一种基于节点密度分割和标签传播的 Web 页面挖掘方法

张乃洲[1)] 曹 薇[1)] 李石君[2)]

[1)](河南财经政法大学计算机与信息工程学院 郑州 450002)

[2)](武汉大学计算机学院 武汉 430072)

摘 要 获取 Web 页面中的重要内容如文本和链接,在许多 Web 挖掘研究领域有着重要的应用价值.目前针对该问题主要采用 Web 页面分割和区块识别的方法.但现有的方法将 Web 页面中重要文本和链接的识别视为两个相互独立的问题,这种做法忽略了 Web 页面中文本和链接的内在语义关系,同时降低了页面处理的效率.文中提出了一种 Web 页面重要内容挖掘的统一框架,该框架主要由 3 个部分组成:第一,先将 Web 页面转换为 DOM 树表示,然后采用节点密度熵为度量将 DOM 树分割为不同的页面块;第二,采用基于 K 最近邻标签传播的半监督方法自动扩展页面块训练集;第三,在扩展的页面块训练集上对 SVM 分类器进行训练,并用来对页面块进行分类.采用该框架可以将 Web 页面块区分为多种类型,并且该框架独立于 Web 页面的类型和布局.我们在真实的 Web 环境下进行了广泛的实验,实验结果表明了该方法的有效性.

关键词 页面分割;节点密度;标签传播;DOM 树;块分类;社会计算;社交网络

中图法分类号 TP391 **DOI 号** 10.3724/SP.J.1016.2015.00349

A Method Based on Node Density Segmentation and Label Propagation for Mining Web Page

ZHANG Nai-Zhou[1)] CAO Wei[1)] LI Shi-Jun[2)]

[1)](*College of Computer and Information Engineering, Henan University of Economics and Law, Zhengzhou* 450002)

[2)](*School of Computer, Wuhan University, Wuhan* 430072)

Abstract For many research fields in Web mining, how to get the important content in a Web page, such as texts and links, has important applications. At present, the main method for solving this problem is to adopt Web page segmentation and informative sections recognition. However, existing approaches use decoupled strategies that attempt to do text content and link content identification in two separate phases. This ignores the inner semantic relationships between texts and links in a Web page, and also results in low efficiency of the processing of Web page. In this paper, we propose a uniform framework for mining important content in a Web page. This framework consists of three components. First, a Web page is transformed into a DOM tree, and then it is segmented into several Web page blocks with a metric based on node density entropy. Second, a semi-supervised approach based on K-Nearest Neighbor label propagation is proposed to automatically extend the training set for classification. Third, a SVM-based classifier is trained over the extended training set, and eventually it is leveraged to classify Web page

收稿日期:2013-05-18;最终修改稿收到日期:2014-10-28.本课题得到国家自然科学基金(61272109,61202285)、国家星火计划项目(2012GA750007)、河南省科技厅基础与前沿技术研究项目(122300410378)、河南省教育厅科学技术研究重点项目(13A520032)资助.

张乃洲,男,1970 年生,博士,副教授,中国计算机学会(CCF)会员,主要研究方向为 Web 搜索与挖掘.E-mail: nazze@whu.edu.cn. **曹 薇**,女,1977 年生,硕士,副教授,主要研究方向为机器学习、Web 数据挖掘. **李石君**,男,1964 年生,博士,教授,博士生导师,主要研究领域为数据库技术、Web 数据管理与挖掘.

blocks. The framework can distinguish Web page blocks into a variety of types, and it is independent of the type and layout of Web pages. We conduct the extensive experiment over real Web environment, and the experimental results show that the proposed methods are effective.

Keywords Web page segmentation; node density; label propagation; DOM tree; block classification; social computing; social networks

1 引 言

随着 Web 技术的飞速发展,特别是 Web2.0 的出现,当前 Web 页面包含越来越丰富的内容,以满足商业和终端用户的需求.目前,一个典型的 Web 页面除了主要内容(相关文本和链接)外,还包括导航菜单、广告、用户注释、版权和服务信息等内容.但对于许多 Web 应用来说(如 Web 信息检索、Web 数据挖掘、移动互联网等),这可能会成为一种障碍.例如:(1)Web 数据挖掘任务(如 Web 信息抽取),需要分析 Web 页面中的主要文本内容以完成有用知识的发现.但当前 Web 页面中包含的大量"噪音"内容会对数据挖掘任务造成不小的干扰;(2)搜索引擎爬虫在抓取给定页面中的链接时,会遇到许多"无用"链接(如页面中的导航菜单、广告链接、服务链接等),如果不能有效去除这些链接,将会大大降低爬虫的爬行效率,甚至造成爬行主题的漂移;(3)为满足 Web 信息检索的需要,被抓取的 Web 页面往往需要被缓存(Cached)和存储在本地数据库中,而 Web 页面中大量冗余内容的存在会造成系统存储空间和网络带宽的严重浪费;(4)在移动互联网方面,由于手机或 PDA 设备具有屏幕尺寸较小和内存容量有限等特点,使得包含大量冗余信息的 Web 页面不适合在这些移动终端中直接传送和显示,而必须对页面中的主要内容进行抽取,然后进行页面重构[1-3].

基于以上这些问题,使得 Web 页面重要内容挖掘成为当前学术界的一个研究热点[1-15].根据应用目的不同,该问题又可以分为两个子任务:一是 Web 页面主要文本内容(Main Content)的挖掘;二是 Web 页面主要链接(Main Links)的挖掘.但当前很多研究是将这两个子任务视为相互独立的任务对待,这种分解的做法存在两个方面的主要问题.其一,降低了页面处理的效率.实际上很多应用同时包含了这两个子任务,如 Web 信息检索、移动互联网应用等.其二,忽略了 Web 页面中主要文本和主要链接的内在语义关系.例如,通过直觉观察可以得到这样的结论:在一个 Web 页面中,主要文本块中包含的链接以及邻近该主要文本块的链接,与该主要文本块具有较强的主题相关性.这一规则对 Web 聚焦爬行来说,具有重要的意义,可被用于指导聚焦爬虫进行主题爬行.因为对于链接来说,其锚链文本越短,包含的语义信息越少;主要文本块往往具有丰富的语义信息,更容易度量其主题相关性.而上述分解的做法却无法利用这种内在的语义关系.本文提出的方法,能够在一次页面处理过程中同时完成这两个任务,并能够在此基础上充分利用 Web 页面块内在的结构和语义关系,进行深层次的数据挖掘工作.

由于 Web 页面结构存在很强的异构性,即相同的页面内容可以有不同的页面布局,而相同页面布局又可以使用不同的实现方式,并且不同的 Web 站点风格也千差万别.因此,当前 Web 页面重要内容挖掘研究还存在不少问题和挑战.该研究的关键在于:(1)如何有效分割(Segment)Web 页面;(2)如何提取页面分割块的特征;(3)如何利用这些特征来区分不同的分割块;(4)如何提高上述几个步骤的精度和效率.

而现存研究普遍存在的问题主要有:提出的页面分割算法要么采用的方法较简单,很难适应复杂的页面[4],要么比较复杂,使得算法复杂度过高[5];提出的方法不能同时解决 Web 页面内主要文本块和链接块的识别问题等.

本文提出了一个新的 Web 页面重要内容挖掘的统一框架,用于发现给定 Web 页面中重要的文本和链接.本文的主要贡献如下:

(1)提出了一个高效的基于 DOM 树的页面分割算法.该算法首先将给定的 Web 页面转换为 DOM 树表示,然后采用节点密度熵为度量将 DOM 树自动分割为若干页面块,并采用启发式规则将相似页面块融合为较大的块.实验表明该算法独立于页面布局,具有很好的泛化能力和较好的页面分割精度.

(2)提出了一个有效的基于半监督学习的页面块分类算法框架. 该框架首先确定页面块分类特征集 F,然后在较小的已标记训练集上,采用基于 K 最近邻标签传播(KNNLP)算法自动生成较大的已标记训练集,并利用该训练集训练一个基于 SVM 的块分类器. 最后采用该分类器来识别 Web 页面中重要的文本和链接块. 实验表明本文提出的 KNNLP 算法在精度上要优于现有基于图的标签传播算法,并且整个算法框架具有较好的性能.

(3)提出的方法能够在一次页面处理过程中同时完成 Web 页面中主要文本内容和主要链接的发现,从而显著提高了页面处理的效率. 此外,该框架能够在页面分割的基础上充分利用 Web 页面块内在结构和语义关系,进行深层次的数据挖掘,并与其他 Web 应用有效结合.

本文第 2 节简要介绍相关的研究工作;第 3 节给出问题的定义和形式化描述;第 4 节介绍所提出框架各部分的主要设计思想和算法;第 5 节是实验和分析部分;最后一节给出了结语以及对未来工作的展望.

2　相关工作

2.1　Web 页面挖掘

从研究目标的角度来看,当前 Web 页面重要内容挖掘研究主要集中在 Web 页面的内容抽取(Content Extraction)[4]. 该任务的主要目标是发现和识别 Web 页面中的主要内容. 这一问题的一个重要研究方向是 Web 信息抽取(Web Information Extraction),该研究的主要任务是采用基于规则或机器学习的方法从海量的 Web 数据中发现和获取指定主题或格式的半结构化或非结构化信息,并最终将其转换为结构化信息. 但该研究不属于本文研究的范围. 本文研究的内容抽取问题可以视为一种"浅层"(Shallow)的信息抽取,可作为 WIE 问题的预处理步骤. 而从所采用技术的角度来看,内容抽取问题可以分为两类:

第一类是采用抽样的方法从同一 Web 站点获得一定数量的 Web 页面,然后发现其中的重复模式,或称为模板(Template). 如 Yi 等人[6]提出了一种称为站点风格树(Site Style Tree)的结构来捕获给定 Web 站点中公共的表示风格和页面的实际内容,然后使用一种基于信息的度量方法来确定站点中的噪音和主要内容. Ramaswamy 等人[7]利用同一 Web 站点内的页面具有内容共享性的特点,提出了一种 Shingling 算法来识别 Web 页面中的片段. 利用该方法可以去除多个页面中的共享内容,从而能显著减少对 Web 缓存的需求. 而 Debnath 等人[8]也利用了同一 Web 站点的页面在内容和结构上具有相似性的特点,定义了四种抽取器来识别页面中的主要内容块. 针对许多实际的模板检测任务是在没有任何有关站点结构先验知识的情况,Kolcz 等人[9]提出一种使用站点内部模板发现技术来驱动站点独立的模板检测器的归纳过程,该方法减少了对人工标注数据的要求. 其他类似的文献有[10-11]等. 该类方法的缺点是需要通过获取和比较多个 Web 页面才能得到有关站点的结构信息,这将导致在无法获取足够页面的情况下,该方法的失效. 此外,这类方法还会产生额外的系统存储耗费和处理延时[11].

