浙江大学

本科生毕业设计
开题报告

学生姓名：___周自强_____

学生学号：3120101943

指导教师：陈华钧

年级与专业：大四 计算机科学与技术

所在学院：计算机学院

一、题目：基于大规模知识图谱的规则挖掘系统的实现

二、指导教师对开题报告、外文翻译和中期报告的具体要求：

指导教师（签名）_____

年　　月　　日

毕业设计开题报告、外文翻译和中期报告考核

导师对开题报告、外文翻译和中期报告评语及成绩评定：

成绩比例	开题报告 占（20%）	外文翻译 占（10%）	中期报告 占（10%）
分 值			

导师签名_____

年　　月　　日

答辩小组对开题报告、外文翻译和中期报告评语及成绩评定：

成绩比例	开题报告 占（20%）	外文翻译 占（10%）	中期报告 占（10%）
分 值			

开题报告答辩小组负责人（签名）_____

年　　月　　日

-

目录

本科毕业设计开题报告

1. 项目背景

最近几年，许多知识库例如 Cyc、YAGO、DBpedia、Freebase 兴起。大规模知识库在各种应用中例如自然语言问答、语义搜索引擎等等都有作用。这些知识库由数百万的世界实体以及它们之间的关系所组成，以有向图的形式存储，结点表示实体，连接表示实体间的关系。尽管这样的知识库包含成千上万的实体，它仍然是稀疏的，即它在实体之间缺失了大量的关系。在大规模知识库中存储的实体（entity），包括了人、国家、河流、城市、大学、电影、动物等等信息；还有许多事实（fact），例如：伦敦是英国的首都、每一个歌手都是人类等等。从知识库中可以知道，谁出生在哪里，一个演员演了哪几部电影，或者是一个城市在哪个国家。现今的知识库包括了成千上万的实体（entity），以及更多的事实（fact）。

然而知识库并不是完整的。这时候，我们需要从已有的知识库中进行规则挖掘，进而找寻两个实体之间的关系，使得知识图谱密集化。例如一条规则：livesIn(h, p) ^ marriedTo(h, w) => livesIn(w, p)，它表示如果两个人是夫妻关系，那么他们（通常）住在同一个城市。

挖掘出这样的规则，我们能够利用它做什么？首先，我们可以进行知识库的补全，例如，我们知道奥巴马住在哪里，并且知道他的妻子是米歇尔，那么，我们通过挖掘出的规则，就可以知道米歇尔住在哪里；第二，这些规则可以找出知识库中一些潜藏的错误信息，例如知识库中说奥巴马的妻子米歇尔居住在中国，我们就能够判断这条信息很有可能是错误的；第三，规则可用于推理。最后一点，这些规则可以使我们更好地理解整个知识库，我们可以发现使用相同语言国家之间贸易往来频繁、婚姻是一种对称的关系等等。

2. 目标和任务

目标：实现一个规则挖掘的系统，分析大规模知识图谱，进行规则挖掘，通过挖掘出的规则进行知识库的补全。

具体分为以下几个部分：

1. 以 AMIE 为基础，分析大规模知识图谱，在知识图谱中实现封闭规则挖掘。
2. 在 AMIE 的基础上编写代码，进一步完善规则挖掘算法的细节，并使用提供的知识库数据进行挖掘测试。
3. 提供对中文知识图谱规则挖掘的支持。
4. 由于 AMIE 对于封闭霍恩规则的挖掘已经有了很好的支持，我们的工作就是在 AMIE 的基础上进行拓展。在封闭规则挖掘的基础上，进一步进行非封闭的规则挖掘。

3. 可行性分析

大规模知识库的规则挖掘这一课题有一定难度，但基本可行，原因主要基于以下几个方面：

1. 在大规模知识图谱中，存在成千上万的事实（fact），从这些已有事实中我们可以找到一些规则，用来描述这些事实。而通过挖掘出来的规则，我们可以进一步完善知识库。因此，对知识库进行相应的规则的挖掘是可行的。
2. 关联规则是反映一个食物与其他事物之间的相互依存和关联性，是数据挖掘中的一个重要技术。现在有许多关于关联规则挖掘的算法，例如 AMIE 在不完整知识库中的关联规则挖掘，或者使用桥接实体以及随机游走算法来进行规则的挖掘。并且 AMIE 已有相应的代码来实现关联规则挖掘的部分功能。由此可以看出，在知识库中进行关联规则的挖掘并不是不可能的。
3. 规则挖掘的算法越来越成熟，实际应用也越来越广泛，越来越多的人在研究学习规则挖掘，并提供了许多高质量的开源代码，这对于我们实现关联规则挖掘有很大的帮助。
4. 网络上提供了许多知识库数据例如 YAGO、DBpedia 的数据集用于测试，这对于我们实现并调试规则挖掘算法提供了便利。

