梯度。首先我们可以应用 ΔAP 代替式(34)中的 ΔNDCG，完成 MAP 的 λ-梯度的定义。如果 $r_i > r_j$，$l_i > l_j$，那么 $S_{ij} = 1$；我们可以得到：

$$\lambda_{ij} = \left| \frac{1}{R} \left(\sum_{k=r_j}^{r_i} l(k) P@k - \sum_{k=r_j}^{r_i} l'(k) P'@k \right) \left(\frac{1}{1 + e^{o_{ij}}} \right) \right| \quad (36)$$

这里 $l(k) = 1$ 当位置 k 的文档是相关文档的时候，如果不相关则为 0；$P@k$ 是在 k 位置的准确率；R 是查询的相关文档数。$l'(k)$ 是位置 r_i 和 r_j 交换位置后的相关性值。实际上，$l'(k) = l(k)$ 当 $k \in \{r_j + 1, \cdots, r_i - 1\}$ 时，同时这些位置的相关性标注也是相等的。$P@k$ 是在 r_i 和 r_j 交换位置后准确率。我们可以对式(36)进行重写：

$$\lambda_{ij} = \left| \frac{1}{R} \left[\left(\frac{n+1}{r_j} - \frac{m}{r_i} \right) + \sum_{k=r_j+1}^{r_i - 1} \frac{l(k)}{k} \right] \left(\frac{1}{1 + e^{o_{ij}}} \right) \right| \quad (37)$$

这里 n 和 $m(n <= m)$ 分别为前 r_i 和 r_j 位置的相关文档数。这样就完成了 MAP 的 λ-梯度的定义。

除此之外，文献[25]还给出了 MRR 的 λ-梯度，假设文档 i 是相关文档而 j 不是，则 $l_i > l_j$ 并且 $S_{ij} = 1$，则：

$$\lambda_{ij} = \left| \Delta RR(r_i, r_j) \left(\frac{1}{1 + e^{o_{ij}}} \right) \right| \quad (38)$$

$\Delta RR(r_i, r_j)$ 交换 r_i 和 r_j 文档位置前后的 MRR 的差值得到：

$$\Delta RR(r_i, r_j) = \begin{cases} \frac{1}{r_j} - \frac{1}{r}, & r_j < r \leqslant r_i \\ 0, & otherwise_i \end{cases} \quad (39)$$

这里 r 是排序列表中第一个相关文档的位置，很明显，如果交换后第一个相关文档的位置不变则没 MRR 损失。自此完成了对 MRR 的 λ-梯度的定义。

通过文献[25]的实验结果可以看到，对于不同评价方法定义对应的 λ-梯度，会对该方法最终在测试集合中指标值的提高，有促进左右，尤其是 NDCG 和 MAP 的 λ-梯度在测试集的 NDCG 和 MAP 指标分别取得了最好的结果。

自此完成了基于神经网络的直接优化信息评价方法的排序学习方法的介绍，实际上这类方法大致分为两种，一种是直接显示的定义信息检索评价方法的替代函数如 SoftRank 而另一种则是避开直接定义目标函数，而是直接定义与评价方法相关的梯度，如 LambdaRank，这两类方法都是该类方法研究的主要方向，尤其是定义信息检索评价方法的替代函数

进行优化的思想,随着替代函数与信息检索评价方法近似程度的加深,能够更加显著的提高最终的排序效果。

4 总 结

本文详细介绍了当今排序学习领域以神经网络为基础的多种重要算法,以及对这些方法的一些改进策略,希望通过这些方法的介绍使读者能够对当今排序学习方法的算法和改进策略有一定的了解。本文着重介绍了该领域两类重要的方法:定义Listwise损失函数以及直接优化信息检索评价方法,在对前人工作的研究和总结的基础上对其存在的问题进行阐述和分析,对改进方法的介绍显示了排序学习算法思想的传承和一致性,介绍了对排序学习方法进行改进的一般思路,希望本文能够对于今后排序学习问题研究有一定的促进作用,使排序学习准确率在此基础上更进一步。

参 考 文 献

[1] Liu T Y. Learning to rank for information retrieval[J]. Foundations and Trends in Information Retrieval,2009,3(3):225-331.

[2] Nallapati R. Discriminative models for information retrieval [C]// SIGIR Proceedings of the 27th International Conference on Research and Development in Information Retrieval,Sheffield,UK,2004:64-71.

[3] Li P,Burges C,Wu Q. McRank:Learning to rank using multiple classi? cation and gradient boosting [C]// Proceedings of the 21st Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia,Canada,2007:845-852.

[4] Joachims T. Optimizing search engines using clickthrough data [C]// Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,Edmonton,Alberta,Canada 2002:133-142.

[5] Freund Y,Iyer R,Schapire R,et al. An efficient boosting algorithm for combining preferences [J]. Journal of Machine Learning Research,2003,4:933-969.

[6] Burges C J,Shaked T,Renshaw E,et al. Learning to rank using gradient descent [C]// Proceedings of the 22nd International Conference,Bonn,Germany,2005:89-96.

[7] Xu J,Li H. AdaRank:A boosting algorithm for information retrieval [C]// Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam, The Netherlands,2007:391-398.

[8] Yue Y,Finley T,Radlinski F,et al. A support vector method for optimizing average precision [C]// Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam,The Netherlands,2007:271-278.

[9] Cao Z, Qin T, Liu T Y, et al. Learning to rank: From pairwise approach to listwise approach [C]// Proceedings of the 24th International Conference on Machine Learning,Corvallis,Oregon,USA,2007:129-136.