另一类是在单一的 Web 页面中发现重要的区域. 如 Yin 等人[1]提出了一种类似于 PageRank 的排序算法来排序一个 Web 页面中的内容块,使得在一个 Web 页面中,仅仅重要的部分才会被抽取,然后将其传送到移动设备中. 而 Baluja[3]提出了一个关于 Web 页面分割的机器学习的框架,通过采用一个有效的页面分割算法,利用透镜信息熵的减少和决策树学习来实现 Web 页面在移动设备屏幕上不失真地缩放.

以上两类方法大多使用了页面分割的思想,特别是后者. 当前已有不少方法被用来将 Web 页面分割成块(Block)或区域(Region). 如 Cai 等人[5]提出了一个基于视觉的页面分割(VIPS)算法. 该算法利用了 Web 页面中的视觉线索如页面元素的高度、宽度、背景色、位置等,来获取页面的最优划分. 该算法的优点是块的划分粒度适中,块的语义内聚度好. 但缺点是使用了较多的启发式规则,算法复杂度较高,系统的运行效率存在问题. Chakrabarti 等人[12]提出了一个基于图切割的 Web 页面分割方法. 该算法将 Web 页面分割问题形式化为一个权重图上最优化问题,并提出了一个在手工标记的数据集上进行学习的框架,用来学习这些权重. 而在进行图切割时根据节点间的权重大小来决定节点是否需要分割. Bing 等人[13]提出了一个基于 Web 页面分割图模型的结构化预测方法. 该方法将页面分割问题形式化为一个结构化标记问题,而 Web 页面分割图建模了一个 Web 页面的候选分割区边界以及相邻页面分

割区之间的依赖关系. Web 页面分割图上的每一个标记模式对应于一个可能的 Web 页面分割,而算法的核心是采用整数线性规划方法来发现最优的 Web 页面分割图标记. 近来一些研究提出了使用 DOM 树节点的密度特征来进行页面的分割[4,14-15]. 如 Sun 等人[4]提出了一个使用文本密度来进行内容抽取(CETD)的方法. 该方法首先将 Web 页面转化为 DOM 树,然后给出了一个复合文本密度公式(CTD),并利用该公式计算每个 DOM 树节点的 CTD 值,以决定该节点的内容是否为待抽取的文本内容.

本文的工作属于第二类方法,并吸收了文献[4,14-15]等的思想. 但与这些文献不同的是本文使用了多个节点密度的度量,并通过计算节点密度熵来划分 Web 页面,而页面块的识别是使用 SVM 分类器来完成的. 这样做的目的是为了适应多种不同的应用(文本块、链接块或其他类型块的识别和抽取),同时又能提高页面块识别的精度和系统在线处理的效率.

2.2 标签传播算法

将 Web 页面分割成页面块后,即可使用现有的分类算法来进行页面块的识别. 当前的分类算法主要分为 3 类:(1)有监督积极学习(Eager Leaner)方法. 如决策树、人工神经网络(ANN)、支持向量机(SVM)等. 该类算法的特点是在给定标记训练集上学习到一个最优模型,然后通过模型来预测未知数据. 其优点是分类精度和运行效率高,但缺点是分类精度依赖训练集的规模,且训练数据一般需要手工标注、人工耗费大、训练过程较慢等;(2)有监督懒散学习(Lazy Leaner)方法. 典型的是 K 最近邻(KNN)分类. 该类算法的优点是不需要训练模型,在新数据点加入时直接分类. 缺点是需要记住所有的训练样本,分类开销大,效率较低;(3)半监督学习方法. 如标签传播算法[16-18]、半监督支持向量机[19]等. 该类算法的优点是利用了未标记数据参与分类,需要的人工耗费较小. 但缺点是随着未标记数据比例的增加,分类精度下降,且运行效率低于有监督方法.

为保证页面块分类器的在线运行效率,本文主要采用 SVM 模型来进行页面块的分类. 但本文 Web 页面重要内容挖掘研究的一个显著特点是:一个页面会产生多个块(几个到数十个不等),因此需要的训练数据较大. 如果完全手工训练数据,则会产生很大的人工耗费. 因此本文结合了有监督和半监督方法来完成页面块的分类问题. 主要思想是:先采用半监督的标签传播算法来扩展初始的标记训练集,当训练集的大小达到指定值时,再使用 SVM 模型在扩展的训练集上进行模型训练和分类.

标签传播(Label Propagation)算法最早由文献[16]提出,其主要动机是在现实的数据挖掘应用中(如分类),未标记数据丰富而容易获得,而标记数据稀少且需要大的人工耗费进行标记. 由此提出的问题是能否利用未标记数据进行分类,从而减少人工标记的代价?而其主要思想是利用少量标记数据和大量未标记数据内在的流形结构来辅助分类. 其底层的基本假设[18]为:(1)空间中的近邻点很可能具有相同的标签;(2)在相同结构上的点很可能具有相同的标签.

本文提出的 KNNLP 算法与文献[16,18]提出方法的主要区别如下:

(1)文献[16]采用的图模型为完全连接图,即在进行标签传播过程中,对任意一个给定的数据点 $\boldsymbol{x}$,其他所有点的标签均会按照其与 $\boldsymbol{x}$ 相似度的大小对其施加影响. 其缺点是增加了算法的时间和空间复杂度. 同时,在实验中我们发现,这会降低算法的鲁棒性. 在某些情形下,如当分类数据集中存在桥接点(Bridge Point)[18]时,算法的性能会严重下降.

(2)文献[18]提出了 LNP(Linear Neighborhood Propagation)算法. 其核心思想是借用了流形学习中经典的 LLE(Local Linear Embedding)算法的思想,即对任意一个给定的数据点 $\boldsymbol{x}$,可用其 k 个最近邻的线性组合来进行重构,而学习到的最优系数向量则做为 $\boldsymbol{x}$ 与其最近邻相似度大小的度量. 与文献[16]相比,其优点在于:在进行标签传播过程中,对任意一个给定的数据点 $\boldsymbol{x}$,仅使用了 k 个最近邻点对其施加影响,这大大降低了算法的时间和空间复杂度,同时也提高了算法的鲁棒性. 但与 LLE 算法相比,由于 LNP 算法要求重构系数非负,由此可能导致其学习到的某些数据点对应的最优系数向量存在较大的重构误差. 在这种情况下,表明该系数向量并未真正反映数据点 $\boldsymbol{x}$ 与其最近邻相似度的大小关系,这会降低算法的性能,特别是当存在桥接点时. 我们的相关实验也证明了这一点. 因此,为了保证算法的性能,文献[18]采取了预处理步骤,去除了数据集中的桥接点和孤立点.

(3)本文提出的 KNNLP 算法考虑了以上两种

算法存在的问题. 其一，与文献[18]类似，在进行标签传播过程中，对任意一个给定的数据点 $\boldsymbol{x}$，仅使用其 k 个最近邻点对其施加影响. 其二，与文献[16]类似，数据点 $\boldsymbol{x}$ 与其最近邻相似度大小的度量采用了高斯函数来计算. 但与文献[16]不同的是，提出了相似度度量中平衡因子的定量计算方法. 因此，从算法效率来看，KNNLP 算法要优于文献[16]，同时也优于文献[18]. 因为 LNP 算法在学习最优系数向量的过程中，使用二次规划是比较耗时的. 而从鲁棒性上来看，我们进行了不同数据集下的相关实验，包括存在桥接点的情况，实验表明 KNNLP 算法要优于以上两种方法. 因此，KNNLP 算法并不需要对数据集进行额外的预处理.

3　问题描述及相关定义

为说明本文提出的算法框架，本节给出 Web 页面重要内容挖掘问题描述及相关概念的定义.

定义 1.　每一个 Web 页面均可以表示成一个 DOM 树 T_d. T_d 是一个有向图 $\langle V,E\rangle$，其中 V 为顶点的集合，$V=\{v\mid v\in html$ 标签集 $Tag\}$. E 为有向边的集合，$E=\{\langle u,v\rangle\mid u,v\in V$，其中 u 称为 v 的父顶点，而 v 称为 u 的子顶点，且在 $html$ 结构上，v 对应的标签被 u 对应的标签所包含$\}$. T_d 中离开顶点 v 的边的个数称为 v 的出度；指向顶点 v 的边的个数称为 v 的入度. T_d 中入度为零的顶点称为 T_d 的根；而出度为零的顶点称为 T_d 的叶节点.

定义 2.　一个 DOM 树 T_d 可表示为一个页面块的集合 $B=\{b_i\mid b_i\in V$，且 b_i 为叶节点$\}$.

定义 3.　页面块集合 B 中的任意元素 b_i 可表示为一个 k 维向量 $\boldsymbol{x}_i=\langle f_1,f_2,\cdots,f_k\rangle$，其中 f_j 代表 b_i 的第 j 个特征.

