第 43 卷 第 2 期 计 算 机 工 程 2017 年 2 月
Vol. 43 No. 2 Computer Engineering February 2017

·人工智能及识别技术· 文章编号:1000-3428(2017)02-0227-07 文献标志码:A 中图分类号:TP399

基于路径特征的复杂本体匹配

王汉博,孙启霖

(中国科学院数学与系统科学研究院,北京 100190)

摘 要: 复杂本体匹配方法主要分为基于匹配模式的方法和基于机器学习的方法,前者需要人工制定启发式规则而后者易陷入局部最优解。针对上述问题,提出一种融合匹配模式和机器学习的复杂匹配方法。引入路径特征刻画本体中实例具有的性质,路径特征是匹配模式的具体化。在此基础上,使用一阶归纳学习器组合路径特征得到复杂的匹配结果。实验结果表明,该方法可以自动学习到本体间的复杂匹配,且与基于一阶归纳学习器的复杂匹配方法相比,能够有效缓解局部最优问题。

关键词: 链接开放数据;本体;复杂本体匹配;路径特征;属性信息

中文引用格式: 王汉博,孙启霖. 基于路径特征的复杂本体匹配[J]. 计算机工程,2017,43(2):227-233,240.

英文引用格式: Wang Hanbo, Sun Qilin. Complex Ontology Matching Based on Path Feature[J]. Computer Engineering, 2017, 43(2):227-233, 240.

Complex Ontology Matching Based on Path Feature

WANG Hanbo, SUN Qilin

(Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China)

【Abstract】 The complex ontology matching methods are mainly classified into correspondence pattern based methods and machine learning based ones. The former are heuristic ones that are designed with some knowledge about the ontologies to be aligned and the latter are easy to fall into local optimums. This paper proposes a novel complex ontology matching method which benefits from both correspondence pattern and machine learning. The key of the method is the introduction of path feature to characterize the information of instances and path feature is an instantiation of correspondence pattern. Path features are combined by First Order Inductive Learner (FOIL) to acquire complex mappings. Experimental results show that this method automatically learns complex correspondences between ontologies, and effectively relieves the problem of local optimum compared with FOIL-based complex matching method.

【Key words】 Linked Open Data(LOD); ontology; complex ontology matching; path feature; attribute information

DOI: 10.3969/j.issn.1000-3428.2017.02.038

0 概述

近年来,链接开放数据(Linked Open Data, LOD)[1]繁荣发展,构建了大量本体,推动了语义网技术的发展。这些本体包含海量 RDF 三元组并且覆盖了许多领域如人类、科学出版、电影、音乐等。显著的成果包括 DBpedia[2],YAGO[3]等知识库。然而许多描述交叉领域甚至是相同领域的本体在不同层面存在异构性,如语法层面、结构层面和语义层面[4],这阻碍了本体所标注信息的互操作。针对该问题,本体匹配研究如何建立不同本体之间实体(类 class、属性 property 或实例 instance)的对应关系[5]。

为了实现本体匹配,出现了许多自动或半自动的方法[5-6]。大多数方法关注于由本体间单个实体的对应关系构成的 1:1 简单匹配。这种简单匹配存在局限性:表达能力有时不足以体现实体之间的依赖关系。比如,给定 2 个待匹配的本体 YGAO 和 DBpedia。源本体 YAGO 中的类 Film_directors_from_Beijing 描述了一种人:职业是电影导演且出生地为北京。在目标本体 DBpedia 中,不存在一个与该类匹配的实体。但是 DBpedia 中存在刻画职业的属性 occupation 和刻画出生地的属性 birthplace,并且类 Film_directors_from_Beijing 与 DBpedia 中 occupation 属性是“Film_director”,birthplace 属性是“Beijing”的人构成的类语义等价。这种源本体中的一个类(属性)与目标本体中多个类和属性的组合匹

基金项目: 国家自然科学基金(61232015)。

作者简介: 王汉博(1991—),女,硕士研究生,主研方向为模式识别、机器学习;孙启霖,博士研究生。

收稿日期: 2016-02-04 **修回日期:** 2016-03-13 **E-mail:** hbwang@amss.ac.cn

配,称为复杂本体匹配[7]。现有复杂本体匹配方法主要有基于模式的方法和基于机器学习的方法。

基于模式方法的基础是匹配模式[8-10]。匹配模式刻画本体间常见的匹配形式,例如匹配模式 Class By Property Value Correspondence[10]代表源本体中类与目标本体中具有特定属性值的实例构成的类语义等价。领域专家可以通过这些模式手工构建实体间的复杂匹配。近一步,利用这些模式可以自动地发现复杂匹配[7,11-12]。在文献[7]中,如果一个实体的名称满足某种模式,那么一个相应的复杂匹配就可以生成。文献[11]在此基础上加入语言学技术和背景知识。这些方法可以发现有意义的复杂匹配,但它们需要人工制定启发式规则,匹配的结果依赖于启发式规则的好坏。文献[12]利用本体中实例获得约束类,并使用析取和合取组合不同的约束类来获得复杂匹配,其中约束类是指由某个属性为特定值的实例构成的类。尽管文献[12]不需要启发式规则,但该方法仅组合了约束类形式的匹配,并没有覆盖其他常见的匹配模式。

基于机器学习的方法是解决复杂本体匹配的另一种选择。文献[13]探讨了使用归纳逻辑程序(Inductive Logic Programming,ILP)方法来学习复杂匹配的可行性。文献[14]把复杂匹配问题转化成一个规则学习问题:利用实例匹配生成训练数据并使用扩展的一阶归纳学习器(First Order Inductive Learner,FOIL)[15]来学习得到复杂匹配。实验表明该方法可以自动地学习出一些有意义的复杂匹配。然而,将该方法移植到 LOD 数据集 YAGO 和 DBpedia 的匹配时,发现该方法学习到的部分匹配覆盖的正例较少。这主要是由于 FOIL 方法是一个“短视算法”(lack of lookahead),每次贪婪地加入一个文字,容易陷入局部最优解[15-16]。

