上海交通大学硕士学位论文

# 基于可满足性问题求解的辅助诊断内核的设计与实现


**硕士研究生**：丁如江

**学　　号**：117037910020

**导　　师**：李国强


**申请学位**：工学硕士

**学　　科**：软件工程


**所在单位**：电子信息与电气工程学院



**答辩日期**：2019 年 11 月 30 日

**授予学位单位**：上海交通大学

Dissertation Submitted to Shanghai Jiao Tong University
for the Degree of Master

# DESIGN AND IMPLEMENTATION OF COMPUTER AIDED DIAGNOSIS CORE BASED ON SATISFIABILITY PROBLEM SOLVING


| | |
|---|---|
| **Candidate:** | Rujiang Ding |
| **Student ID:** | 117037910020 |
| **Supervisor:** | Prof. Guoqiang Li |
| **Academic Degree Applied for:** | Master of Engineering |
| **Speciality:** | Software Engineering |
| **Affiliation:** | School of Electronic Information and Electronic |
| **Date of Defence:** | Nov. 30th, 2019 |
| **Degree-Conferring-Institution:** | Shanghai Jiao Tong University |

# 上海交通大学
# 学位论文原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：某某

日　期：某年某月某日

# 上海交通大学
# 学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权上海交通大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

**保　密** □，在 _____ 年解密后适用本授权书。

本学位论文属于

**不保密** □。

（请在以上方框内打“✓”）

学位论文作者签名：　　指导教师签名：

日期：　　年　　月　　日　　日期：　　年　　月　　日

# 基于可满足性问题求解的辅助诊断内核的设计与实现

## 摘　要

“健康中国”是当前的国家战略，也是解决民生痛点、满足老百姓健康需求的重要战略。由于国内公共医疗管理系统的不完善，医疗成本高、渠道少、覆盖面低等问题困扰着大众民生。尤其以“效率较低的医疗体系、质量欠佳的医疗服务、看病难且贵的就医现状”为代表的医疗问题为社会所诟病。当前国内的医疗问题主要分为以下三类：首先是医疗资源的匮乏，医疗资源供不应求，医护人员严重不足，很多人根本无法享受到完整的医疗服务。其次是医疗资源分配不均匀，尤其是优质医疗资源分配不均。好医生，好药方，好设备全部集中在发达城市的三甲医院，人民对乡村以及社区医院不信任，导致大医院人满为患，社区医院无人问津。最后是医疗系统的碎片化，不同医院之间信息格式不统一，医疗资源不共享，导致医疗服务，医疗支付之间数据没有打通，存在“数据孤岛”现象。

针对上述问题，“智慧医疗”概念应运而生，其旨在优化医院的诊断流程，提高医院操作的信息化以及自动化程度，使患者用较短的等疗时间、支付基本的医疗费用，就可以享受安全、便利、优质的诊疗服务。智慧医疗应用场景包括医疗系统的信息化管理、医学的影像诊断、医疗的辅助决策、疾病的风险预测、药物的挖掘等。本文主要针对医疗中的辅助诊断，即可以根据患者的主述以及简单的人机交互可以给出对疾病的诊断。对于不算严重的疾病，比如感冒发烧，我们的辅助诊断系统可以直接给出对疾病的诊断。患者就无需经历到医院进行挂号，诊疗，取药的冗长流程，而是可以根据辅助诊断的结果直接到医院开具药品。而对于严重的疾病，辅助诊断系统可以预先获取患者的症状信息，推荐患者做相应的检查，这样医生可以直接根据症状信息以及检查结果给出诊断，这样也简化了患者的诊断流程，减轻了

医生的负担。

现在市面上的智慧医疗产品大多是在做医疗系统的信息化管理，而为数不多的做辅助诊断的产品采用的解决方案基本都是大数据方案。即从大量的电子病历中提取相互关联的疾病和症状，然后采用深度学习的方法学习出一个疾病和症状的关联模型，这样就可以从患者的症状信息中推断出患者的疾病。这种方案的优势在于数据量大，几乎可以包含所有的常见疾病及症状。但是其问题也很明显，首先就是严重依赖病历，而病历中一般都是常见疾病，对于罕见疾病病历可能无法覆盖。其次就是结果的可解释性，大数据方案旨在找寻数据之间的关联性，缺乏因果关系。比如根据“低热，盗汗，咯血”给出诊断结果是“肺结核”，但是为什么给出这样的诊断缺乏医学知识的支持，可能增强患者的不信任感。最后是数据的可靠性问题，病历中的数据可能存在错误，如何甄别这些错误是大数据方案无法解决的。本文就是为了解决这些问题，提出了新的辅助诊断的解决方案。主要方法是：首先从可信的医学资源中提取医学知识，并对医学知识进行审核验证保证可靠性。其次将医学知识表达成命题逻辑公式，组合这些逻辑公式构建可满足性问题的实例。然后采用 SMT solver 求解可满足性问题的实例，最后根据问题的解来做出诊断。