4. 初步技术方案和关键技术考虑

初步技术方案：

AMIE 是一个数据规则挖掘系统，它能够实现从一个知识库中提取出逻辑规则的置信度等信息，从而进行规则的挖掘。我们在 AMIE 的基础上进行规则挖掘系统的开发。

1. 首先我们要了解一些相关概念：

RDF 知识库：

知识库中的一个 fact 可以使用一个三元组来表示，形式<x, r, y>，其中 x 为主语（subject），r 为关系（relation or predicate），y 为宾语（object）。这里我们表示成 r(x, y)。

函数（function）：

在关系 r 中，对于每一个 subject，最多只有一个 object 阈值对应，这样的关系称为函数（function）。另外，我们使用标记 functionality，关系 r 的 functionality 是一个 0 到 1 的值，当 r 是一个 function 的时候值为 1。

$$\mathrm{fun}(\mathrm{r}) \coloneqq \frac{\#x: \exists y: r(x,y)}{\#(x,y): r(x,y)}$$

规则（rule）：

形式如$\mathrm{B1} \wedge \mathrm{B2} \wedge \ldots \wedge \mathrm{Bn} \Rightarrow \mathrm{r(x,y)}$，缩写为$\vec{B} \Rightarrow \mathrm{r(x,y)}$

其中 B1、B2…为 atom，r(x,y)为 head，$\mathrm{B1} \wedge \mathrm{B2} \wedge \ldots \wedge \mathrm{Bn}$为 body。

Support：

对于一个规则，需要在知识库中有对应的 fact 来支持。Support 表示在 head 中不同的 subject 与 object 对的数量。

$$\mathrm{supp}(\vec{B} \Rightarrow \mathrm{r(x,y)}) \coloneqq \#(\mathrm{x,y}): \exists z1, \ldots, zm: \vec{B} \wedge \mathrm{r(x,y)}$$

其中 z1,…zm 是除了 x、y 之外的变量。

Head Coverage：

由于 support 是一个绝对数字，在这里我们使用 head coverage 来表示一个相对值：

$$\mathrm{hc}\left(\vec{B} \Rightarrow \mathrm{r(x,y)}\right) \coloneqq \frac{\mathrm{supp}(\vec{B} \Rightarrow \mathrm{r(x,y)})}{size(r)}$$

其中 size(r):=#(x',y') : r(x',y')，表示关系 r 对应的 fact 的数量。

标准置信度（standard confidence）：

$$\mathrm{conf}\left(\vec{B} \Rightarrow \mathrm{r(x,y)}\right) \coloneqq \frac{\mathrm{supp}(\vec{B} \Rightarrow \mathrm{r(x,y)})}{\#(\mathrm{x,y}): \exists z1, \ldots, zm: \vec{B}}$$

PCA 置信度（partial completeness assumption confidence）

$$\mathrm{conf}_{pca}\left(\vec{B} \Rightarrow \mathrm{r(x,y)}\right) := \frac{\mathrm{supp}(\vec{B} \Rightarrow \mathrm{r(x,y)})}{\#(\mathrm{x,y}): \exists z1, \ldots, zm, y': \vec{B} \wedge \mathrm{r(x,y')}}$$

2. AMIE 规则挖掘算法

输入知识库 KB，阈值 minHC，规则的最大长度 maxLen，最小置信度 minConf。输出规则。

首先获取一个规则队列，初始包含所有的 head atom，且 size 为 1。

然后每次从队列中获取一个 rule，如果这个 rule 满足一定条件，则添加到输出队列；如果该 rule 的长度不超过最大长度 maxLen，则对该 rule 进行进一步完善，并将完善后的满足条件的 rule 添加到规则队列中。