由于基于模式的方法和基于机器学习的方法具有各自的局限性,本文提出一种融合这两者的复杂匹配方法。该方法通过实例信息来学习复杂匹配,假设源本体中的实体具有多方面性质,每一个性质刻画的信息可以表示成一个匹配模式的具体化,使用路径特征来表示匹配模式的具体化。

1 形式化

一个本体在描述逻辑中被定义为 TBox 和 ABox[17]。其中,TBox 由类定义(class definition)和属性定义(property definition)构成。类也称概念,是一组实例的集合,通常对应一个抽象概念。属性也称关系,表示实例之间的关系。ABox 由实例声明(instance assertion)构成,实例声明包括类声明和属性声明。下文使用符号 C 和 R 分别代表本体中类和属性。实例类型声明 $C(a)$ 表示实例 a 是类 C 的实例,属性声明 $R(a,b)$ 表示实例 a 与实例 b 有 R 关系。由于方法通过实例的类信息和属性信息来学习复杂匹配,为方便方法叙述,将本体中实例信息直观地表示成一个带标记的有向图。

定义 1(实例图) 实例图由一个三元组 $<N,E,L>$ 构成,其中,N 是节点的集合;E 是边的集合;L 是标签的集合。节点表示本体中声明的实例。边表示两个实例间的关系。节点 a 具有类标签 $\{C|C(a)\}$。从节点 a 到节点 b 的边具有属性标签 $\{R|R(a,b)\}$。

图 1 展现了 DBpedia 中实例“Tian_Zhuangzhuang”和“Chen_Kaige”局部信息的实例图。如节点“Tian_Zhuangzhuang”代表一个实例,该节点的标记{Person}表示“Tian_Zhuangzhuang”是类 Person 的一个实例,从节点“Tian_Zhuangzhuang”到节点“Film_director”的边标记{occupation}表示“Tian_Zhuangzhuang”和“Film_diector”有 occupation 关系。

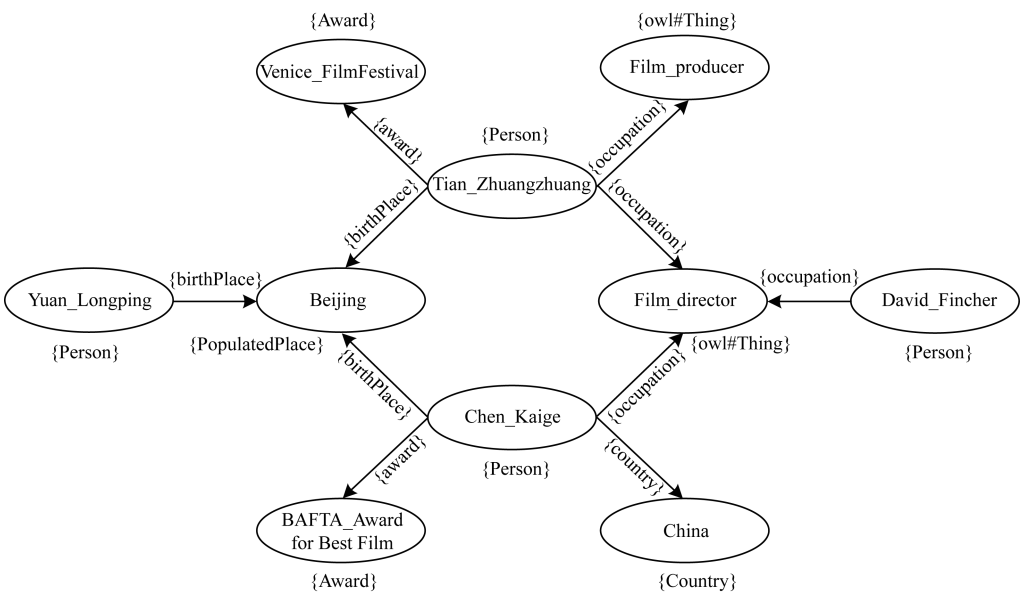

图 1 从 DBpedia 中构建的局部实例

在构造实例图过程中,首先使用推理机将 ABox 进行扩充。比如使用推理机得到 TBox 中的类层次结构,并使用这些层次结构丰富 ABox 中实例类声明。区别于本体 RDF 图,实例图仅刻画了实例的类信息以及实例间的关系。而本体 RDF 图可以表达本体更多信息,比如类定义、类的层次关系等。显然实例图可以由本体 RDF 图直接转化。

首先给出一个具体的例子,然后引出本文方法中的本体匹配的形式化定义。已知 YAGO 中类 Film_directors_from_Beijing,试图在 DBpedia 中找到与该类语义等价的复杂匹配。假设图 1 所示 DBpedia 中的实例“Tian_Zhuangzhuang”和“Chen_Kaige”语义上属于 YAGO 中 Film_directors_from_Beijing 类,可以使用这些实例的性质来学习其匹配。