[10] Xia F,Liu T Y,Wang J. Listwise approach to learning to rank-theory and algorithm [C]// Proceedings of the 25th International Conference on Machine Learning, Helsinki,Finland,2008:192-199.

[11] Liu T Y,Qin T,Xu J,et al. LETOR:Benchmark dataset for research on learning to rank for information retrieval [C]// SIGIR2007 Workshop on Learning to Rank for IR,Amsterdam,The Netherlands,2007:3-10.

[12] Qin T,Zhang X D,Tsai M F,et al. Query-level loss functions for information retrieval [J]. Information Processing and Management,2008,44(2):838-855.

[13] Luce R D. Individual Choice Behavior [M]. New York: Wiley,1959.

[14] Tsai M F,Liu T Y,Qin T,et al. FRank:A Ranking Method with Fidelity Loss[C]//Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam,The Netherlands,2007:383-390.

[15] Zhou K,Xue G R,Zha H Y,et al. Learning to Rank with Ties.//Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval,Singapore,2008:275-282.

[16] Xia F,Liu T Y,Li H. Statistical consistency of Top-K ranking [C]// Proceedings of 23rd Annual Conference on Neural Information Processing Systems, Vancouver, B. C. ,Canada,2009:2098-2106.

[17] Chen W,Liu T Y,Lan Y,et al. Ranking Measures and Loss Functions in Learning to Rank [C]// Proceedings of 23rd Annual Conference on Neural Information Processing Systems, Vancouver, B. C. , Canada, 2009: 315-323.

[18] Baeza-Yates R, Ribeiro-Neto B. Modern Information Retrieval [M]. Massachusetts:Addison Wesley,1999.

[19] Jarvelin K,Kekalainen J. IR evaluation methods for retrieving highly relevant documents [C]// Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval,J Athens,Greece,2000:41-48.

[20] Jarvelin K, Kekalainen J. Cumulated gain-based evaluation of IR techniques [J]. ACM Transactions on Information Systems, 2002, 20: 422-446.

[21] Xu J, Li H. AdaRank: a boosting algorithm for information retrieval [C]// Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam, The Netherlands, 2007: 391-398.

[22] Taylor M, Guiver J, Robertson S, et al. SoftRank: Optimising Non-Smooth Rank Metrics [C]// Proceedings of the International Conference on Web Search and Web Data Mining, WSDM 2008, Palo Alto, California, USA, February 11-12, 2008: 77-86.

[23] Burges C J C, Ragno R, Le Q V. Learning to Rank with Nonsmooth Cost Functions [C]// Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006: 193-200.

[24] Papoulis A. Probability, Random Variables and Stochastic Processes [M]. 3rd. , New York: McGraw-Hill, 1991.

[25] Donmez P, Svore K M, Burges C J C. On the local optimality of LambdaRank [C]// Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, 2009: 460-467.

（责任编辑　王建平）

基于神经网络的Listwise排序学习方法的研究

作者： 林原， 林鸿飞， Lin Yuan， Lin Hongfei
作者单位： 大连理工大学计算机科学与技术学院,大连,116024
刊名： 情报学报 ISTIC PKU CSSCI
英文刊名： JOURNAL OF THE CHINA SOCIETY FOR SCIENTIFIC ANDTECHNICAL INFORMATION
年，卷(期)： 2012,31(1)

参考文献(25条)

1. Luce R D Individual Choice Behavior 1959

2. Qin T;Zhang X D;Tsai M F Query-level loss functions for information retrieval[外文期刊] 2008(02)

3. Liu T Y;Qin T;Xu J LETOR:Benchmark dataset for research on learning to rank for information retrieval 2007

4. Xia F;Liu T Y;Wang J Listwise approach to learning to rank-theory and algorithm 2008

5. Cao Z;Qin T;Liu T Y Learning to rank:From pairwise approach to listwise approach 2007

6. Yue Y;Finley T;Radlinski F A support vector met-hod for optimizing average precision 2007

7. Xu J;Li H AdaRank:A boosting algorithm for information retrieval 2007

8. Burges C J;Shaked T;Renshaw E Learning to rank using gradient descent 2005

9. Donmez P;Svore K M;Burges C J C On the local optimality of LambdaRank 2009

10. Papoulis A Probability,Random Variables and Stochastic Processes 1991

11. Burges C J C;Ragno R;Le Q V Learning to Rank with Nonsmooth Cost Functions 2006

12. Taylor M;Guiver J;Robertson S SoftRank:Optim-ising Non-Smooth Rank Metrics 2008

13. Xu J;Li H AdaRank:a boosting algorithm for inform-ation retrieval 2007

14. Jarvelin K;Kekalainen J Cumulated gain-based evaluati-on of IR techniques 2002

15. Jarvelin K;Kekalainen J IR evaluation methods for retri-eving highly relevant documents 2000

16. Baeza-Yates R;Ribeiro-Neto B Modern Information Retrieval 1999

17. Chen W;Liu T Y;Lan Y Ranking Measures and Loss Functions in Learning to Rank 2009

18. Xia F;Liu T Y;Li H Statistical consistency of Top-K ranking 2009

19. Zhou K;Xue G R;Zha H Y Learning to Rank with Ties 2008

20. Tsai M F;Liu T Y;Qin T FRank:A Ranking Method with Fidelity Loss 2007

21. Freund Y;Iyer R;Schapire R An efficient boosting algorithm for combining preferences 2003

22. Joachims T Optimizing search engines using clickthrough data 2002

23. Li P;Burges C;Wu Q McRank:Learning to rank using multiple classi?cation and gradient boosting 2007

24. Nallapati R Discriminative models for information retrie-val 2004

25. Liu T Y Learning to rank for information retrieval 2009(03)