定义 4.　给定 Web 页面集合 P，其上的训练集 $\widetilde{D}$ 可表示为 $\{\langle \boldsymbol{x}_1,y_1\rangle,\langle \boldsymbol{x}_2,y_2\rangle,\cdots,\langle \boldsymbol{x}_n,y_n\rangle\}$，其中 $y_i\in Y=\{1,2,\cdots,C\}$，$Y$ 为类标签集合.

由以上给出的 4 个基本定义可以得到本文关于 Web 页面重要内容挖掘问题的形式化描述，该问题可分为如下 3 个部分：

(1) Web 页面分割. 该过程可定义为映射 S_p：$P\rightarrow B$，S_p 将给定 Web 页面集合 P 中的每个页面 p 映射为块的集合.

(2) 基于半监督学习的训练集扩展. 给定已标记训练集 $\widetilde{D}_L=\{\langle \boldsymbol{x}_1,y_1\rangle,\langle \boldsymbol{x}_2,y_2\rangle,\cdots,\langle \boldsymbol{x}_l,y_l\rangle\}$，$y_i\in Y_L=\{1,2,\cdots,n\}$，$1\leq i\leq l$. 这里 Y_L 为已知的类标签集合. 给定未标记训练集 $\widetilde{D}_U=\{\langle \boldsymbol{x}_{l+1},y_{l+1}\rangle,\langle \boldsymbol{x}_{l+2},y_{l+2}\rangle,\cdots,\langle \boldsymbol{x}_{l+u},y_{l+u}\rangle\}$，其中，$y_j\in Y_U$，$l+1\leq j\leq l+u$. Y_U 为未知标签集合，且 $l\ll u$. 该训练集扩展过程是在 $\widetilde{D}_L$ 和 $\widetilde{D}_U$ 上，产生一个映射 ℓ：$Y_U\rightarrow Y_L$，最后合并 $\widetilde{D}_L$ 和 $\widetilde{D}_U$ 为 $\widetilde{D}$.

(3) 页面块分类. 使用 $\widetilde{D}$ 训练一个分类函数 β. β 可定义为 $\beta(\widetilde{D})$：$B\rightarrow Y_L$，其作用是将任意的页面块 b_i 映射为一个已知的标签 y_i. 给定一个 Web 页面 p，使用 S_p 得到相应的页面块集 B，然后使用块分类函数 β 对 B 进行分类.

为了说明本文提出的 Web 页面重要内容挖掘方法，图 1 给出了一个例子页面(http://tech.sina.com.cn/t/2012-04-20/05006988760.shtml)①. 该页面中虚线矩形框表示页面分块，其中标记为①和②的块是该页面的重要内容，分别对应于相关文本块和链接块. 本文提出的 Web 页面重要内容挖掘方法的目标是自动发现和标记出给定页面中的这些重要区域.

图 1　一个 Web 页面及分割的例子

4　所提出的方法

上节给出了本文提出 Web 页面重要内容挖掘

① 由于该页面较长，基于篇幅的原因，部分内容被删去.

方法的形式化描述，本节详细介绍该框架 3 个组成部分的原理及算法描述.

4.1 基于节点密度熵的自适应 Web 页面分割方法

当前对 Web 页面分割问题已进行了深入的研究. 文献[15]对各种分割方法从视觉问题、语言学问题、密度问题和一维问题等四个不同角度分别进行了详细的讨论. 本文在设计 Web 页面分割算法时，主要考虑了如下因素：(1) 应充分利用 HTML 文档自身的结构；(2) 选择能反映待挖掘内容的本质特征；(3) 简单高效. 为此，结合现存研究本文采用了基于密度度量的 DOM 树自适应分割算法. 该算法首先需要将给定的 Web 页面转换为 DOM 树表示. 在实际的系统实现中，我们采用了开源项目 Nekohtml① 来完成 DOM 树的转换. Nekohtml 是一个强大的 HTML 扫描器和标签平衡器. 与其他解析器相比，Nekohtml 对非良结构的 HTML 文档具有很强的鲁棒性. 我们的实验表明它能够正确解析大部分的 HTML 文档，这为我们的研究工作提供了很大的便利. 下面先给出 3 个密度定义：

定义 5. 设 n_i 为 DOM 树 T_d 中的一个节点(顶点)，则 n_i 的文本密度定义为

$$\rho_{text}^{i}=\frac{N_{n_i}}{N_T} \tag{1}$$

其中，N_{n_i} 为节点 n_i 代表的子树中，去除所有 *html* 标签(包括链接文本)后的纯文本数(字符数)；N_T 为 T_d 代表的整个文档中去除所有 *html* 标签(包括链接文本)后的纯文本数(字符数).

ρ_{text}^{i} 反映了在全局页面中，文本内容在某个局部区域的相对集中度. 通过观察和实验，我们发现 ρ_{text}^{i} 越大，往往意味着该节点越有可能包含待发现的主要文本块. 从图 1 中的标记块①可以直觉地看到这一点.

定义 6. 设 n_i 为 DOM 树 T_d 中的一个节点，则 n_i 的链接密度定义为

$$\rho_{link}^{i}=\frac{lN_{n_i}}{lN_T} \tag{2}$$

其中，lN_{n_i} 为节点 n_i 代表的子树中，所包含的链接数(〈*a*〉标签数)；lN_T 为 T_d 代表的整个文档中所包含的链接数.

ρ_{link}^{i} 反映了在全局页面中，链接在某个局部区域的相对集中度. 一般而言，ρ_{link}^{i} 越大则表示该节点越有可能包含待发现的主要链接块(如图 1 中的标记块②). 引入 ρ_{link}^{i} 指标的目的在于发现给定页面中的重要链接块.

定义 7. 设 n_i 为 DOM 树 T_d 中的一个节点，则 n_i 的链接文本密度定义为

$$\rho_{ltext}^{i}=\frac{lT_{n_i}}{lT_T} \tag{3}$$

其中，lT_{n_i} 为节点 n_i 代表的子树中，所有链接所包含的文本数(包含在〈*a*〉〈/*a*〉标签之间的文本数)；lT_T 为 T_d 代表的整个文档中所包含的链接文本数.

ρ_{ltext}^{i} 反映了链接文本在页面某个局部区域的相对集中度. 引入该指标主要考虑到这样一种页面分割场景：某些节点包含较少的链接，但所包含的链接文本数却较多(具有较长的平均链接长度)，而这些链接往往具有相关链接的特征(如图 1 中的标记块②).

给出了 3 个密度度量后，可以引入如下节点密度熵②的定义：

$$H(n_i)=-\sum_{i}^{n}p_i\log_n p_i \tag{4}$$

本文中取 $n=3$，$p_1={\rho_{text}^{i}}^{1/2}$，$p_2={\rho_{link}^{i}}^{1/2}$，$p_3={\rho_{ltext}^{i}}^{1/2}$.

利用节点密度熵 $H(n_i)$，可以得到如下页面分割算法.

算法 1. segmentPage.

输入：DOM 树根节点 *root*

输出：页面融合块集合 B_f

```
Begin
1. B=segmentDomTree(root) //B 为页面分割块集合
2. B_f=fuseBlocks(B)
End
函数:segmentDomTree(Nd)
输入：DOM 树节点 Nd；
输出：页面分割块集合 B；
Begin
1. B←∅
2. if Nd has child nodes then
3.   nds[n]←getChildNodes(Nd)
4.   for each node k∈nds [n] do
5.     subnd←nds [k]
6.     if subnd is ELEMENT_NODE then
7.       H(subnd)←calNodeDensityEntropy(subnd)
8.       if H(subnd)>α then
9.         segmentDomTree(subnd)
10.      else if H(subnd)>β then
11.        B←subnd
```

① http://nekohtml.sourceforge.net/

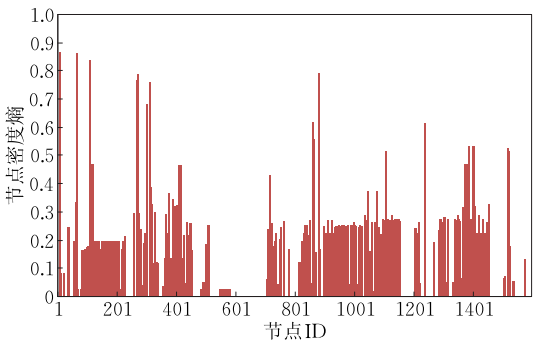
② 此处借用了熵的概念，节点密度熵与严格意义上的熵有一定区别.

```
12.     else if subnd is TEXT_NODE then
13.       H(subnd) ← calNodeDensityEntropy(subnd)
14.       if H(subnd) > γ then
15.         B ← subnd
End
函数：fuseBlocks(B)
输入：页面分割块集合 B
输出：页面融合块集合 B_f
Begin
1. B_f ← ∅
2. while B ≠ ∅ do
3.    ∀b_i ∈ B, remove b_i from B
4.    for b_j ∈ B do
        //规则集 R：①两节点有共同的父节点；②两节
          点拥有共同或相似的标签；③两节点相邻
5.      if b_i and b_j satisfy the rule set R then
6.        b_i ← b_i ∪ b_j
7.        remove b_j from B
8.    B_f ← b_i
End
```

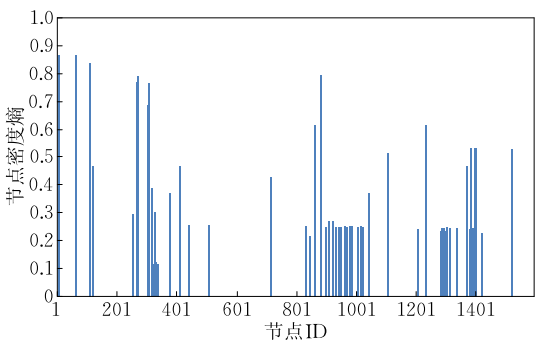
算法 1 由两个部分组成：函数 *segmentDomTree* 和函数 *fuseBlocks*. 首先执行第 1 步：调用函数 *segmentDomTree*. 该函数是一个递归函数，主要完成页面的初次分割，并产生页面分割块集合 B. 其中，*subnd* 表示当前子节点. *ELEMENT_NODE* 和 *TEXT_NODE* 为节点的类型，分别代表元素节点和文本节点. $H(subnd)$表示节点 *subnd* 的节点密度熵，而函数 *calNodeDensityEntropy* 用于计算给定节点的节点密度熵.