从“Tian_Zhuangzhuang”出发,能够得到一些不同长度的路径,这些路径记录了节点和边及其对应标记。为了叙述简洁,列举其中 4 条长度为 2 的带标记路径:

path1:Person(Tian_Zhuangzhuang),
occupation(Tian_Zhuangzhuang,Film_director)

path2:Person(Tian_Zhuangzhuang),
birthplace(Tian_Zhuangzhuang,Beijing)

path3:Person(Tian_Zhuangzhuang),
award(Tian_Zhuangzhuang,Venice_Film_Festival)

path4:Person(Tian_Zhuangzhuang),
occupation(Tian_Zhuangzhuang,Film_producer)

path1 ~ path4 表示实例“Tian_Zhuangzhuang”具备的特征:职业是电影导演;出生地是北京;获得过维纳斯电影节奖;另一个职业是电影制片人。如果把实例“Tian_Zhuangzhuang”替换成变量 x,那么得到带标记的路径 path1 ~ path4 泛化后的结果,有下面的形式:

f_1:Person(x),occupation(x,Film_director)

f_2:Person(x),birthplace(x,Beijing)

f_3:Person(x),award(x,Venice_Film_Festival)

f_4:Person(x),occupation(x,Film_producer)

其中,f_1 ~f_4 分别描述了实例 x 具有的性质。如 f_1 表示 x 是一个人并且 x 的职业是电影导演。因为 f_1 ~ f_4 来源于从“Tian_Zhuangzhuang”出发的带标记的路径的抽象表示,所以把它们称为实例“Tian_Zhuangzhuang”的路径特征。执行类似的操作,得到实例”Chen_Kaige”的路径特征:

f_1:Person(x),occupation(x,Film_director)

f_2:Person(x),birthplace(x,Beijing)

f_5:Person(x),award(x,BAFTA_Award for Best Film)

f_6:Person(x),country(x,China)

通过观察可得,实例“Tian_Zhuangzhuang”和实例“Chen_Kaige”拥有共同的路径特征 f_1 和 f_2。回顾在概述中提到的内容,类 Filma_directors_from_Beijing 刻画的一类人具有 2 个性质:这类人不仅是电影导演,而且出生于北京。路径特征 f_1 和 f_2 分别表达了这两方面的信息。更进一步,路径特征可以统一为一个以文字合取表达的模式:

$$C_0(x_0),R_1(x_0,x_1),C_1(x_1),\cdots,R_n(x_{n-1},x_n),C_n(x_n) \tag{1}$$

其中,一元谓词 $C_i(i=0,1,\cdots,n)$ 和二元谓词 $R_i(1,2,\cdots,n)$ 分别对应本体中的类和属性。每个路径特征是将式(1)中谓词具体化为本体中特定类或属性。需要注意的是,最后一个谓词 C_n 也可以是一个枚举类。例如,令 C_0 = Person, R_1 = occupation, C_1 = {Tian_Zhuangzhuang},那么可以得到路径特征 f_1。

式(1)覆盖了本体匹配问题中常见的匹配模式,这些模式在文献[10]进行了详细的描述,例如:

1)Class By Property Value Correspondence:

$C(x_0)\leftarrow R_0(x_0,a)$

2)Class Correspondence by Path Attribute Value:

$C(x_0)\leftarrow R_0(x_0,x_1),R_1(x_1,x_2),\cdots,R_n(x_n,a)$

3)Class by Attribute Type Correspondence:

$C(x_0)\leftarrow R_0(x_0,x_1),\ C_0(x_1)$

从以上这个具体的例子中得到启发,引入关于路径特征和复杂匹配的形式化定义。

定义 2(路径特征)　将从实例 a 出发的带标记的路径中实例替换为变量得到式(1)的形式称为实例 a 的路径特征。

定义 3(复杂匹配)　给定源本体 O_s 中的一个类 C,如果 C 具备 n 个方面的性质,那么 C 的复杂匹配可以表示成一个 Horn 规则:

$$C\leftarrow f_1,f_2,\cdots,f_n \tag{2}$$

其中,f_i 是由目标本体 O_t 中实体构成的一个路径特征,具备式(1)的形式,表示 C 的第 i 个性质。

2　方法设计

基于路径特征的复杂匹配的实现可以分成3 步,如算法 1 所示。

1)实例匹配和实例图构造:生成源本体 O_s 和目标本体 O_t 间的实例匹配,根据目标本体构造实例图(算法 1 的 1 行、2 行)。

2)训练数据构造和路径特征提取:对源本体 O_s 中的每一个类,利用实例匹配结果获得训练数据,并且利用目标本体 O_t 的实例图提取出训练数据集中实例的路径特征(算法 1的 4 行 ~7 行)。

3)规则获取:选择能代表类的各个性质的路径特征,基于这些路径特征产生复杂匹配的规则(算法 1的 8 行)。

算法 1　复杂匹配

输入　source ontology O_s;target ontology O_t

输出　a set of correspondences M for classes in O_s

1:construct instance matching I between O_s and O_t

2:construct instance graph G_t

3:for class C ∈ O_s do
4: create training dataset:
positive dataset D_C^t and negative dataset $D_{\bar{C}}^t$
5: for instancea ∈ $D_C^t \cup D_{\bar{C}}^t$ do
6: create path features f^a from G_t
7: end for
8: rule set $M_C \leftarrow$AcquireRules(C,D_C^t,$D_{\bar{C}}^t$)//算法 2
9: add M_C to M
10:end for