本文的主要贡献如下：

(1) 知识库的构建

为了实现本文的辅助诊断系统，我们首先构建了符合我们需求的医学知识图谱。通过预先设计好的知识结构，从可信知识源中抽取符合特定模式的知识，最终表达成知识图谱，供后续的算法使用。

(2) 知识纠错方案

由于知识源的数据可能存在错误，以及知识抽取的过程中也可能引入错误，所以最终获取的知识可能存在错误。因为本文采用的是基于医学知识进行诊断而非大数据方案，知识的错误会很大程度影响最终诊断的结果。所以必须对抽取的知识进行校对以及纠错，本文就从逻辑的角度，提出了纠正知识库中错误的方案。

(3) 疾病诊断算法

基于已经构建好的知识库，本文设计了可以根据知识库以及患者的症状信息进行诊断的算法。

(4) 实际应用的系统

本文的所有方案不仅仅是停留在理论层面，而是实现出了可用的系统，并且已经在合作医院中落地。

**关键词**：智慧医疗, 知识图谱, 知识纠错, 逻辑推理, 可满足性问题, SMT 求解器

# DESIGN AND IMPLEMENTATION OF COMPUTER AIDED DIAGNOSIS CORE BASED ON SATISFIABILITY PROBLEM SOLVING

## ABSTRACT

Shanghai Jiao Tong University (SJTU) is a key university in China. SJTU was founded in 1896. It is one of the oldest universities in China. The University has nurtured large numbers of outstanding figures include JIANG Zemin, DING Guangen, QIAN Xuesen, Wu Wenjun, WANG An, etc.

SJTU has beautiful campuses, Bao Zhaolong Library, Various laboratories. It has been actively involved in international academic exchange programs. It is the center of CERNet in east China region, through computer networks, SJTU has faster and closer connection with the world.

**KEY WORDS:** intelligent healthcare,knowledge graph, knowledge error correction, logic reasoning, satisfiability problem, SMT solver

# 目 录

# 插图索引

# 表格索引

# 算法索引

# 第一章　绪论

本章是绪论部分。在本章节中, 首先介绍研究的背景, 之后探讨了研究的目的以及研究意义, 并对国内外与本文相关的研究现状进行了总结分析。然后详细的说明一下本文的主要贡献, 最后是本文的章节安排。

## 1.1　研究背景及意义

多年以来，医疗一直都是困扰民生的痛点问题之一。由于国内公共医疗管理系统的不完善，医疗成本高、渠道少、覆盖面低等问题困扰着大众民生。尤其以“效率较低的医疗体系、质量欠佳的医疗服务、看病难且贵的就医现状”为代表的医疗问题为社会所诟病。当前国内的医疗问题主要分为三类：

1) 首先是医疗资源的匮乏。根据《2018 年中国卫生统计年鉴》统计数据，截至 2018 年 11 月底，全国医院数量达 3.2 万个，其中公立医院 12072 个，民营医院 20404 个。医院中总床位数为 612 个，每千人卫生执业医师为 2.44 个，其中城市为 3.97 个，农村为 1.68 个。相比与其他发达国家，我国民众享受到的医疗资源非常有限。
2) 其次是医疗资源分配严重不均。在医疗资源本来就相对匮乏的情况下，资源分配不均更加重了民众对国内医疗体系的不满意度。在《2018 年中国卫生统计年鉴》的每项数据中，城乡之间的差异都有 2-5 倍之多。除了城乡差异，城市与城市之间，医院与医院之间也有很大差异。优质的医疗资源几乎全部集中在发达城市的三甲医院，好医生，好药方，好设备都在这些医院当中。这就造成了很多民众对基层医疗卫生机构不信任，不算严重的病也要去三甲医院就诊，导致大医院人满为患，社区医院无人问津，一定程度上造成了医疗资源的浪费。
3) 最后是各地各医院医疗系统的碎片化。不同的医院内部都有一套自己的诊疗数据系统，之间的信息格式不统一，医疗资源不共享,，导致医疗服务，医疗支付之间数据没有打通，即存在“数据孤岛”现象