函数 *segmentDomTree* 初始的输入是整个 DOM 树 T_d，该函数只处理 DOM 树中的元素节点和文本节点，其中 α、β 和 γ 是分割控制参数，其值由实验来确定. 从算法描述中可以看出，当节点 *subnd* 为元素节点时，α 决定了 *subnd* 节点是否被继续分割. 因此，α 决定了块分割的大小. α 越小，页面块分割得越小. 如果 α 取得太小，可能使得本来在语义上属于同一个块的内容被分割开，同时增加了算法的时间复杂度；而 α 取得太大，又可能由于页面块分割得太大使得最后得到的页面块中的信息区域没有进行有效区分，从而降低了分割召回率. 而当节点 *subnd* 不再被分割时，β 决定了 *subnd*（非文本节点块）是否被放置到集合 B 中. 因此，β 决定了非文本节点块尺寸的最小值. 若该值取得太大，则可能漏掉一些分割块；取得太小，则可能会保留一些没有实际意义的块. 当节点 *subnd* 为文本节点时，*subnd*（文本节点块）是否被放置到集合 B 中由 γ 决定. γ 的作用类似 β，不同之处是 γ 决定了文本节点块尺寸的最小值. 在后续的实验部分我们将讨论它们的取值问题.

图 2 给出了采用函数 *segmentDomTree* 来分割例子页面（图 1 所示）所得到的节点密度熵分布图，其中(a)部分是分割前所有的节点（只包括元素节点和文本节点）对应的节点密度熵 $H(n_i)$；而(b)部分是分割过程中所有被处理的中间节点（被分割节点）和叶节点（块），它们分别对应于图 3 中的黑色节点和白色节点. 从图 2 可以看到，在分割过程中大部分的节点被过滤掉了（未被处理）. 原 Web 页面共有 1594 个节点，而实际处理的只有 63 个节点，约占 4%. 图 2 直观地说明了使用节点密度熵为度量来分割 DOM 树的高效.



(a) 分割前所有节点对应的节点密度熵



(b) 被处理的中间节点和叶节点

图 2　例子 Web 页面节点密度熵分布图

调用函数 *segmentDomTree* 对 DOM 树进行分割后还需要进行的一个重要处理是块融合问题. 函数 *segmentDomTree* 仅仅考虑了节点的密度属性，而未考虑到节点之间的语义关系. 例如，若 DOM 树中存在两个互为兄弟节点的叶节点，其标签均为〈*P*〉，那么这意味着它们对应于页面中某一文本区中的段落，则在语义上它们应该属于同一个块. 如果页面分割过程中，这两个节点被分割为两个块，则必须将它们融合为一个块. 为此算法 1 执行第 2 步：调

用函数 *fuseBlocks*. 该函数主要用于扫描页面分割块集合 B,并完成对分割块的融合操作. 需要指出的是,函数 *fuseBlocks* 中任意两个块节点必须同时满足规则集 R 中的 3 条规则,才能进行块融合操作. 其中相似标签是指标签具有相似的功能,例如格式标签〈*em*〉、〈*strong*〉等都是对文本格式进行修饰,因此这些元素类型的节点等价于文本类型的节点. 对相邻关系本文采用如下方法判别:对页面分割块集 B 按 ID 号对块排序,如果两个块在 B 中相邻,则它们为相邻关系.

图 3 显示了调用算法 1 对例子页面进行 DOM 树分割后的结果. 其中白色节点为叶节点(执行算法 1 后得到的块节点),图中虚线矩形框表示调用算法 1 中的 *fuseBlocks* 函数进行块融合后,得到的更大页面块. 从图上可以看到最后整个页面被划分为 11 个不同的区域(块),其中包括 8 个融合块和 3 个未融合块.

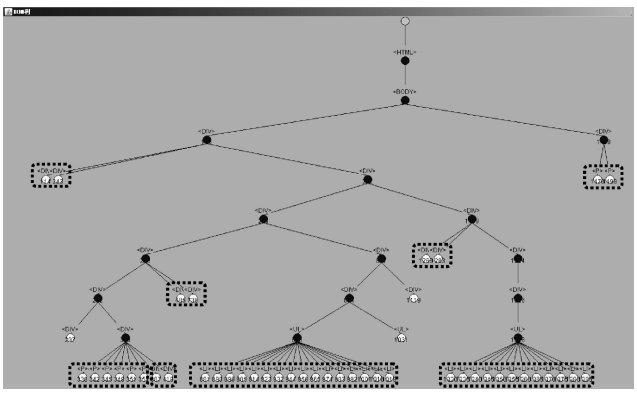


图 3 对例子页面进行 DOM 树分割及页面块融合的结果

4.2 页面块特征选择

为了对页面块进行有效分类,本文将任意的块 b_i 建模为一个 6 元组:

$$\langle \rho^i_{text}, \rho^i_{link}, \rho^i_{ltext}, width, height, sp \rangle \tag{5}$$

式中,ρ^i_{text}、ρ^i_{link} 和 ρ^i_{ltext} 分别为块 b_i 的文本密度、链接密度和链接文本密度,它们的定义参见式(1)~(2). 而式(5)中,后 3 个特征为块的视觉特征:*width* 代表块的宽度,*height* 表示块的高度,而 *sp* 表示块在页面中的位置所对应的重要度分值.

页面块的视觉特征一般包括:块的大小、块内字体大小和颜色、块的背景色、块在页面中的位置等. 之所以使用视觉特征是源于这样的观察:Web 页面的设计者在进行页面布局设计时是遵循一定原则和惯例的. 例如,页面布局大体上是按头部、内容区和尾部划分;一般重要的内容放在页面的中部等. 因此通过块的视觉信息能够在一定程度上区分块的不同. 如导航菜单在视觉上一般具备狭长的特征,其位置往往位于页面的头部;而页面的主要文本内容一般具有较大的 *width* 和 *height*,且往往位于页面的中部等.

Web 页面块的视觉信息可以通过抽取并分析页面的 CSS(Cascading Style Sheet)来获取,为此在实际的系统中我们设计了 CSS 抽取和分析器,以获取每个 DOM 树节点和页面块的视觉属性.

在 *width* 和 *height* 取值上,我们以屏幕的宽度和高度为基准,将 *width* 和 *height* 映射到(0,1]. 而 *sp* 的取值复杂一些,本文采用了如表 1 的策略.

表 1 *sp* 的取值

		X		
		左	中	右
Y	上	0.2	0.2	0.2
	中	0.6	0.8	0.5
	下	0.1	0.1	0.1

表 1 中的 X 表示块在页面中水平方向的位置,其值可以取:"左"、"中"、"右",该值可以通过 CSS 来获取. 而 Y 表示块在页面中垂直方向的位置,其值可以取:"上"、"中"、"下". 本文通过块内节点 ID 号来近似计算该值. 其依据是:本文采用的 DOM 树遍历算法(前序遍历)产生的节点 ID 号与浏览器对 HTML 页面标签渲染的顺序大致相同. 从图 3 可以直观地看到这一点. 这样,具有较小 ID 号的节点一般位于页面的头部,而 ID 号较大的节点一般位于页面的尾部. 表 1 单元格中的数值为 *sp* 的分值. 从表 1 可知,一个 Web 页面大致被分成了头(上)、左、中、右、尾(下)5 个区域.

4.3 基于 *K* 最近邻标签传播增量训练集扩展算法

本小节主要描述本文提出的基于 K 最近邻标签传播(KNNLP)的增量训练集扩展算法.

给定已标记训练集 $\widetilde{D}_L = \{\langle \boldsymbol{x}_1, y_1 \rangle, \langle \boldsymbol{x}_2, y_2 \rangle, \cdots, \langle \boldsymbol{x}_l, y_l \rangle\}$, $y_i \in Y_L = \{1, 2, \cdots, n\}$, $1 \le i \le l$. 这里 Y_L 为已知的类标签集合. 给定未标记训练集 $\widetilde{D}_U = \{\langle \boldsymbol{x}_{l+1}, y_{l+1} \rangle, \langle \boldsymbol{x}_{l+2}, y_{l+2} \rangle, \cdots, \langle \boldsymbol{x}_{l+u}, y_{l+u} \rangle\}$, $y_j \in Y_U$, $l+1 \le j \le l+u$. 其中 Y_U 为未知(未观察到)标签集合,且 $l \ll u$. 令 $\widetilde{D} = \widetilde{D}_L \cup \widetilde{D}_U$,则标签传播问题定义为:如何从 $\widetilde{D}$ 和 Y_L 中估计 Y_U.

给定一个近邻连接图 $G = \langle V, E \rangle$,其中 V 为顶点的集合,$V = \{v | v \in \widetilde{D}\}$. E 为边的集合,$E = \{\langle u, v \rangle | u, v \in \widetilde{D}$,且 $v \in N_k(u)$,$N_k(u)$ 为顶点 u 的 k 个最近邻

集合}. 则 G 中任意一条边$\langle i,j\rangle$的权重 w_{ij} 定义为

$$w_{ij}=\exp\left(-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2}{\delta^2}\right) \tag{6}$$

其中，δ 为平衡因子，本文利用一个 $\widetilde{D}$ 上所有数据的方差函数来计算：

$$\delta=\min\left(v,\ \frac{\sqrt{v}}{\ln^2 v}\right) \tag{7}$$

其中，v 为 $\widetilde{D}$ 上所有数据的方差.

定义如下$(l+u)\times k$ 的近邻概率转移矩阵 $\boldsymbol{T}$：

$$\boldsymbol{T}_{ij}=\frac{w_{ij}}{\sum_{m=1}^{k} w_{im}} \tag{8}$$

其中，$\boldsymbol{T}$ 反映了数据集中各个数据点与其 k 个最近邻数据点之间相似度的大小.