2.1 实例匹配和图构造

本文使用实例信息来学习源本体 O_s 中类 C 在目标本体 O_t 中的复杂匹配,因此,需要知道目标本体 O_t 中的哪些实例在语义上属于类 C。这些知识可以使用实例匹配来获取。实例匹配(也就是在实例层面匹配不同的本体)起到了连接不同本体的作用。详细的描述和方法可参考文献[18-19]。特别地,LOD 数据集包含了大量表示实例匹配的 same-as 声明,这些链接关系可以被直接使用。用 $\mathcal{I}$ 表示实例匹配的集合:

$$\mathcal{I}=\{(a,b)\mid a\in\Delta_s,b\in\Delta_t,match(a,b)\} \quad (3)$$

其中,$match(a,b)$ 表示实例 a 和实例 b 匹配;Δ_s 和 Δ_t 分别指源本体 O_s 和目标本体 O_t 的全体实例集合。

此外,为了获得训练数据集和抽取路径特征,需要针对目标本体 O_t 构造定义 1 中描述的实例图 $\mathcal{G}_t$。具体来说,首先使用推理机来扩充 ABox,然后为每一个实例构造一个节点,节点的标签为对应实例所属的类,节点间边的标签为对应实例间的属性关系。

2.2 训练数据构造和路径特征抽取

给定源本体 O_s 的一个类 C,首先得到类 C 的实例集:$D_C^s=\{a\mid C(a),a\in\Delta_s\}$,其中,数据集中的实例声明或者是本体中已经显式定义的,或者是通过推理机获得的。再根据实例集 D_C^s 和实例匹配集 $\mathcal{I}$,在目标本体 O_t 中得到语义对应的实例集合。对于 C,分别构造正例集 D_C^t:

$$D_C^t=\{b\mid a\in D_C^s,b\in\Delta_t,(a,b)\in\mathcal{I}\} \quad (4)$$

和负例集 $D_{\bar{C}}^t$:

$$D_{\bar{C}}^t=\{b\mid a\in\Delta_s\backslash D_C^s,b\in\Delta_t,(a,b)\in\mathcal{I}\} \quad (5)$$

然后为 D_C^t 和 $D_{\bar{C}}^t$ 中的每一个实例构造路径特征。具体来说,在目标本体 O_t 的实例图 $\mathcal{G}_t$ 中,从数据集中的某个实例 a 出发遍历所有带标记的路径,通过将实例替换为变量后得到关于 a 的一组路径特征 $f^a=\{f_1^a,f_2^a,\cdots,f_n^a\}$。学习复杂匹配规则的任务就被转化成识别特定的一个路径特征或者多个路径特征的组合来区分 D_C^t 和 $D_{\bar{C}}^t$。

2.3 规则获取

类似于一阶归纳学习器 FOIL,使用从一般到特殊的策略来组合路径特征。与 FOIL 不同,当学习匹配规则时,不以本体中单个实体为规则中的基本单位,而是以路径特征为基本单位。

算法 2 规则获取
输入 class C;positive dataset D_C^t;negative dataset $D_{\bar{C}}^t$
输出 correspondence rule set M_C for C
1:$M_C\leftarrow\varnothing$
2:while $|D_C^t|>$min_instance do
3: selected path feature set $S_f\leftarrow\varnothing$
4: positivedataset P←D_C^t,negative dataset N←$D_{\bar{C}}^t$
5: while termination condition is not satisfied do
6: candidate path features $S^c\leftarrow\varnothing$
7: for instance a ∈ P do
8: add f^a to S^c
9: end for
10: for path feature f ∈ S^c do
11: compute FoilGain of f
12: end for
13: find feature f_{opt} from S^c with maximum FoilGain
14: add f_{opt} to S_f
15: update P,N based on f_{opt}
16:end while
17:get rule r from S_f and add r to M_C
18:remove all the instances that satisfy r from D_C^t
19:end while

算法 2 的 3 行~17 行描述了如何通过组合路径特征学习匹配规则。首先目标路径特征集合 S_f 初始化为空集,正负训练数据集 P 和 N 分别初始化为 D_C^t 和 $D_{\bar{C}}^t$。再按照下文介绍的准则逐个选择当前最优路径特征,并把选择的特征 f_{opt} 添加到特征集合 S_f,循环该过程直至满足一定的终止条件。在内层循环中(6 行~15 行),首先构造类 C 的候选路径特征集合 S^c:如果 P 中实例 a 具备路径特征 f,那么将 f 加入到集合 S^c 中。再计算集合 S^c 中的每个路径特征信息增益值 *FoilGain*(式(6)),并选择 *FoilGain* 值最大的路径特征加入到集合 S_f 中,然后,通过移除那些不满足当前路径特征的实例来更新训练数据集。在内层循环满足终止条件后,选择出来的路径特征集合 S_f 构成一个形如定义 3 的匹配规则 r。其中,规则头是待匹配的类,规则体是 S_f 中路径特征的合取。

FoilGain 值可以度量当前路径特征集合 S_f 添加路径特征时所增加的信息量。假设 S_f 是当前选择的路径特征集合,$|P|$和$|N|$分别表示数据集中满足 S_f 的正例和负例个数。如果添加一个新的路径特征 f,路径特征集合变为 S_f',使得满足 S_f'的正例个数和负例个数分别变为$|P'|$和$|N'|$。则添加路径特征 f 获得的信息增益是:

$$FoilGain(f)=|P'|\times\left(\mathrm{lb}\frac{|P'|}{|P'|+|N'|}-\mathrm{lb}\frac{|P|}{|P|+|N|}\right) \quad (6)$$

关于内层循环的终止条件(5 行):每次试图添加一个新的特征之前,计算添加特征后的集合 S_f 的 F 值。F 值是信息检索中常用的度量:

$$F\text{-}measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{7}$$

$$Precision = \frac{|P|}{|P| + |N|} \tag{8}$$

$$Recall = \frac{|P|}{|D_C^t|} \tag{9}$$

如果新的特征集合能够获得更高的 F 值,那么这个新特征可以被接受并且算法进入下一轮循环来寻找下一个特征。否则,该特征被舍弃并且内层循环终止,此时已经挑选出的路径特征组成的集合可以构成一条匹配规则(如式(2)所示)。