在这样的医疗现状之下，“智慧医疗”的概念应运而生。其旨在优化医院的诊断流程，提高医院操作的信息化以及自动化程度，使患者用较短的等疗时间、支付基本的医疗费用，就可以享受安全、便利、优质的诊疗服务。智慧医疗应用场景包括医疗系统的信息化管理、医学的影像诊断、医疗的辅助决策、疾病的风险预测、药物的挖掘等。其中医疗系统的信息化管理就可以一定程度上解决“数据孤岛”问题，通过将医院中的数据，诊疗流程信息化，患者可以直接在手机或者其他电子设备上进行挂号，缴费等流程，极大程度缩短了诊疗的时间周期。而医院也可以获取电子数据，比如电子病历，电子检查单等，通过统一数据格式，各医院之间的数据也可以共享。除此之外，辅助诊断也是智慧医疗中非常重要的场景之一。通过患者的主述以及简单的人机交互可以给出对疾病的诊断，这样患者可以直接到药店开具药物或者到对应科室做跟详细的诊断，解决了患者盲目就医的问题，也极大减轻了医院导诊台的压力。本文就是针对辅助诊断的场景，设计并实现了实际可用的辅助诊断系统。对于不算

严重的疾病，比如感冒发烧，本文的系统可以直接给出诊断结果，这样患者可以直接开具药品进行治疗，无须再到大医院排队等待。对于严重的疾病，比如肺结核，本文的系统可以给出初步的诊断，并推荐相应的检查与就诊科室，患者可以做完检查后到对应科室做更详细的诊断，这样就简化了“问诊-检查-复诊-取药”的一般就诊流程，节省了患者的时间，也减轻了医生的负担。

现在市面上的智慧医疗产品大多是在做医疗系统的信息化管理，而为数不多的辅助诊断系统采用的解决方案基本都是大数据 + 深度学习。即从大量的电子病历中提取相互关联的疾病和症状，然后采用深度学习的方法训练出一个疾病和症状的关联模型，这样就可以根据患者的诊断信息推断出疾病。这种方案的优势在于数据量大，几乎可以完全覆盖常见的疾病和症状。但是其问题也很明显，首先是严重依赖电子病历，病历中一般都是常见疾病，对于罕见疾病难以覆盖。其次就是其诊断结果的可解释性，大数据 + 深度学习方案旨在找到数据之间的关联性，而没法获取数据之间的因果联系。比如比如根据“低热，盗汗，咯血”给出诊断结果是“肺结核”，但是为什么给出这样的诊断缺乏医学知识的支持，可能增强患者的不信任感。最后是数据的可靠性问题，我国的电子病历尚处于起步阶段，很多医生不太习惯使用电子病历，不少医生填写病历就像在写“八股文”，这样就造成电子病历中可能存在错误，如何甄别并修改这些错误是大数据方案无法解决的。

综上所述，本文针对大数据 + 深度学习方案所面临的问题，提出了一个新的辅助诊断解决方案：通过从医书等可信医学知识源中提取医学知识，然后将其表达成逻辑公式，通过逻辑推理来进行辅助诊断。具体来说：首先从可信的医学资源中提取医学知识，并对医学知识进行审核及验证保证其可靠性。然后将医学知识表达成命题逻辑公式，组合这些逻辑公式构建可满足性问题的实例。再然后采用 SMT 求解器求解可满足性问题的实例，最后根据问题的解来做出诊断。因为本文的所有数据都是系统的医学知识而非病历这样的经验数据，所以就能保证诊断结果的可解释性。而且因为本文的辅助诊断系统会对提取的知识进行审核并采用一定的算法来进行验证修改，就在一定程度上保证了数据的可靠性。

## 1.2 国内外研究现状

由于本文采用从医书中提取医学知识构建医学知识库，然后根据知识库进行辅助诊断的方案。而医学知识库的构建，辅助诊断其实是两个相对独立的研究领域，目前工业界和学术界在这两个领域都有了一些较为突出的研究成果和产品。本节会分别对医学知识库构建和辅助诊断两个领域的研究现状进行介绍，同时会总结两个领域目前的研究优势以及仍然存在的挑战和可以进一步优化的地方。

### 1.2.1 医学知识库

知识库（Knowledge base）参考文献：《维基百科》是用于知识管理的一种特殊的数据库，以便于有关领域知识的采集、整理以及提取。知识库中的知识源于领域专家，它是求解问题所需领域知识的集合，包括基本事实、规则和其它有关信息。知识库起源于早期的专家系统，专家系统依赖知识库中的事实与规则进行推理问答，而后来人工智能技术的火热也进一步推

动了知识库研究的发展。广义而言，知识库参考文献：《我国医学知识库应用现状研究》就是知识工程的结构化，是易操作、易使用、全面有组织的知识集群，是为了适应某一特定领域的需要，采用特定的知识表达方式在计算机中存储的结构化、相互联系的知识集合。而医学知识库，顾名思义，就是针对医学领域人工智能的需要而构建的知识库。其中的知识多为医学知识，比如药物，病症，诊疗等维度的知识。医学知识库是智慧医疗应用的基石，可以为机器阅读理解医学文本、智能咨询、智能诊断提供知识基础。医学知识库的研究在国内外都较为火热，本节会对国内外的研究成果做详细的介绍。