定义 $\boldsymbol{L}$ 为$(l+u)\times|Y_{\boldsymbol{L}}|$的标签矩阵，其行代表数据点，列代表类别，且 $\boldsymbol{L}_{ij}$ 定义为

$$\boldsymbol{L}_{ij}=\begin{cases}1, & y_i=j\\ 0, & \text{其他}\end{cases} \tag{9}$$

定义 $\boldsymbol{H}^m$ 为数据点 m 的 $k\times|Y_{\boldsymbol{L}}|$近邻标签矩阵，其行表示数据点 m 的近邻，列表示类别，则 $\boldsymbol{H}_{ij}^m$ 定义为

$$\boldsymbol{H}_{ij}^m=\boldsymbol{L}_{\mathrm{ij}},\ \ \mathrm{i}\in \mathrm{N_k(m)} \tag{10}$$

其中 $\mathrm{N_k(m)}$表示数据点 m 的 k 个最近邻集合.

设 $\boldsymbol{T}_i$表示近邻概率转移矩阵 $\boldsymbol{T}$ 的第 i 个行向量，$\boldsymbol{L}_i$表示标签矩阵 $\boldsymbol{L}$ 的第 i 个行向量，$\boldsymbol{L}^{(0)}$ 为初始的标签矩阵，则 KNNLP 算法的主要步骤为

1. 对于所有数据点 i，计算 $\boldsymbol{T}_i\boldsymbol{H}^i$，并将 $\boldsymbol{L}_i\leftarrow\boldsymbol{T}_i\boldsymbol{H}^i$.

2. 对 $\boldsymbol{L}_i$进行行规范化，并重新恢复 $\boldsymbol{L}_i$ 中已标记数据的初始值：$\boldsymbol{L}_i=\alpha\boldsymbol{L}_i+(1-\alpha)\boldsymbol{L}_i^{(0)}$，$\alpha\in(0,1)$.

3. 重复步 1 和 2，直到 $\boldsymbol{L}$ 收敛.

4. 根据 $\boldsymbol{L}$ 获取 $\widetilde{D}_U$ 中任意数据 $\boldsymbol{x}_i(l+1\leq i\leq l+u)$的标签 y_i. 其中 $y_i=\arg\max_j \boldsymbol{L}_{ij}$.

关于 $\boldsymbol{L}$ 收敛性的证明，可以参看文献[16-18]. 此外，KNNLP 算法中最优近邻数 k 的取值并无确定的方法，与数据点本身的维度大小和数据集本身的特征有关，通常需要通过实验来确定. 根据我们的实验，一般当 $k\geq4$ 时，才能取得较理想的结果. 因此在本文后续实验中，我们取 $k=7$.

利用 KNNLP 算法，可以给出如下的增量训练集扩展算法.

算法 2.　extendTrainSet.

输入：初始的训练页面集合 P_l 和 P_u；$Y_{\boldsymbol{L}}=\{1,2,3\}$，分别代表：相关文本块、相关链接块、其他类别块

输出：扩展的页面分割块训练集 TS

Begin

1. 预处理. 对初始的训练页面集合 P_l 采用算法 1 进行页面分割，并获取页面块集 B. $\forall b_i\in B$，采用式(5)将其表示为特征向量 $\boldsymbol{x}_i$，并使用手工方式将其标记为$(\boldsymbol{x}_i,y_i)$，其中 $y_i\in Y_{\boldsymbol{L}}$，最后得到 TS_l. 对训练页面集合 P_u 采用相同的方法进行页面分割和块特征向量表示，但不对块进行类别标识(即设置 $y_i=0$)，最后得到 TS_u.

2. 令 $TS\leftarrow\varnothing$.

3. 任取 TS_u 中 k 个元素构成 TS_u^k，并将这 k 个元素从 TS_u 删除. 在 TS_l 和 TS_u^k 上采用式(6)～(8)构造近邻概率转移矩阵 $\boldsymbol{T}$，并对 $\boldsymbol{T}$ 做行规范化. 利用式(9)构造标签矩阵 $\boldsymbol{L}$.

4. 使用 KNNLP 算法，得到被自动标记的训练集 TS_l^k(即获得标记的 TS_u^k)，并令 $TS\leftarrow TS_l^k$.

5. 回到步骤 3，继续进行训练集的扩展，直到 TS_u 为空.

6. $TS\leftarrow TS\cup TS_l$.

7. 返回扩展的训练集 TS.

End

使用算法 2 进行标签扩展时，每次需要添加 k 个未标记数据到训练集中. 为保证标签传播的精度，必须合理选择 k 的取值. 假设初始已标记训练集大小为 m，实验表明，当取 $k\approx0.56m$ 时，效果较好. 具体情况将在实验部分讨论.

4.4　基于 SVM 的页面块分类算法

使用算法 2 获取足够数量的已标记页面块训练集之后，可以利用现有的分类算法对待识别页面块进行分类. 本文采用基于支持向量机(SVM)的页面块分类器，页面块被分为 3 个类别：相关文本块、相关链接块、其他类别. 基于 SVM 的页面块分类器的工作过程如下：

(1) 调用算法 2 构造页面块训练样本集 TS.

(2) 在 TS 上训练 SVM 分类器，得到训练模型 M. 本文采用高斯径向核函数作为空间变换函数.

(3) 对于待分类的页面块，先根据式(5)对页面块进行特征表示，然后使用 M 进行分类.

在实际的系统实现中，我们采用了开源项目 libsvm① 来实现 SVM 分类器.

5　实验评估

为了验证本文提出方法的有效性，我们使用 Java 语言在 Eclipse 平台上实现了相应的原型系统——WPBMiner. 该原型系统的输入是给定的 Web 页面，

① http://www.csie.ntu.edu.tw/～cjlin/libsvm/

输出是该页面的分块及相应的块分类标记. 实验环境为 CPU(Intel Core Duo Processor,1.73 GHz)+RAM (2 GB)+Window XP+Eclipse3.4.

5.1 数据集

本文所使用的数据集 SLPWebSet 是我们通过半手工方式(种子 URL+爬虫+手工筛选)从互联网上收集得到的. 该数据集包括 3200 个页面,分别来自新浪、搜狐、IT168、东方财富、百度知道、51job、CSDN、猫扑、土豆网等国内 10 多个著名的门户网站以及论坛、博客. 主题类目分别涉及新闻、财经、购物、IT 产品、招聘、房产、视频等类别. 数据集中的页面涵盖了新闻页面、产品页面、论坛页面、博客页面、视频页面等多种类型,以验证提出的方法对不同类型页面处理的鲁棒性. 而在具体实验过程中,我们又从 SLPWebSet 中产生了 3 个子数据集:(1) 页面分割数据集 SLPWebSet-1. 包括 1000 个页面,从 SLPWebSet 中随机抽取,但要求覆盖到每个站点和主题. 经过页面分割后,一共得到 8623 个页面分块;(2) 标签传播数据集 SLPWebSet-2. 包括 500 个页面,4306 个页面分块;(3) 页面块分类数据集 SLPWebSet-3. 包括 500 个页面,4275 个页面分块. 这里 SLPWebSet-1、SLPWebSet-2 和 SLPWebSet-3 互不相交.

5.2 评价指标

本文的实验主要分为 4 个部分:页面分割、基于标签传播的训练集扩展、页面块分类和 Web 聚焦爬行. 下面分别介绍各部分的评价指标.

5.2.1 基于节点密度熵的 Web 页面分割

对页面分割算法的评价,本文采用了传统的召回率(Recall)指标. 其定义如下:

$$R@N=\frac{N_s}{N_r} \tag{11}$$

式中,N_s是当分割页面数为 N 时,由算法 1 得到的正确的分割页面块的数量. N_r是当分割页面数为 N 时,实际正确的分割页面块的数量.

由于对页面分割结果的评价是一个较主观的过程,为了给出尽可能客观的结果,我们采用了如下的评价过程:

(1) 聘请了 3 个人员独立从事页面手工分割工作,主要从视觉的角度对测试页面进行分割并标记. 他们相互之间不了解其他人的评价结果,以获取客观的评价结果.

(2) 将 3 个人员的分割结果进行表决,最后得到对测试集页面的手工分割结果.

(3) 使用算法 1 对测试集进行自动页面分割,然后将分割结果与手工标记的结果进行对比和统计,最后计算出 $R@N$.

为了获取算法 1 中 α、β 和 γ 这 3 个分割控制参数的最优值,本文专门设计了相关实验,给出了在不同参数组合条件下其页面分割情况.

5.2.2 基于标签传播的训练集扩展

对基于标签传播的训练集扩展的评价,我们采用了传统的精度(Precision)指标,其定义如下:

$$P@N=\frac{|S_r\cap S_g|}{|S_g|} \tag{12}$$

式中,N 是指当前测试数据集中未标记页面块的数量. S_r是测试数据集中具有真实类别标记的所有页面块的集合. S_g为使用算法 2 后,测试数据集中被自动标记类别后的所有页面块集合.

5.2.3 基于 SVM 的页面块分类

对页面块分类的评价采用了精度、召回率和 $F1$ 指标,精度定义如下:

$$P@N=\frac{tp}{tp+fp} \tag{13}$$

式中,tp 是指测试数据集中,当页面块的数量为 N 时,被正确分类为正例的数据数量;fp 是指被错误分类为正例的数据数量.

召回率定义如下:

$$R@N=\frac{tp}{tp+fn} \tag{14}$$

式中,tp 含义同式(13);fn 是指被错误分类为反例的数据数量.