内层循环获得一个匹配规则 r 之后,删去所有被 r 覆盖的正实例,进而迭代地学习新的匹配规则直到正例个数小于预先设定的阈值 *min_instance* (2 行)。设定一个阈值的原因是太少的实例难以得到可靠的规则。经过外层的循环迭代,一个类可能会找到多个匹配规则,其中每个匹配规则刻画了类的一部分内容。

2.4　时间复杂度

本文方法的时间复杂度主要来源于 2 个方面:抽取实例的路径特征和组合这些路径特征。

事实上,在具体的实现过程中,对于每个实例的路径特征只需要抽取一次,所以,抽取实例的路径特征的时间复杂度是 $O(n\bar{d}^L\bar{l}^{(2\times L+1)})$,其中,$n$ 是训练数据集中实例的个数;$\bar{d}$ 是构造的实例图中节点的平均出度;L 是路径特征的最大长度;$\bar{l}$ 是节点和边平均标记个数。

组合路径特征的时间复杂度是 $O(mnk)$[20],其中,m 是源本体中的待匹配的类的个数;k 是正数据集中路径特征的个数。这个复杂度与 m,n 和 k 呈线性关系。k 可以在不损失学习能力的情况下大幅度减小,即通过特征筛选方法去除那些覆盖较少正实例的路径特征。

在复杂匹配中,一个小的 L 足以表达大多数的匹配。$\bar{l}$ 与目标本体的 TBox 结构有关,通常也很小。因此,算法的时间复杂度主要取决于 $\bar{d}$,它反映了实例图的结构。从理论分析上来讲,本文方法非常适合出度相对较小的本体。

3　实验结果与分析

用 Java 实现了提出的方法,并设计实验对其进行了评估。实验环境为 Core Duo 3.00 GHz CPU 和 8 GB RAM 的计算机。使用 LOD 数据集中 YAGO 和 DBpedia 作为实验数据。

3.1　数据集

YAGO 和 DBpedia 拥有数百万的实例,它们均属于数据规模较大、覆盖范围较广的本体。YAGO 是一个主要从 Wikipedia, WordNet 和 GeoNames 得来的巨大的语义知识库。DBpedia 同样也是一个巨大的、多领域的知识库,它从 Wikipedia 中抽取了大量结构化信息,包含了 100 多种的语言。尽管 YAGO 和 DBpedia 在实例层面有大量的链接关系,但是它们的类和属性的构建相对独立。在实验中,主要关注 2 个本体的英语部分。同时由于方法论述中仅考虑了对象属性,因此去掉了 2 个本体中的数值属性,但方法本身可以扩充以支持数值属性。这样产生的数据集相关统计如表 1 所示。通过初步观察和对比 2 个本体,发现 YAGO 的类定义相对于 DBpedia 更为细致,所以,选择在 DBpedia 中学习能够语义对应 YAGO 中类的复杂匹配。

表 1　YAGO 和 DBpedia 的相关统计

分类	YAGO	DBpedia
类个数	483 312	1 165
对象属性个数	37	1 334
实例个数	3 696 617	4 542 476
类声明个数	16 928 118	29 263 124
对象属性声明个数	5 602 427	14 533 363

3.2　从 YAGO 到 DBpedia 的类映射

在实验中,移除实例个数少于 10 的类,因为这部分类提供的有效信息过少。通过这样的筛选,YAGO 剩下 131 332 个类。用式(7)评价每一个类学习到的匹配结果。图 2 记录了 F 值大于不同阈值时的匹配结果的个数。比如,当 F 值大于 0.5 时,大约有 30 000 个类可以找到匹配。

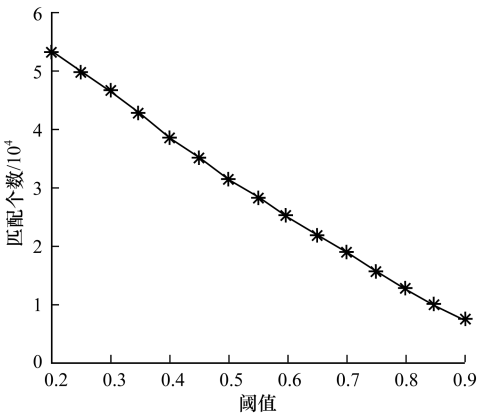

图 2　F 值高于特定阈值的匹配个数

进一步,将方法得到的匹配结果分为 4 类:1)得到的匹配语义等价于待匹配的类;2)得到的匹配包含待匹配的类;3)得到的匹配包含于待匹配的类;4)其他。

从质量上评估匹配结果以验证方法的有效性。由于 YABO 和 DBpedia 规模庞大且目前没有关于两者复杂匹配的黄金标准,采取人工评估的方法,即随机抽取 500 个 F 值大于 0.5 的匹配,并请课题组同学统计不同 F 值阈值下语义等价匹配占所有匹配的比率。图 3 展示了统计结果:在这 500 个匹配中,F 值高于 0.5 的匹配中语义等价匹配约占 75% ,F 值高于 0.9 的匹配中语义等价匹配约占 98% 。

