

基于排序学习方法的本体映射算法

高 炜¹, 兰美辉²

(1 云南师范大学 信息学院, 云南 昆明 650092; 2 曲靖师范学院 计算机科学与工程学院, 云南 曲靖 655011)

摘 要: 本体映射作为实现多本体间相互操作的重要手段, 已广泛应用于诸多科学领域. 根据本体自身的结构设计边权值函数和函数, 利用排序算法将多本体图映射成实直线, 通过比较两概念对应实数间的差值或排名的接近程度判断它们的相似程度, 由此得到对应的本体映射. 实验表明, 该算法有较高的效率.

关键词: 本体映射; 排序; 铰接亏损函数

中图分类号: TP391.3

文献标识码: A

文章编号: 1000-7180(2011)09-0059-03

Ontology Mapping Algorithm Based on Ranking Learning Method

GAO Wei¹, LAN Mei-hui²

(1 Department of Information, Yunnan Normal University, Kunming 650092, China;

2 Department of Computer Science and Engineering, Qujing Normal University, Qujing 655011, China)

Abstract: Ontology mapping as an important means of multi ontologies mutual operation has been widely used in many fields. We propose that the weight functions and function can be designed based on the structure of ontology. Via the ranking learning algorithm, the multiple ontology graph is mapped into a line consists of real numbers. The similarity between two concepts then can be measured by comparing the difference between their corresponding real numbers or the proximity of their positions on ranking list, and thus get the ontology mapping. Experimental results show that the proposed algorithm has high accuracy and efficiency.

Key words: ontology mapping; ranking; hinge loss function

1 引言

近年来, 排序学习算法由于应用价值广泛而受到越来越多的关注^[1-2]. 排序函数是一个得分函数. 它将每个待排序对象映射成一个实数, 通过比较实数间的大小来对实例进行排名, 再将实例按照排序的顺序以列表的方式返回给用户.

本体(Ontology)一词来源于哲学领域, 用来描述事物的本质^[3]. 在计算机科学领域, 本体被定义为共享概念模型的形式化规范说明^[4]. 目前本体已应用在智能信息集成、协作信息系统、信息检索、电子商务和知识管理等领域. 作为一种有效表现概念层次结构和语义的模型, 本体技术目前已经具备比较

系统、完善的工程理论、表示方法和构造工具. 本体映射是解决本体异构问题、实现多本体间相互操作的重要手段. 本体映射算法是往往通过计算相似度得到^[5]; 对于本体 O_1 中的每个元素, 在本体 O_2 中找到与其在语义上相同或相似的元素, 反之亦然. 本体映射函数 $\text{map} : O_1 \rightarrow O_2$ 的形式化定义为: 确定阈值 M , 若 $\text{Sim}(A, B) > M$, 则 $\text{map}(A) = B$, 其中 A, B 分别是 O_1 和 O_2 中的元素. 类似地, 可定义多本体间的映射.

文献[6]提出了图顶点排序的学习算法, 文献[7]将其算法应用于本体相似度计算. 文中将图顶点排序的方法应用于本体映射. 设图 $G_1, G_2, \dots, G_m$ 分别对应本体 $O_1, O_2, \dots, O_m$, 令 $G = G_1 + G_2 + \dots +$

G_m . 通过排序算法将 G 中的所有顶点排成一个序列, 从而通过两概念对应顶点间的排序关系判断它们的相似程度, 得到对应的本体映射.

2 图顶点的排序学习算法

设 $G = (V, E, w)$ 是一个无向带权图, V 是顶点集, E 是边集, $w: E \rightarrow R^+$ 为定义在边集上的权值函数, 它的大小在一定程度上反映两顶点关联程度. 图中顶点 v_i 的度 $d(v_i)$ 定义为:

$$d(v_i) = \sum_{(v_i, v_j) \in E} w(v_i, v_j) \quad (1)$$

图顶点的排序算法的目标是得到一个实值函数 $f: V \rightarrow R$, 它将各顶点映射成实数. 对于图 G 中任意两个顶点 v_i, v_j , 若 $f(v_i) > f(v_j)$, 则认为顶点 v_i 的排名比顶点 v_j 高.

在初始阶段, 专家对图中若干个顶点对 (v_i, v_j) 给出它们的次序关系. 由此构成原图 G 的次序图 $\Gamma = (V, \Sigma, \tau)$, 其中 $\Sigma \subseteq V \times V$ 为专家所给次序对集合, $\tau: \Sigma \rightarrow R^+$ 为惩罚函数, 即当 $f(v_i)$ 与 $f(v_j)$ 的大小关系和专家给出的排序次序不一致时, 对排序函数作一个惩罚, 其值用 $\tau(v_i, v_j)$ 来衡量. 图顶点排序学习算法的过程就是如何通过 Γ 得到最优的 f . 文中约定若 $(v_i, v_j) \in \Sigma$, 则 v_i 的排名比 v_j 高.

