### Tutorial: 基于开放知识库的QA系统gAnswer

1. 系统简介

gAnswer系统[1]是一个基于海量知识库的自然语言问答系统，针对用户的自然语言问题，能够输出SPARQL格式的知识库查询表达式以及查询答案的结果。整个项目用JAVA编写，因此可以在不同操作系统上使用。

gAnswer同时支持中文问答和英文问答。对于中文问答，我们使用PKUBASE知识库；对于英文问答，使用DBpedia知识库。<http://ganswer.gstore-pku.com/> 是英文问答的实例网站。若需要获取源码及相关数据，可以访问<http://www.openkg.cn/tool/ganswer>

1. 系统优点

**端到端**：目前绝大多数同类开源项目只提供简单的算法模型，其功能只包括将自然语言问题转化为某种逻辑表达式（即只进行语义解析）。与之相比，gAnswer系统提供一套包含数据处理、索引构建、语义解析、查询执行的完整代码。经过封装处理，用户只需要非常简单的部署操作（参见4部署说明），即可通过http请求进行交互，返回给定问题的答案。

**支持复杂问题**：目前绝大多数问答系统只能回答简单问题或两跳问题（包含两个谓词）。这些系统所生成的查询结构是预先定义的，它们无法处理更复杂的问题。gAnswer使用节点优先的查询图构建方法，理论上其所能产生的查询图结构是不受限的，即可以处理包含三个以上谓词的更复杂问题。

**大规模，高效率**：现有一些问答系统完全不使用知识库信息，只能在特定小规模数据集上回答问题。另外一些系统将语义解析和知识库割裂开来，先生成若干候选SPARQL查询后再使用引擎访问知识库进行查询，将各候选的查询结果作为特征进行排序。这样做极大的降低了系统的效率。gAnswer将语义解析和知识库结合起来，借助知识库索引使用数据驱动的消歧技术，避免了大量的知识库访问查询时间，极大的提高了整体问答效率。

**多语言，多知识库**：与现有只针对单一语言问答的系统相比，gAnswer提供多个版本来支持中文问答和英文问答。目前分别提供在中文知识库PKUBASE和英文知识库DBpedia上预处理好的数据索引、预训练好的关系抽取模型或谓词复述词典、预生成的实体链接词典或第三方实体链接工具。

**易扩展**：gAnswer非常便于进行迁移和扩展。若不需要更换知识库，只希望针对特定应用场景进行问答性能的适应性提升，则可以通过直接修改补充谓词复述词典、实体链接词典的方式来进行扩展。若希望更换知识库，则可以通过提供的数据预处理代码，对新的知识库构建索引，并更换适用于新知识库的实体链接器和谓词复述词典或关系抽取模型即可。

1. 系统技术架构

图1展示了gAnswer系统的整体处理流程。主要分为三个阶段：语义查询图构建，SPARQL生成和答案查询。



图 1 gAnswer系统架构

* 在查询图生成阶段，系统借助数据集的信息以及自然语言分析工具，对问句进行实体识别和关系抽取，构建语法依存树，并用这些结果来构建对应的查询图。这时，并不对其中的实体和关系做消歧处理，而是利用谓词词典，记录词或短语可能对应的谓词或实体。图2展示了这一处理过程。
* 在SPARQL生成阶段，系统利用查询图生成多个SPARQL，并利用数据集中的部分信息对多个SPARQL进行过滤和优化，其中就包括歧义的消除。
* 在答案生成阶段，借助gStore系统返回的SPARQL查询结果，返回并展示给用户。



图 2 gAnswer系统流程

在部署执行gAnswer系统前，需要注意：

1. **系统配置要求：**
* gAnswer系统需要使用RDF格式的数据集，目前使用的中文数据集是PKUBASE，英文数据集是dbpedia2016，并对其进行了一定的筛选和预处理。
* 外部图数据库系统。gAnswer 系统的运行需要借助支持 SPARQL 查询的图数据库系统来获取最终答案。在目前的版本中，使用 gStore 系统，并且在其基础上构建相应数据库，并使用 http 请求与其交互。关于 gStore 系统，详情请参阅其 github 页面 <https://github.com/pkumod/gStore>
* 外部工具包。包括maltparser、StanfordNLP，在sparql生成阶段，需要借助Lucene对辅助信息进行索引。
* 其他，见下表。其中磁盘容量和内存空间由不同知识库规模决定。鉴于全量知识库数据对内存的需求较大，我们也提供小规模知识库子集便于用户测试。

|  |  |
| --- | --- |
| 项目 | 要求 |
| 操作系统 | Linux，Windows |
| 架构 | x86\_64 |
| 磁盘容量 | >8GB (for DBpedia) |
| 内存空间 | >20GB (for DBpedia) |
| Java | 版本 >= 1.6 |