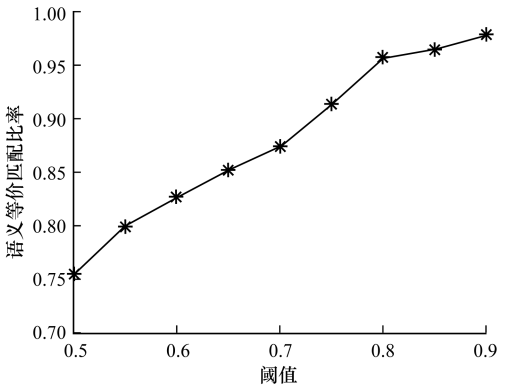

图 3 F 值高于特定阈值的语义等价匹配比率

表 2 列举了一些由本文方法得到语义等价匹配:前 3 个由一个路径特征组成,后 3 个由多个路径特征组成。每一个复杂匹配都由 Horn 规则表示。这些学习得到的匹配都能够等价地表达待匹配的类。如类 Video_game_companies_of_Sweden 刻画了一类公司,这类公司属于视频游戏产业,地址位于瑞典,学习得到的复杂匹配是:

$$\text{Video\_game\_companies\_of\_Sweden}(x_0) \leftarrow \text{Company}(x_0), \text{industry}(x_0, \text{Video\_game\_industry}), \text{locationCountry}(x_0, \text{Sweden})$$

表 2 语义等价匹配的例子

YAGO 中的类	DBpedia 中的匹配
Coldplay_albums	$\text{Album}(x_0), \text{artist}(x_0, \text{Coldplay})$
Films_directed_by_Woody_Allen	$\text{Film}(x_0), \text{director}(x_0, \text{Woody\_Allen})$
Radio_Disney_stations	$\text{RadioStation}(x_0), \text{broadcastNetwork}(x_0, \text{Radio\_Disney})$
Video_game_companies_of_Sweden	$\text{Company}(x_0), \text{locationCountry}(x_0, \text{Sweden}), \text{industry}(x_0, \text{Video\_game\_industry})$
Private_schools_in_Victoria_(Australia)	$\text{School}(x_0), \text{state}(x_0, \text{Victoria\_(Australia)}), \text{type}(x_0, \text{Independent\_school})$
Architecture_firms_based_in_New_York_City	$\text{Company}(x_0), \text{industry}(x_0, \text{Architecture}), \text{foundationPlace}(x_0, \text{New\_York})$

YAGO 中的有些类没有在 DBpedia 中找到语义等价的匹配,而是找到了后 3 种类型的匹配结果。表 3 根据这 3 种类型分别列举了一些例子。如关于 YAGO 中的类 Olympic_bascketball_players_of_Greece,学习得到的复杂匹配是:

$$\text{Olympic\_bascketball\_players\_of\_Greece}(x_0) \leftarrow \text{BasketballPlayer}(x_0), \text{birthPlace}(x_0, x_1), \text{PopulatedPlace}(x_1), \text{country}(x_1, \text{Greece})$$

这个结果属于第 2 种情况。虽然这样的匹配不是语义等价的,但是也具备重要的意义,因为相较于 1:1 形式的匹配,给出了一个更加精细化的匹配。如简单本体匹配系统 PARIS[21] 系统针对这个类学习到的匹配结果是:

$$\text{Olympic\_basketball\_players\_of\_Greece}(x_0) \subseteq \text{BasketballPlayer}$$

与本文方法的结果相比,该匹配只找到了篮球运动员这一信息。同样地,第 3 种类型的匹配也是有意义的,因为这样的匹配可以刻画源本体中类的部分信息。

表 3 非语义等价匹配的例子

类型	YAGO 中的类	DBpedia 中的匹配
1	Olympic_basketball_players_of_Greece	$\text{BasketballPlayer}(x_0), \text{birthPlace}(x_0, x_1), \text{PopulatedPlace}(x_1), \text{country}(x_1, \text{Greece})$
2	Towns_and_villages_in_Latvia	$\text{Village}(x_0), \text{country}(x_0, \text{Latvia})$
3	Chechen_wars	$\text{MilitaryConflict}(x_0), \text{commander}(x_0, x_1), \text{President}(x_1), \text{deathPlace}(x_1, \text{Chechnya})$

导致非语义等价匹配的原因包括:1)本体中记录的数据可能是不完整的。如每一个人都有出生地,但是本体中有些类型为人的实例 birthplace 属性并没有被声明。2)YAGO 和 DBpedia 可能在定义实例的时候存在一定的偏差。如在 DBpedia 中,88% 的 Cook 岛人的后代被定义为运动员。

3.3 与其他方法的比较

基于模式的复杂匹配方法需要人工制定启发式规则,其结果依赖于启发式规则的质量和数量。规则的制定需要对本体有全面认识,这通常需要大量的人工参与,尤其是处理 YAGO 和 DBpedia 这样的大规模本体。因此,以例子来对比本文方法与基于模式的方法。在 YAGO 中有一些表示某个运动队 X 的队员的类,这些类名称具有“X_players”的形式,如类 Apollon_Limassol_FC_players 表示足球队 Apollon_Limassol_FC 的运动员。针对该形式定义启发式规则:如果类的名称满足形式“X_players”,那么匹配为 $\text{X\_players}(x_0) \leftarrow \text{Athlete}(x_0), \text{team}(x_0, \text{X})$。根据这条规则能够得到很多语义等价的匹配。然而“X_players”还可能表示很多其他含义,如类 American_ice_hockey_players 表示美国冰上曲棍球运动员,使用上述规则会得到错误的匹配。此外,基于模式的方法只能为符合启发式规则的类构造匹配,即使使用大量的启发式规则也很难覆盖所有的类。与之相比,本文方法不需要人工制定启发式规则,并且能够自动发现类名称的歧义现