而 $F1$ 定义为

$$F1@N=\frac{2P@N\times R@N}{P@N+R@N} \tag{15}$$

5.2.4 Web 聚焦爬行实验

对 Web 聚焦爬行实验结果的评价,我们采用了平均收益率(Average Harvest Rate). 该指标用于模拟精度指标(Precision). 其定义[20]如下:

$$harvest_rate@N=\frac{1}{|V|}\sum_{i\in V}r_i \tag{16}$$

式中,$harvest_rate@N$ 是指在 t 时刻,系统已爬取的页面数量为 N 时的收益率. V 是当前系统已经爬取的页面集合. r_i代表页面 i 与主题的相似度,如果页面 i 与主题相似,则 r_i取 1;否则取 0.

5.3 实验结果及分析

按照 5.2 节的评价指标,我们对本文提出的方法进行了系统实现,并在 SLPWebSet 数据集上进行了实验. 下面分别按照系统实现的步骤,对实验结

果进行分析.

5.3.1　基于节点密度熵的 Web 页面分割实验

(1) 页面分割控制参数取值分析

为了确定算法 1 中页面分割控制参数 α、β 和 γ 的最优组合，我们采用了正交实验法来确定这 3 个参数的近似最优组合. 根据初步的实验和经验，我们首先确定了 3 个参数的大致取值范围. 其中，α 的取值在 0.4～0.6；β 的取值在 0.10～0.25；γ 的取值在 0.03～0.2. 为此选用正交表 $L_{25}(5^6)$，即选用 5 水平、6 因素正交表，这样最多进行 25 个单独实验，便可确定最优参数组合. 对于本文，只需选取 3 个因素，分别使用正交表的 1、3、6 列.

表 2 给出了实验因素和水平的取值情况. 实验结果选取系统的平均页面分割召回率 $R@50$ 为指标. 为此，我们从 SLPWebSet-1 数据集上随机抽取了 50 个来自不同站点且具有不同布局风格的页面，然后计算其 $R@50$.

表 2　实验因素和水平

水平	因素 1	因素 2	因素 3
	α	β	γ
1	0.40	0.25	0.20
2	0.45	0.20	0.15
3	0.50	0.15	0.10
4	0.52	0.12	0.05
5	0.55	0.10	0.03

表 3 给出了实验结果和分析. 通过极差可以看到，参数的主次因素顺序为 α、γ、β，其中 α 对分割召回率的影响较大，这与我们的实际观察是吻合的. 此外，最优参数组合为 $(\alpha_4, \beta_4, \gamma_4)$，即 (0.52, 0.12, 0.05).

表 3　实验结果和分析

实验号	因素1 α	因素2 β	因素3 γ	实验结果 $R@50$	实验号	因素1 α	因素2 β	因素3 γ	实验结果 $R@50$
1	1	1	1	0.41	17	4	5	4	0.85
2	1	2	2	0.56	18	4	1	5	0.83
3	1	3	3	0.63	19	4	2	1	0.82
4	1	4	4	0.76	20	4	3	3	0.84
5	1	5	5	0.73	21	5	5	2	0.80
6	2	2	5	0.72	22	5	1	3	0.87
7	2	3	1	0.71	23	5	2	4	0.80
8	2	4	2	0.79	24	5	3	4	0.81
9	2	5	3	0.72	25	5	4	5	0.79
10	2	1	4	0.73	t_1	0.618	0.696	0.682	
11	3	3	4	0.77	t_2	0.734	0.722	0.698	
12	3	4	5	0.76	t_3	0.712	0.752	0.773	
13	3	5	1	0.68	t_4	0.842	0.794	0.782	
14	3	1	2	0.64	t_5	0.814	0.756	0.770	
15	3	2	3	0.71	极差	0.224	0.098	0.100	
16	4	4	3	0.87		α_4	β_4	γ_4	

(2) 页面分割平均召回率结果分析

图 4 显示了在页面分割控制参数 $(\alpha, \beta, \gamma)=(0.52, 0.12, 0.05)$ 下，系统的平均召回率 $R@N$ 的情况. 其中，X 轴表示系统已分割的页面数，Y 轴表示在系统已分割的页面数量为 N 时平均召回率. 我们从 SLPWebSet-1 数据集上随机抽取了 500 个页面进行页面分割实验，并统计了在分割页面数量递增的情况下的平均召回率. 从图 4 上可以看出，$R@N$ 的值具有一定的波动性，但随着 N 的增加 $R@N$ 的值趋于稳定，并保持在 0.82 以上. 由此表明了本文提出的页面分割算法的有效性，同时也证明了该算法独立于待分割页面的布局风格和内容.

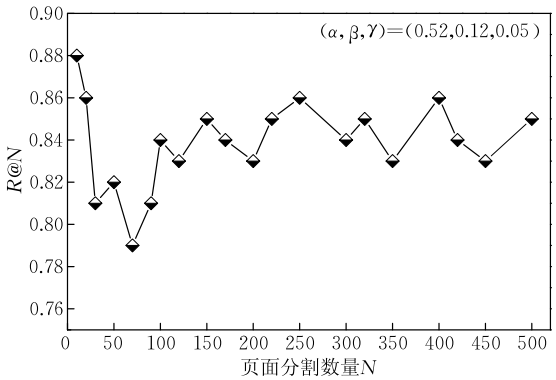


图 4　基于节点密度熵的 Web 页面分割算法的平均召回率

5.3.2　基于 KNNLP 算法的训练集扩展算法实验

(1) 增量添加未知标签页面块数 k 的取值分析

算法 2 中每次需要增量添加 k 个未标记页面块到初始已标记训练集中. 其中 k 的取值是算法的关键，它决定了算法的精度. 为了确定 k 的最佳取值，我们选取了基于 SVM、KNN($K=4$) 的分类方法与 KNNLP 算法进行对比实验. 针对本文使用的 KNN 算法，我们做了 $K=1$～5 的对比实验，发现 K 越大，分类精度越高，但 $K=4$ 与 $K=5$ 差别不大，考虑到算法效率，取 $K=4$ 较合适. 初始的未标记数据集 M 是从数据集 SLPWebSet-2 中随机抽取的 100 个页面，然后使用算法 1 对其进行页面分割，共产生 876 个块. 再分别向 M 中添加 r%(此处 r 是指已标记页面块占 M 数量的百分比) 的标记页面块.

图 5 给出了实验结果. 其中，X 轴表示训练集中已标记页面块占未标记页面块数量的百分比. Y 轴表示在不同 r 下的平均标签分类精度. KNNLP 算法的最优近邻数取 7. 从图 5 中可以看到 3 种算法的平均 $P@r$(与 $P@N$ 的定义略有不同，但作用完全相同) 随着 r 的增加均呈上升趋势. 在 r 较小时，由于已标记的页面块数量较少，基于监督的 SVM

算法精度对训练集的大小较敏感，所以此时其 $P@r$ 较低. 而非监督的 KNN 算法由于缺少足够的标记数据，因此初始时，$P@r$ 也较低；对半监督的 KNNLP 算法，由于能利用非标记数据和标记数据内在结构的相似性，因此在初始时，其 $P@r$ 明显高于 SVM 和 KNN 算法. 随着 r 的增加，3 种算法的精度都得到了提高，其中 KNN 和 SVM 算法提高得较显著，而 KNN 算法在 $r>10\%$ 后，迅速提高到 0.6 以上，但其后基本趋于平缓. 这表明当数据集中的已标记数据达到一定数量后，KNN 算法对已标记数据数量的增加并不敏感，对其产生影响的只是其 K 个最近邻（如 $K=4$）. 总体来看 3 种算法中 KNNLP 算法取得的精度最好. 而当 $r>150\%$ 时，KNNLP 算法的精度达到 0.86 以上. 因此本文确定 $k=1/1.8$ 为最优值，即未标记页面块和标记页面块数量的比率在 1∶1.8. 这样可以在保证 KNNLP 算法精度的前提下，通过设置较小的初始未标记页面块数量（这样标记页面块数量也相对较小，从而减少了手工标注的耗费），来自动扩展训练集.

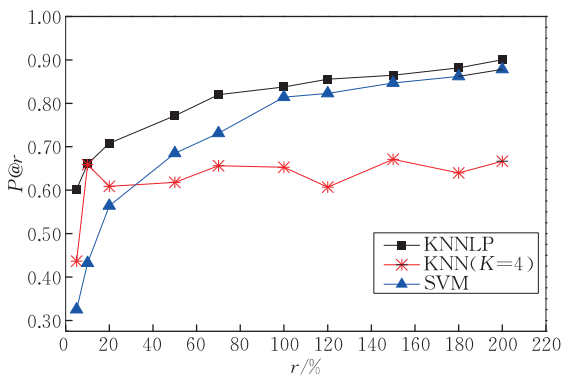


图 5 不同 r 下的平均标签分类精度

（2）KNNLP 算法实验结果分析

图 6 显示了算法 2 取得的平均精度的情况. 其中，X 轴表示 KNNLP 算法迭代的次数，右侧 Y 轴表示算法迭代次数为 n 时的平均标签分类精度，左侧 Y 轴表示当前训练集中已标记的页面块数量. KNNLP 算法的最优近邻数取 7. 按上一小结的实验，我们取 $k=1/1.8$，初始的已标记数据集是从数据集 SLPWebSet-2 中随机抽取的 5 个页面，共 44 个块，对其进行手工标记. 然后分别按比例 k 向该已标记数据集中添加未标记页面块，即每轮添加 25 个未标记页面块，共进行了 53 轮添加，并测试在不同 n 下算法的 $P@N$. 图 6 中灰色的柱状图表示每次迭代后，当前扩展训练集中已标记页面块的总数量. 曲线为平均精度曲线 $P@N$，可以看到 $P@N$ 呈一定的波动性. 其原因是每轮用于扩展的已标记训练集（44 个）是固定的，而每轮加入的 25 个未标记页面块是不同的，因此每次扩展时，其 $P@N$ 出现一定的波动. 需要指出的是算法 2 提出的扩展策略保证了本轮扩展产生的标签分类误差不会传播到下一轮迭代. 因此，算法的总体标签分类误差等于每轮迭代分类误差的平均值：$\varepsilon=\frac{1}{n}\sum_{i=1}^{n}\varepsilon_i$. 从图中可看到，当 $n=53$ 时，$N=1369$，算法的平均 $P@N$ 约为 0.87. 对后续的 SVM 分类器来说，其精度和已标记页面块的数量已经能够达到分类的要求.