1. **外部依赖：**
* gAnswer系统运行需要引入一系列jar包以及外部文件。而gAnswer的开发者提供了一个打包好的版本，只需要下载jar包和文件压缩包，并解压到对应的位置即可。在第三部分部署说明中将阐述如何使用开发者提供的打包文件。
* 除此之外，在目前的 gAnswer 系统需要借助一些外部系统接口。在公开的版本中，提供了远程的外部系统调用函数，因此，用户并不需要在自己的计算机上安装这些外部系统。但是，开发者强烈建议用户安装自己的版本，以保证性能。具体见下表：

|  |  |  |
| --- | --- | --- |
| 项目 | 要求 | 位置 |
| gStore | 版本 >= v0.7.0 | qa.GAnswer.getAnswerFromGStore2() |
| DBpediaLookup |  | qa.mapping.DBpediaLookup |

如果需要安装gStore，请从gStore的github主页上获取资源与相关信息：https://github.com/pkumod/gStore

如果需要安装DBpediaLookup，请从DBpedia的官方网站上获取资源与相关信息：<https://wiki.dbpedia.org/lookup/>

1. 部署使用
* 用户可以使用打包好的jar文件部署gAnswer，具体步骤为：
* 下载Ganswer.jar与data.rar两个文件，开发者建议用户从github的release页面下载最新版的Ganswer.jar与data.rar，以保证稳定性。
* 在控制台下解压Ganswer.jar，用户可以解压到任意文件路径下，但需要保证Ganswer.jar文件与解压得到的文件处在统一路径下。
* 在控制台下解压data.rar，用户需要把解压得到的文件夹置于Ganswer.jar文件所在的路径下。
这时，对应的文件结构应该如下所示：

 ./

 ++ addition

 ++ application

 ++ data

 ++ fgmt

 ++ jgsc

 ++ lcn

 ++ lib

 ++ log

 ++ META-INF

 ++ nlp

 ++ paradict

 ++ qa

 ++ rdf

 ++ utils

 ++ Ganswer.jar

* 在控制台下运行Ganswer.jar，等待系统初始化结束，出现Server Ready！字样后，则说明初始化成功，您可以开始通过Http请求访问gAnswer的服务了。
1. **http API：**

gAnswer的http服务通过发送json格式的数据进行交互。例如：

{

“maxAnswerNum”: “3”

“needSparql”: “1”

“question”: “闻一多创作了哪些十四行诗？”

}

上述json数据的含义为：回答“闻一多创作了哪些十四行诗？”这个问题，要求最多返回3个不同的答案，1条生成的SPARQL查询。将此json数据转化为字符串，进行url转码，然后使用ip:port/gSolve/?data=%json string%（在%json string%处放入json数据字符串）这一uri来调用gAnswer系统。本地运行ip为localhost，默认端口为9999。在样例中，实际访问的uri为：

http://localhost:9999/gSolve/?data={ maxAnswerNum:3, needSparql:1, question:闻一多创作了哪些十四行诗？}

对于上例，返回的结果是：

{

 "query":"闻一多创作了哪些十四行诗？",

 "vars":[

 "?x"

 ],

 "sparql":[

 "select ?x where { ?x <作者> <闻一多> . ?x <文学体裁> <十四行诗> . }"

 ],

 "results":{

 "bindings":[

 {

 "?x":{

 "type":"uri",

 "value":"<回来\_（闻一多创作现代诗）>"

 }

 }

 ]

 },

 "status":"200"

}

需要特别说明的是，其中“vars”代表识别到的变量名，“results”中为实际得到的答案，“value”中为实际答案的值，“status”则说明这是一次正常的请求返回。

1. **java API：**

我们在源代码的application.gAnswerHttpConnector中给出了使用Java通过http请求访问gAnswer系统的示例。

1. **知识库迁移：**

使gAnswer系统在新的知识库上运行，需要更新查询引擎、离线索引和词典。下面具体说明。

1. 将新的知识库组织成三元组形式，如下例：

kb.txt：

<库木库萨尔乡> <电话区号> "998" .