象,如“X_players”形式的类的多重含义。

下面比较本文方法与基于 FOIL 的方法。尽管本文方法与 FOIL 相比都使用贪婪搜索的方式构造规则,但是两者具有不同的粒度。在 FOIL 中,组成规则的基本单位是文字。然而,本文方法中组成规则的基本单位是路径特征,每一个路径特征是若干文字的合取。因此,本文方法与 FOIL 相比在构造匹配规则时粒度更大。通过使用路径特征作为基本单位,本文的方法约束规则为一个特定的形式,使得本文方法的解空间和 FOIL 方法的解空间相比被压缩为一个更具有结构化的一部分。

这些约束导致方法无法学习一些较强表达能力的规则,如规则 $C(x_0)\leftarrow R_1(x_0,x_1),R_2(x_0,x_2),R_3(x_1,x_2)$。但是由于匹配模式覆盖了常见的复杂本体匹配形式,而方法对匹配模式进行了扩充,因此方法表达了常见的复杂匹配形式。同时,更大的粒度以及更小的解空间使得本文方法能够缓解 FOIL 的局部解问题。

为了比较 2 种方法,进行了与基于 FOIL 的复杂匹配方法的对比实验。由于局部最优匹配与全局最优匹配相比往往覆盖较少的正例。一个合理的比较方式是对于同一个待匹配的类,比较 2 个方法得到的匹配召回率(式(9))。为保证规则的正确性,筛选出 2 种方法得到匹配的准确率(式(8))均大于 0.9的类,这样获得 11 189 个类。其中,有 5 584 个类通过本文方法得到更高的召回率,有 341 个类通过 FOIL 方法得到更高的召回率,其余两者得到的匹配召回率相等。

通过统计两者学习到的高召回率(假定召回率大于 0.5 为高召回率)的匹配个数来间接反映两者的优劣。具体来说,对于 2 个方法学习到的匹配,先去除准确率低于 0.9 的部分从而保证正确性,再分别统计 2 个方法大于不同召回率阈值的匹配个数。图 4 表明,本文方法得到的高召回率匹配基本上是基于 FOIL 方法的 2 倍,即间接证明了本文方法能够有效缓解 FOIL 方法的局部最优问题。

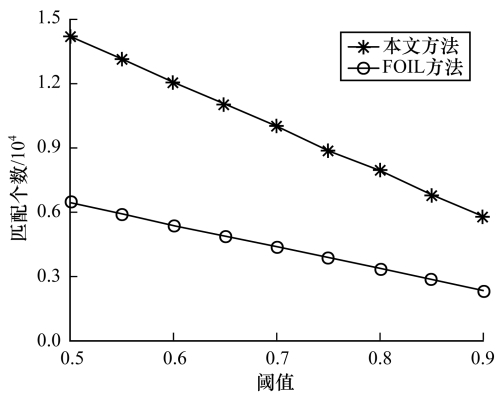

图 4　2 种方法不同召回率阈值下的匹配个数

下面通过一个直观的例子体现局部最优带来的不利影响。YAGO 中的类 BK_Forward_players 表示 BK_Forward(一个瑞典足球队)的运动员。本文方法得到的结果是:

$$\text{BK\_Forward\_players}(x_0)\leftarrow \text{SoccerPlayer}(x_0),\text{team}(x_0,\text{BK\_forward})$$

然而基于 FOIL 的方法得到的结果是:

$$\text{BK\_Forward\_players}(x_0)\leftarrow \text{Athlete}(x_0),\text{careerStation}(x_0,x_1),\text{position}(x_0,\text{Defender\_association\_football}),\text{team}(x_1,\text{BK\_Forward})$$

这个匹配只刻画了 BK_Forward 的防守队员。

4　结束语

本文研究本体之间的异构问题,提出一种基于路径特征的复杂本体匹配方法。该方法结合匹配模式和 FOIL 两者的优点,从本体的实例出发,抽取相应的路径特征,进而通过学习选择和组合路径特征,获得复杂匹配结果。实验结果证明了方法的有效性。但是,本文方法独立地学习每个类的复杂匹配,没有考虑不同类之间的影响。此外,没有处理本体中的不完整数据,这会使得结果出现偏差。因此,下一步的工作重点是实现复杂匹配的联合学习以及处理不完整数据。

参考文献

[1] Bizer C, Heath T, Berners-Lee T. Linked Data——The Story So Far [J]. Semantic Services, International Journal on Semantic Web and Information Systems, 2009, 5(3): 1-22.

[2] Auer S, Bizer C, Kobilarov G, et al. Dbpedia: A Nucleus for a Web of Open Data [C]//Proceedings of the 6th International Semantic Web Conference. Busan, Korea: SWSA, 2007: 722-735.

[3] Suchanek F M, Kasneci G, Weikum G. Yago: A Core of Semantic Knowledge [C]//Proceedings of the 16th International Conference on World Wide Web. New York, USA: ACM Press, 2007: 697-706.

[4] Stuckenschmidt H, van Harmelen F. Information Sharing on the Semantic Web[M]. Berlin, Germany: Springer, 2005.

[5] Euzenat J, Shvaiko P. Ontology Matching [M]. Berlin, Germany: Springer, 2013.

[6] 李　荣. 异构本体映射中的关键方法研究[D]. 长春: 吉林大学, 2014.

[7] Ritze D, Meilicke C, Sváb-Zamazal O, et al. A Pattern-based Ontology Matching Approach for Detecting Complex Correspondences [C]//Proceedings of ISWC Workshop on Ontology Matching. Chantilly, USA: SWSA, 2009: 25-36.

[8] Scharffe F, Fensel D. Correspondence Patterns for Ontology Alignment[C]//Proceedings of the 16th International Conference on Knowledge Engineering: Practice and Patterns. Acitrezza, Italy: [s. n.], 2008: 83-92.

(下转第 240 页)

参考文献

[1] 车万翔,刘 挺,李 生.实体关系自动抽取[J].中文信息学报,2005,19(2):1-6.