衡量一个排序函数 f 的标准是它的亏损是否小. 假设顶点对内两顶点之间的关系被随机地打破, 亏损函数 $l(f; v_i, v_j)$ 用来衡量当 $f(v_i)$ 与 $f(v_j)$ 的大小关系与 v_i, v_j 的实际排名顺序不符时的亏损量. 在具体应用中一般使用铰接亏损函数:

$$l_h(f; v_i, v_j) = (1 - f(v_i) - f(v_j))_+ \quad (2)$$

$$\text{其中 } \alpha_+ = \begin{cases} a & \text{若 } a > 0 \\ 0 & \text{否则} \end{cases}$$

在排序函数 f 下关于次序图 Γ 的经验排序误差可表示为

$$R_h = (f; \Gamma) = \frac{1}{|\Sigma|} \sum_{(v_i, v_j) \in \Sigma} \tau(v_i, v_j) l_h(f; v_i, v_j) \quad (3)$$

惩罚函数 τ 和亏损函数 l 都是对误排的惩罚, 它们的区别在于惩罚函数 τ 只与图结构有关, 与 f 无关, 而 l 则由排序函数决定.

但是, 只最小化 $R_h = (f; \Gamma)$ 得到的排序函数往往是不稳定的, 光滑性较差, 上下波动较大. 解决这一问题的方法是在 $R_h = (f; \Gamma)$ 之后增加一个正则项, 当排序函数 f 上下波动很大时, 该正则项会变的很大, 因此该项可以做到平衡的作用. 关于函数 f

在顶点 v_i 处沿着边 (v_i, v_j) 的边导数定义为^[8]:

$$\frac{\partial f}{\partial (v_i, v_j)} \Big|_{v_i} = \sqrt{\frac{w(v_i, v_j)}{d(v_i)}} f(v_i) - \sqrt{\frac{w(v_i, v_j)}{d(v_j)}} f(v_j)$$

从而 f 在顶点 v_i 处的局部方差为:

$$\|\nabla_{v_i} f\| = \sqrt{\sum_{j: (v_i, v_j) \in E} \left(\frac{\partial f}{\partial (v_i, v_j)} \Big|_{v_i} \right)^2} \quad (4)$$

正则项可表示为图中所有顶点沿各条关联边的边导数的局部方差的平均值, 即:

$$S(f) = \frac{1}{2} \sum_{i=1}^n \|\nabla_{v_i} f\|^2 \quad (5)$$

因此文中的排序函数通过解如下最小化问题得到:

$$\min_{f: V \rightarrow R} \{R_h = (f; \Gamma) + \lambda S(f)\} \quad (6)$$

其中, $\lambda > 0$ 为正则常数.

3 本体图上顶点排序及映射的建立

通过对上述排序算法的分析可知, 将算法融入本体映射算法, 只需根据本体自身的结构设计边权值函数和函数 τ 即可.

3.1 权值函数的选取

计算边权重的方法一般有以下两类: (1) 各项加权的方法; (2) 核函数方法. 每条边的权值与该边的类型、深度、密度、强度以及两关联顶点的属性等密切相关. 方法(1)是充分考虑上述各种因素对边权重的影响, 通过各部分影响的加权得到边权值. 文献[9]给出了一类具体计算方法如下:

$$w(u, v) = a * \text{类型} + b * \text{深度} + c * \text{密度} + d * \text{强度} + e * \text{属性} \quad (7)$$

其中, $a, b, c, d, e \in [0, 1]$, 且 $a + b + c + d + e = 1$. 关于各项的计算方法可参考文献[9].

使用核函数方法首先需要向量化顶点信息. 设 $u, v \in V(G)$, 可使用以下几种核函数来计算 $w(u, v)$:

$$\text{高斯核 } w(u, v) = e^{-\frac{\|u-v\|^2}{2}} \quad (8)$$

逆多元二次核

$$w(u, v) = (c^2 + \|u - v\|^2)^{-\alpha} \quad (9)$$

其中, c 和 α 均为正数.

3.2 τ 函数的选取

τ 函数主要的功能是体现排序函数误排后的惩罚量, 因此两个越不相似的概念如果被误排则会得到越严重的惩罚. 设 v_i, v_j 是图 G 中任意两个点, 这里 $\tau(v_i, v_j)$ 的值即为 v_i, v_j 的层次差, 即

$$\tau(v_i, v_j) = |\text{level}(v_i) - \text{level}(v_j)| \quad (10)$$