<库木库萨尔乡> <地理位置> <新疆维吾尔自治区喀什地区> .

<库木库萨尔乡> <面积> "3.9万亩" .

<库木库萨尔乡> <人口> "11200" .

<库木库萨尔乡> <车牌代码> "新Q" .

<正清风痛宁缓释片> <药品名称> "正清风痛宁缓释片" .

<正清风痛宁缓释片> <药品类型> <处方药> .

<狐狸的梦想> <书名> "狐狸的梦想" .

<狐狸的梦想> <又名> "The fox's dream" .

<狐狸的梦想> <原版名称> "狐狸的梦想" .

<狐狸的梦想> <类别> <寓言故事> .

<ask insight> <中文名> "上海因尚企业管理咨询有限公司" .

<ask insight> <外文名> "ASK insight" .

<ask insight> <总部> <上海\_（中华人民共和国直辖市）> .

……

将kb.txt文件置于data/kb/中。

通过gStore查询引擎建立基于kb.txt的查询服务，详情参见<http://gstore-pku.com/> 。

1. 根据kb.txt生成实体、谓词、类型的列表文件，如下例：

entity\_id.txt：

<库木库萨尔乡> 0

<正清风痛宁缓释片> 1

<狐狸的梦想> 2

<ask insight> 3

……

predicate\_id.txt：

<电话区号> 0

<地理位置> 1

<面积> 2

<人口> 3

<车牌代码> 4

……

type\_id.txt：

<寓言故事> 0

<教材> 1

<国家> 2

<城市> 3

……

将上述三个文件置于data/kb/fragments/id\_mappings中。

运行src/fgmt/GenerateFragment.java，程序将产生三个编码后的碎片文件entity\_fragment.txt，predicate\_fragment.txt，type\_fragment.txt并置于data/kb/fragments/中。以entity\_fragment.txt为例，格式为eid \t in ent : in edge | out ent : out edge | inEdge list | outEdge list | types，样例如下：

……

12 |229347:1639;2093;,||1639,2093,|102967,116252,

13 |2511063:2107;,||5899,2107,|102967,

14 ||||102967,116252,

15 |861779:2107;,||2107,|102967,

……

为提高效率，我们使用lucene建立索引。运行src/lcn/BuildIndexForEntityFragments.java和src/lcn/ BuildIndexForTypeShortName.java。程序会在data/kb/lucene下生成索引文件。

1. 提供新知识库的实体链接词典和谓词复述词典，示例如下：

mention2ent.txt

逆时针 逆时针\_（汉语名词） 1

逆时针 逆时针\_（张靓颖演唱歌曲） 2

逆时针 逆时针\_（化妆品品牌） 3

张君宝 张君宝\_（起点中文网作者） 1

张君宝 张三丰\_（南宋至明初道士） 2

……

pred2phrase.txt

电话区号 区号 30

地理位置 位置 50

地理位置 在哪里 30

面积 多大 20

……

其中逆时针 逆时针\_（汉语名词） 2是指短语“逆时针”可以链接到实体“逆时针\_（张靓颖演唱歌曲）”，第三列数字2表示这个链接是短语“逆时针”的所有链接中置信度第二高的；地理位置 在哪里表示短语“在哪里”可以匹配到谓词“地理位置”，数字30为该次匹配的置信度。mention2ent和pred2phrase两个文件主要用来支撑实体链接和关系抽取两个子模块。将这两个词典文件置于data/kb/parapharse中。

以上操作完成后，gAnswer即可在新的知识库上提供问答服务。

1. 性能评估

QALD(Question Answering over Linked Data)是由欧盟组织的一系列开放领域问答比赛，从2011年开始至今已举办九届。gAnswer参加了QALD-9 (<https://project-hobbit.eu/challenges/qald-9-challenge/>) 的评测比赛，并取得了第一名的成绩。



表1 QALD-9比赛结果

1. 优化

目前gAnswer系统还存在许多改进空间。例如对于过于复杂的问题，会花费大量时间进行计算，一个可行的方案是进行动态阈值设置以便系统提前结束计算。另外，还可以通过替换更好的实体识别/关系抽取、提供更多的训练数据来提升相应性能。我们会不断对gAnswer系统进行更新维护和优化。

1. 相关论文
2. Hu S , Zou L , Yu J X , et al. Answering Natural Language Questions by Subgraph Matching over Knowledge Graphs[J]. IEEE Transactions on Knowledge and Data Engineering, 2018, 30(5):824-837.