[2] 董 静,孙 乐,冯元勇,等.中文实体关系抽取中的特征选择研究[J].中文信息学报,2007,21(4):80-85.

[3] 黄 鑫,朱巧明,钱龙华,等.基于特征组合的中文实体关系抽取[J].微电子学与计算机,2010,27(4):198-200.

[4] Wang Mingyin, Li Lei, Huang Fang. Semi-supervised Chinese Open Entity Relation Extraction[C]//Proceedings of the 3rd International Conference on Cloud Computing and Intelligence Systems. Washington D. C., USA:IEEE Press,2014:415-420.

[5] Zhu Xiaojin, Ghahramani Z. Learning from Labeled and Unlabeled Data with Label Propagation: CMU-CALD-02-107[R]. Pittsburgh, USA: Carnegie Mellon University,2002.

[6] Chen Jinxiu, Ji Donghong, Tan Chew-lim, et al. Relation Extraction Using Label Propagation Based Semi-supervised Learning[C]//Proceedings of the 21st International Conference on Computational Linguistics. Berlin, Germany: Springer,2006:129-136.

[7] 潘 云,布 勒,布丽汗,等.利用中文在线资源的远程监督人物关系抽取[J].小型微型计算机系统,2015,36(4):701-706.

[8] 张 雁,吴保国,吕丹桔,等.基于 Tri-training 的主动学习算法[J].计算机工程,2014,40(6):215-218,229.

[9] 姚从磊,邸 楠.一种基于 Web 的大规模人物社会关系提取方法[J].模式识别与人工智能,2007,20(6):740-744.

[10] 毛小丽,何中市,邢欣来,等.基于语义角色的实体关系抽取[J].计算机工程,2011,37(17):143-145.

[11] 刘 路,李弼程,张先飞.基于正反例训练的 SVM 命名实体关系抽取[J].计算机应用,2008,28(6):1444-1446.

[12] 李 晶.基于网络抱团发现的命名实体关系抽取[D].武汉:华中师范大学,2006.

[13] 吴伟宁,刘 扬,郭茂祖,等.基于采样策略的主动学习算法研究进展[J].计算机研究与发展,2015,49(6):1162-1173.

[14] 陈耀东,王 挺,陈火旺.半监督学习和主动学习相结合的浅层语义分析[J].中文信息学报,2008,22(2):70-75.

[15] 刘慧婷,熊瑞瑞,赵 鹏.基于信任关系重建和社交网络传递的推荐算法[J].计算机工程,2016,42(1):174-179.

[16] Wang Fei, Zhang Changshui. Label Propagation Through Linear Neighborhoods[J]. IEEE Transactions on Knowledge and Data Engineering,2008,20(1):55-67.

[17] Ren Yong, Kaji N, Yoshinaga N, et al. Sentiment Classification in Resource-scarce Languages by Using Label Propagation[C]//Proceedings of the 25th Pacific Asia Conference on Language, Information and Computation. Singapore:[s. n.],2011:420-429.

编辑 顾逸斐

(上接第 233 页)

[9] Scharffe F, Zamazal O, Fensel D. Ontology Alignment Design Patterns[J]. Knowledge and Information Systems, 2014,40(1):1-28.

[10] Scharffe F. Correspondence Patterns Representation[D]. Innsbruck, Austria: University of Innsbruck,2009.

[11] Ritze D, Völker J, Meilicke C, et al. Linguistic Analysis for Complex Ontology Matching[C]//Proceedings of the 5th International Conference on Ontology Matching. Aachen, Germany: CEUR-WS. org,2010:1-12.

[12] Parundekar R, Knoblock C A, Ambite J L. Discovering Alignments in Ontologies of Linked Data[C]//Proceedings of the 23rd International Joint Conference on Artificial Intelligence. San Jose, USA: AAAI Press, 2013:3032-3036.

[13] Stuckenschmidt H, Predoiu L, Meilicke C. Learning Complex Ontology Alignments a Challenge for ILP Research[C]//Proceedings of the 18th International Conference on Inductive Logic Programming. Prague, Czech Republic:[s. n.],2008:105-110.

[14] Hu Wei, Chen Jianfeng, Zhang Hang, et al. Learning Complex Mappings Between Ontologies[C]//Proceedings of Joint International Conference on the Semantic Web. Berlin, Germany: Springer,2011:350-357.

[15] Quinlan J R. Learning Logical Definitions from Relations[J]. Machine Learning,1990,5(3):239-266.

[16] Richards B L. Learning Relations by Pathfinding[C]//Proceedings of the 10th National Conference on Artificial Intelligence. San Jose, USA: AAAI Press, 1992:50-55.

[17] Baader F. The Description Logic Handbook: Theory, Implementation, and Applications[M]. Cambridge, UK: Cambridge University Press,2003.

[18] Castano S, Ferrara A, Montanelli S, et al. Ontology and Instance Matching[J]. Knowledge-driven Multimedia Information Extraction and Ontology Evolution, 2011, 6050:167-195.

[19] 宋亚楠,仲 茜,钟 远.基于多属性的本体实例匹配算法[J].计算机工程,2011,37(13):63-65.

[20] Yin Xiaoxin, Han Jiawei. CPAR: Classification Based on Predictive Association Rules[C]//Proceedings of 2003 SIAM International Conference on Data Mining. San Francisco, USA: SIAM,2003:331-335.

[21] Suchanek F M, Abiteboul S, Senellart P. Paris: Probabilistic Alignment of Relations, Instances, and Schema[J]. Proceedings of the VLDB Endowment,2011,5(3):157-168.

编辑 刘 冰