伪代码：

```
function AMIE(KB K, minHC, maxLen, minConf)
    q = [r1(x, y), r2(x, y)…rm(x, y)]
    out = < >
    while ¬ q.isEmpty() do
        r = q.dequeue()
        if AcceptedForOutput(r, out, minConf) then
            out.add(r)
        end if
        if length(r) < maxLen then
            R(r) = Refine(r)
            for all rules rc∈R(r) do
                if hc(rc) ≥ minHC & rc ∉ q then
                    q.enqueue(rc)
                end if
            end for
        end if
    end while
    return out
end function
```

3. 进一步细化，对队列中的规则进一步处理，直到满足相应的输出条件：

对规则的细化过程主要包含下面三个 atom 的添加：

a. Dangling atom

往 rule 中添加一个新的 atom，该 atom 使用了一个新的变量，另一个变量为规则中的一个已有变量。

b. Instantiated atom

往 rule 中添加一个新的 atom，该 atom 一个变量为一个实体（entity），另一个变量为规则中的一个已有变量。

c. Closing atom

往 rule 中添加一个新的 atom，该 atom 使用的变量为规则中的已有变量。

4. 算法的优化，包括算法效率的优化，对中文的支持。
5. 增加非封闭规则挖掘的算法，挖掘出更多规则。
6. 获取已有的一些知识库，例如 Yago、DBpedia 的知识库，进行这些知识库进行规则挖掘，进行统计分析。

关键技术：

1. 设计方法查找满足条件的 relation，用以下形式表示（Projection Query）：

   SELECT r COUNT(H)

   WHERE H∧B1∧…∧Bn-1∧r(X,Y)

   SUCH THAT COUNT(H) ≥ k

   由于这些 query 在算法中会大量出现，因此对它的优化是一个很关键的问题。
2. 设计方法实现 Dangling atom、Instantiated atom、Closing atom 三种 atom 的查找。这是 closed Horn rule 挖掘的关键步骤，实现好了这几个方面，可以说基本上完成了算法的大部分工作。
3. 多线程速度优化。
4. 队列规则细化过程的优化，例如通过设置阈值限制规则的最大长度，或者在找到 Perfect rule（PCA confidence≥100%）时停止添加 atom，优化 Projection Query 等。
5. AMIE 不支持对于非封闭霍恩规则挖掘，因此要实现它，需要进一步查询文献，获取一些理论支持，寻找一些较好的算法。

5. 预期工作结果

预期工作结果：在 AMIE 的基础上实现一个规则挖掘的系统，分析大规模知识图

谱，进行规则挖掘，通过挖掘出的规则进行知识库的补全。

具体细节如下：

1. 以 AMIE 为基础，分析大规模知识图谱，在知识图谱中实现封闭规则挖掘。
2. 在 AMIE 的基础上编写代码，进一步完善规则挖掘算法的细节，并使用提供的知识库数据进行挖掘测试。
3. 提供对中文知识图谱规则挖掘的支持。
4. 由于 AMIE 对于封闭霍恩规则的挖掘已经有了很好的支持，我们的工作就是在 AMIE 的基础上进行拓展。在封闭规则挖掘的基础上，进一步进行非封闭的规则挖掘。

6. 进度计划

3 月 1 日~3 月 15 日：熟悉 AMIE，了解 closed Horn 规则挖掘算法。

3 月 16 日~3 月 31 日：熟悉 AMIE 代码，阅读相关论文，了解 AMIE 的整体架构与算法实现。

4 月 1 日~4 月 7 日：获取一些知识库，使用 AMIE 进行知识库中规则的挖掘，将结果进行统计对比。

4 月 8 日~3 月 15 日：在 AMIE 的基础上进行系统开发，实现 closed Horn 规则挖掘。

4 月 16 日~4 月 23 日：编写代码，进一步完善规则挖掘算法的细节，并使用提供的知识库数据进行挖掘测试。

4 月 24 日~4 月 30 日：优化算法，使用多种方法提高算法效率，进行规则挖掘，并进行结果分析与对比。

5 月 1 日~5 月 7 日：在实现基本的规则挖掘功能的基础上，提供系统对中文知识图谱规则挖掘的支持。

5 月 8 日~5 月 12 日：学习非封闭规则挖掘的相关知识，获取理论基础。

5 月 13 日~5 月 23 日：在实现 closed Horn 规则挖掘的基础上，进一步进行非封闭的规则挖掘，并进行数据测试与比较。

5 月 24 日~5 月 28 日：完善一些细节，修复各类 bug，完成毕业设计。

本科毕业设计外文翻译