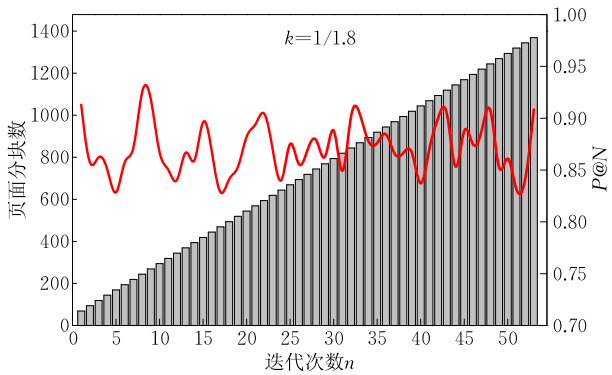


图 6 基于 KNNLP 的训练集扩展算法平均精度

5.3.3 基于 KNNLP+SVM 的页面块分类算法实验

为了评价 KNNLP+SVM 页面块分类算法的实际运行效果，在相关文本块识别实验中，我们选取基本标签传播算法（LP）[16]+SVM、LNP[18]+SVM、半监督 SVM（S^3VM）算法[19]、KNN（$K=4$）算法和 CECTD-DS 算法[4]进行对比实验. 其中 S^3VM 算法采用了 SVM^{light} 工具①，该工具的原理是：训练数据集中包含少量的标记数据和大量未标记数据，算法直接在该数据集上学习分类模型，然后使用该模型进行分类.

在相关链接块识别实验中，因为 CECTD-DS 只能识别页面主要文本块，因此我们只选取了 LP+SVM、LNP+SVM、S^3VM 进行对比实验.

测试数据是从数据集 SLPWebSet-3 中随机抽取 8 组测试数据，每组页面数 N 分别取 10～250 不等（见表 4 和表 5 所示）.

下面对 6 种算法使用的训练集做一说明：

（1）对 CECTD-DS 算法来说，并不需要训练数据，只需直接对输入的 Web 页面进行识别即可.

① http://svmlight.joachims.org/

(2) 对于 KNNLP＋SVM 算法来说，使用的训练集为上一小节实验获得的扩展训练集 TS，TS 共包含 1369 个页面块，分为 3 个类别(相关文本块、相关链接块、其他类别块). 而对于 LP＋SVM、LNP＋SVM 算法来说，其获取训练集的过程与 KNNLP＋SVM 算法相似，区别仅在于采用的标签传播算法不同.

(3) 对 S^3VM 算法来说，其训练集中包含标记训练集和未标记训练集. 其中标记训练集为 KNNLP＋SVM 算法采用的初始标记训练集，即从数据集 SLPWebSet-2 中随机抽取的 5 个页面，共 44 个块，而未标记训练集为其后添加的未标记数据集(见上一小节). 而 KNN($K=4$)算法采用的训练集只使用了 KNNLP＋SVM 算法的初始标记训练集. 表 4 给出了相关文本块识别的实验结果. 从表 4 可以看到，在页面相关文本块的识别方面，6 种算法的 $P@N$ 和 $R@N$ 随着 N 的增加均呈一定的下降趋势，但下降幅度不大. 总的来说，$P@N$ 要好于 $R@N$. 从 $F1@N$ 指标来看 KNNLP＋SVM 算法性能最好，而 CECTD-D 算法性能与之较接近；LNP＋SVS 算法的性能好于 LP＋SVM 算法，LP＋SVM 算法的性能好于 S^3VM 算法，而 KNN 算法的性能最差.

表 4　页面相关文本块识别实验结果

N	KNNLP＋SVM			LNP＋SVM			LP＋SVM		
	$P@N$	$R@N$	$F1@N$	$P@N$	$R@N$	$F1@N$	$P@N$	$R@N$	$F1@N$
10	0.885	0.785	0.832	0.832	0.766	0.798	0.792	0.735	0.762
30	0.871	0.779	0.822	0.816	0.723	0.767	0.781	0.726	0.752
50	0.855	0.782	0.817	0.805	0.726	0.763	0.772	0.693	0.730
100	0.820	0.752	0.785	0.797	0.721	0.757	0.757	0.681	0.717
120	0.818	0.758	0.787	0.787	0.716	0.750	0.723	0.674	0.698
150	0.823	0.739	0.779	0.772	0.693	0.730	0.732	0.651	0.689
200	0.818	0.731	0.772	0.763	0.689	0.724	0.716	0.668	0.691
250	0.827	0.722	0.771	0.758	0.677	0.715	0.706	0.678	0.692

N	S^3VM			KNN($K=4$)			CECTD-DS		
	$P@N$	$R@N$	$F1@N$	$P@N$	$R@N$	$F1@N$	$P@N$	$R@N$	$F1@N$
10	0.781	0.726	0.752	0.724	0.661	0.691	0.862	0.763	0.809
30	0.763	0.692	0.726	0.695	0.662	0.678	0.841	0.758	0.797
50	0.772	0.693	0.730	0.682	0.654	0.668	0.823	0.752	0.786
100	0.757	0.681	0.717	0.665	0.647	0.656	0.815	0.756	0.784
120	0.723	0.674	0.698	0.672	0.642	0.657	0.791	0.738	0.764
150	0.695	0.662	0.678	0.658	0.647	0.652	0.814	0.712	0.760
200	0.724	0.661	0.691	0.643	0.628	0.635	0.798	0.695	0.743
250	0.712	0.651	0.680	0.625	0.613	0.619	0.812	0.686	0.744

表 5 给出了相关链接块识别的实验结果. 与表 4 的实验数据显示的结果类似，从 $F1@N$ 指标来看，4 种算法中，KNNLP＋SVM 算法性能较好，在较大的实验数据集上($N=250$)，其 $F1@N$ 达到了 0.715，LNP＋SVM 算法性能次之，而 S^3VM 算法性能最差.

表 5　页面相关链接块识别实验结果

N	KNNLP＋SVM			LNP＋SVM		
	$P@N$	$R@N$	$F1@N$	$P@N$	$R@N$	$F1@N$
10	0.835	0.734	0.781	0.825	0.736	0.778
30	0.831	0.731	0.778	0.814	0.725	0.767
50	0.815	0.745	0.778	0.821	0.736	0.776
100	0.793	0.738	0.765	0.807	0.716	0.759
120	0.791	0.726	0.757	0.795	0.706	0.748
150	0.786	0.715	0.749	0.773	0.694	0.731
200	0.772	0.688	0.728	0.765	0.692	0.727
250	0.764	0.672	0.715	0.752	0.677	0.713

N	LP＋SVM			S^3VM		
	$P@N$	$R@N$	$F1@N$	$P@N$	$R@N$	$F1@N$
10	0.768	0.735	0.751	0.756	0.682	0.717
30	0.773	0.721	0.746	0.738	0.674	0.705
50	0.765	0.715	0.739	0.741	0.671	0.704
100	0.753	0.693	0.722	0.726	0.663	0.693
120	0.746	0.691	0.717	0.718	0.654	0.685
150	0.725	0.686	0.705	0.706	0.656	0.68
200	0.715	0.667	0.690	0.692	0.651	0.671
250	0.708	0.652	0.679	0.681	0.645	0.663

总之，从表 4 和表 5 的实验结果看，本文提出的方法在页面块分类方面是有效的. 另外，采用标签传播(KNNLP、LNP 和 LP)＋SVM 算法的页面块分类效果要好于 S^3VM 算法和 KNN($K=4$)算法. 其原因可能在于：标签传播＋SVM 算法充分利用了未标记数据的结构特征，从而提高了分类的性能.

5.3.4　基于 KNNLP＋SVM 的 Web 聚焦爬行实验

上节的两个实验只是从相关文本块和相关链接块识别两个方面单独进行的，但如引言部分所述，本文提出的 Web 页面重要内容挖掘统一框架的重要意义还在于充分利用 Web 页面块内在的结构和语义关系. 除了相关文本块和相关链接块识别和抽取的基本功能外，本文提出的方法还可以应用到其他许多 Web 应用领域，如 Web 页面多主题内容抽取、Web 页面噪音内容过滤、Web 聚焦爬行等. 下面给出使用 KNNLP＋SVM 方法用于辅助 Web 聚焦爬行的实验.

Web 聚焦爬行对于搜索引擎有着重要的作用[20-21]. 在本实验中，我们拟通过利用主要文本块来辅助爬虫增强主题链接的识别精度，以提高 Web 聚焦爬行的爬行收益率. 实验采用了较经典的 ATA (Anchor Text Analysis)聚焦爬行算法[20]，该算法主要通过分析锚链文本来获取链接主题相关的评分，以控制聚焦爬行的方向. 提高 Web 聚焦爬行收益率的关键在于精确估算页面中链接的主题相关度. 本文利用了主要文本块来增强主题链接的识别

精度.通过观察和实验,我们确定了如下的启发式规则:

(1)主要文本块中包含的链接具有较强的主题相关性.

(2)邻近主要文本块的链接块所包含的链接具有较强的主题相关性.

为此我们制定了如下的爬行策略:首先爬取并分析主要文本块中的链接,然后是主要文本块近邻链接块中的链接,最后是其他链接块.

本实验采用了文献[20]的数据集,并在 3 个不同的主题域(手机、数码相机、笔记本电脑)上进行了实验.

图 7 显示了两种爬行策略在 3 个不同的主题域上取得的平均收益率情况.图 7 横坐标表示系统已爬行的网页数量,纵坐标表示在系统已爬取的页面数量为 N 时平均收益率 $harvest_rate@N$(参见式(16)).从图 7 可以看到,两种爬行策略随着 N 的增加均出现了下降的趋势.出现下降趋势的主要原因是聚焦主题具有很强的稀疏性,在爬行过程中,已发现的主题相似页面的数量和系统已爬取的页面数量的增量幅度不同,很显然前者是小于后者的,所以造成平均收益率总体上出现下降.从图 7 中还可以看到,KNNLP+SVM+ATA 的平均收益率明显高于 ATA,在 0.79~0.52 之间.由于使用了 KNNLP+SVM 算法对页面块进行分类,并采用了优先爬取并分析主要文本块以及近邻链接块中链接的策略,增加了爬取主题相关链接的概率,因此显著提高了平均收益率.ATA 的平均收益率在 0.68~0.37 之间,由于其仅仅使用了锚链文本的分析技术,但无法利用页面中链接与相关文本块主题相似度的语义关系,所以当带有相似锚链文本的多个链接出现在页面中的不同块中时(可能属于不同的主题),容易造成爬行主题的漂移,从而使其平均收益率出现下降.

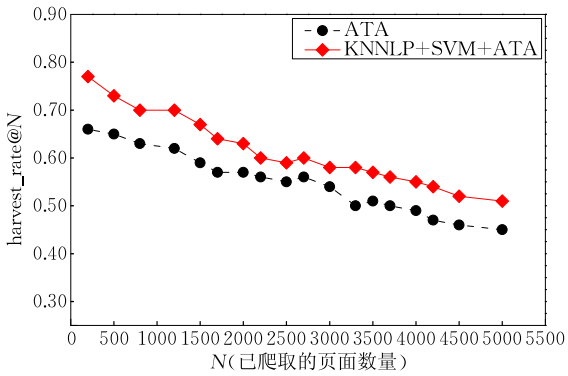


图 7 KNNLP+SVM+ATA 和 ATA 两种爬行算法的平均收益率

6 结 语

获取 Web 页面中的重要内容如文本和链接,是许多 Web 研究领域的热点问题.本文提出了一种 Web 页面重要内容挖掘的统一框架,采用该框架可以将 Web 页面块精确区分为多种类型,并且该框架独立于 Web 页面的类型和布局.通过真实 Web 环境下进行广泛的实验,证明该方法是非常有效的.

然而本文在所提出的方法及其应用方面还存在一些不足:(1)对 KNNLP 算法来说,平衡因子 δ 的选择是一个难点问题,尽管本文提出了一个定量公式,但对其机理未做深入研究;(2)初始已标记数据集对基于 KNNLP 的训练集扩展算法的精度具有一定影响,对如何选择已标记数据集的问题,本文未加讨论;(3)本文提出的 Web 页面重要内容挖掘的统一框架除了能够完成基本的页面块有效分割和识别外,其更重要的意义应在于如何充分利用 Web 页面块内在的结构和语义关系,进行深层次的数据挖掘,以更好地应用到其他 Web 应用中.但本文对此问题,仅作了基于 KNNLP+SVM 辅助的 Web 聚焦爬行实验,而对其他方面的数据挖掘工作未做更深入的探讨.以上这些问题需要在未来的工作中加以解决.

参 考 文 献

[1] Yin Xinyi, Lee Wee Sun. Using link analysis to improve layout on mobile devices//Proceedings of the 13th International Conference on World Wide Web (WWW 2004). New York, USA, 2004: 338-344

[2] Chen Yu, Ma Wei-Ying, Zhang Hong-Jiang. Detecting Web page structure for adaptive viewing on small form factor devices//Proceedings of the 12th International Conference on World Wide Web (WWW 2003). Budapest, Hungary, 2003: 225-233

[3] Baluja S. Browsing on small screens: Recasting Web-page segmentation into an effcient machine learning framework//Proceedings of the 15th International Conference on World Wide Web (WWW 2006). Edinburgh, Scotland, 2006: 33-42

[4] Sun Fei, Song Dandan, Liao Lejian. DOM based content extraction via text density//Proceedings of the 34th Annual International ACM SIGIR Conference (SIGIR 2011). Beijing, China, 2011: 245-254

[5] Cai Deng, Yu Shipeng, Wen Ji-Rong, Ma Wei-Ying. Extracting content structure for Web pages based on visual representation //Proceedings of the 5th Asian-Pacific Web Conference (APWeb 2003). Xi'an, China, 2003: 406-417

[6] Yi Lan, Liu Bing, Li Xiaoli. Eliminating noisy information in Web pages for data mining//Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2003). Washington, USA, 2003: 296-305

[7] Ramaswamy L, Iyengar A, Liu Ling, Douglis F. Automatic fragment detection in dynamic Web pages and its impact on caching. IEEE Transactions on Knowledge and Data Engineering (TKDE), 2005, 17(6): 859-874

[8] Debnath S, Mitra P, Pal N, Giles C L. Automatic identification of informative sections of Web pages. IEEE Transactions on Knowledge and Data Engineering (TKDE), 2005, 17(9): 1233-1246

[9] Kolcz A, Yih Wen-tau. Site-independent template-block detection//Proceedings of the 11th European Conferenceon on Principles and Practice of Knowledge Discovery in Databases (PKDD 2007). Warsaw, Poland, 2007: 152-163

[10] Chakrabarti D, Kumar R, Punera K. Page-level template detection via isotonic smoothing//Proceedings of the 16th International Conference on World Wide Web (WWW 2007). Banff, Canada, 2007: 61-70

[11] Wang Yu, Fang Binxing, Cheng Xueqi, et al. Incremental Web page template detection//Proceedings of the 17th International Conference on World Wide Web (WWW 2008). Beijing, China, 2008: 1247-1248

[12] Chakrabarti D, Kumar R, Punera K. A graph-theoretic approach to webpage segmentation//Proceedings of the 17th International Conference on World Wide Web (WWW 2008). Beijing, China, 2008: 377-386

[13] Bing Lidong, Guo Rui, Lam Wai, et al. Web page segmentation with structured prediction and its application in Web page classification//Proceedings of the 37th Annual International ACM SIGIR Conference (SIGIR 2014). Gold Coast, Australia, 2014: 245-254

[14] Kohlschütter C. A densitometric analysis of Web template content//Proceedings of the 18th International Conference on World Wide Web (WWW 2009). Madrid, Spain, 2009: 1165-1166

[15] Kohlschütter C, Nejdl W. A densitometric approach to Web page segmentation//Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM 2008). California, USA, 2008: 1173-1182

[16] Zhu Xiaojin, Ghahramani Zoubin. Learning from labeled and unlabeled data with label propagation. Pennsylvania, America: Carnegie Mellon University, Technical Report: CMU-CALD-02-107, 2002

[17] Zhou Dengyong, Bousquet O, Lal T N, et al. Learning with local and global consistency//Proceedings of the 17th International Conference on Neural Information Processing Systems (NIPS 2003). Vancouver, Canada, 2003: 321-328

[18] Wang Fei, Zhang Changshui. Label propagation through linear neighborhoods. IEEE Transactions on Knowledge and Data Engineering (TKDE), 2008, 20(1): 55-67

[19] Gieseke F, Airola A, Pahikkala T, Kramer O. Fast and simple gradient-based optimization for semi-supervised support vector machines. Neurocomputing (IJON), 2014, 123(1): 23-32

[20] Zhang Nai-Zhou, Li Shi-Jun, Yu Wei, Zhang Zhuo. Using a joint link similarity evaluation based method for crawling the resources on Web. Chinese Journal of Computers, 2010, 33(12): 2266-2280(in Chinese)
(张乃洲，李石君，余伟，张卓. 使用联合链接相似度评估爬取 Web 资源. 计算机学报，2010，33(12)：2266-2280)

[21] Diligenti M, Coetzee F, Lawrence S, et al. Focused crawling using context graphs//Proceedings of the 26th International Conference on Very Large Data Bases (VLDB 2000). Cairo, Egypt, 2000: 527-534

ZHANG Nai-Zhou, born in 1970, Ph. D., associate professor. His main research interests include Web search and mining.

CAO Wei, born in 1977, M. S., associate professor. Her main research interests include machine learning, Web data mining.

LI Shi-Jun, born in 1964, Ph. D., professor, Ph. D. supervisor. His main research interests include database technology, Web data management and mining.

Background

Nowadays Web pages present rich and colorful content to meet business information needs. For many Web applications, however, too rich content may be an obstacle. For example, they can hamper Web data mining task to some extent, cause crawling topic drift, lead to a serious waste of storage space and network bandwidth, etc. So, how to get

the important content in a Web page, such as text and links, has a significant research and application value.

At present, the main method for this problem is to adopt Web page segmentation and informative sections recognition. From the point of view of research goals, the current research on this problem mainly focused on Web page content extraction, whose goal is to identify the main text contents in a Web page. While from the point of view of the main technique used, Web page content extraction can be mainly classified into two categories: one adopts sampling method to get a certain quantity of Web pages from the same Web site, and then find important patterns. Another exploits single Web page to find import regions.

However, one major drawback of current research is that they use decoupled strategies that attempt to do text content and link content identification in two separate phases. This cuts off the inner semantic relationships between texts and links in a Web page. At the same time, this also results in low efficiency of the processing of Web page. To this end, in this paper, our main research goals are to propose a uniform framework for mining important content in a Web page. On the one hand, this framework can simultaneously fulfill both tasks in one processing of a Web page. On the other hand, using this framework, inner semantic relationships between texts and links in a Web page can be made full use of providing services for further data mining.

This work is supported by the National Natural Science Foundation of China under Grant No. 61272109 and No. 61202285, National Spark Program under Grant No. 2012GA750007, Basis and Cutting-Edge Research Project of Science and Technology Department of Henan Province under Grant No. 122300410378, Key Projects of Science and Technology Research of Education1 Department of Henan Province under Grant No. 13A520032. Our research team has been working on the research field in Web search and mining for many years. So far, we have published over 20 papers on this research field.