#### 1.2.1.1 SNOMED Clinical Terms

SNOMED CT参考文献：《维基百科》（Systematized Nomenclature of Medicine – Clinical Terms，医学系统命名法 - 临床术语），是一部经过系统组织编排的，便于计算机处理的医学术语集，涵盖大多数方面的临床信息，如疾病、所见、操作、微生物、药物等。采用该术语集，可以协调一致地在不同的学科、专业和照护地点之间实现对于临床数据的标引、存储、检索和聚合。同时，它还有助于组织病历内容，减少临床照护和科学研究工作中数据采集、编码及使用方式的变异。

SNOMED 起源于 1965 年美国病理学院（CAP）开发的 Systematized Nomenclature of Pathology (SNOP)，1975 年 CAP 将 SNOP 和 SNDO（New York Academy of Medicine's Standard Nomenclature of Diseases and Operations）整合成了 SNOMED，SNOMED 起初只在医学学术领域有一定的反响，在实际医学场景中并没有太多应用。但是在 1997 年，SNOMED RT（SNOMED Reference Terminology）横空出示，并且在 2002 年与英国国家卫生服务部的临床术语（National Health Service Clinical Terms）合并成了 SNOMED CT，使其一举成为世界上最全面，最专业，最权威的医学术语集。这套术语集提供了一套全面统一的医学术语系统，使得全世界各地各医院、各医疗机构之间的信息交换成为可能。例如，对于心脏病学专科医师来说，心脏病发作、心肌梗死以及 MI 可能指的是同一含义，而对于计算机来说，三者之间则全然不同，而通过 SNOMED CT 规定的标准术语，各地医疗机构记录的医疗信息可以使用统一的术语集合和数据组织结构，极大程度方便了不同的医疗机构、研究人员以及其他相关方协调一致地交换临床信息。

SNOMED CT 中的逻辑模型如图 1–1所示，其组成要素为：

- 概念：每个唯一性数字型代码、唯一性名称（全称，即 Fully Specified Name）和描述（包括一条首选术语和一条或多条同义词）所指定的基本含义单位。SNOMED CT 不再使用词条表的方式对术语进行表示，而是采用概念的形式。概念可以理解为医学中标准的临床术语，每个概念都有唯一的概念码，但每一个概念都可能有多个描述。比如“Pain in throat”（咽喉痛），在 SNOMED CT 中是基本概念。
- 描述：赋予同一概念的不同术语或名称（同义词）。比如对于“Pain in throat”（咽喉痛），在实际应用中，它将会有多种不同的术语表达，如“Sore throat”、“Throat pain”、“Pain in pharynx”、“Throat discomfort”、“Pharyngeal pain”、“Throat soreness”，但它们并不是概念，它们只是对于概念”Pain in throat” 的描述或者同义词。
- 关系：用于在同一层级结构之内或不同层级结构之间将不同的概念联系起来。SNOMED

CT 中的概念与概念间是有一定“关系”存在的。概念有 36 万条，但关系有近 146 万条。这种基于概念间的语义关系令数据的获取充分可靠。在 SNOMED CT 中，关系分为两种:

- IS-A 关系，使用在同一个层面中，表示某些概念间的关系。如图 1–2所示，痔疮属于肛管系统疾病，这样痔疮 → 肛管系统疾病就形成了一种 IS-A 关系;
- Attribute 关系，表示跨层面的概念间的关系，如图 1–2所示，“痔疮” 是一种疾病，但从部位来看，” 痔疮 “发生在身体部位” 痔丛 “处，所以” 阑尾炎 “就会有一个” 部位 “属性，属性的值是” 痔丛 “

经过多年的发展，SNOMED CT 目前包括大约 321900 条概念、超过 80 万条临床概念相关的描述，和超过 700 万条进一步描述概念的关系，其已成为世界上最权威，最全面，最专业的多语言医学知识库。SNOMED CT 已经被广泛用于电子病历生成，医学报告，基因数据库等多个领域。但是 SNOMED CT 在中国却没有太大的反响，首先因为是版权问题，中国政府并不是 SNOMED International 的会员，所以大陆机构在使用 SNOMED CT 时需要缴纳高昂的费用。其次是虽然有一些民间组织在做 SNOMED CT 的本地化，但是其质量和速度一直难以保证。所以说现在 SNOMED CT 在中国几乎没有太多实际的应用，国内的医疗科研机构并不会直接使用 SNOMED CT，而更多是参考 SNOMED CT 对知识的组织方式构建自己的医学知识库。

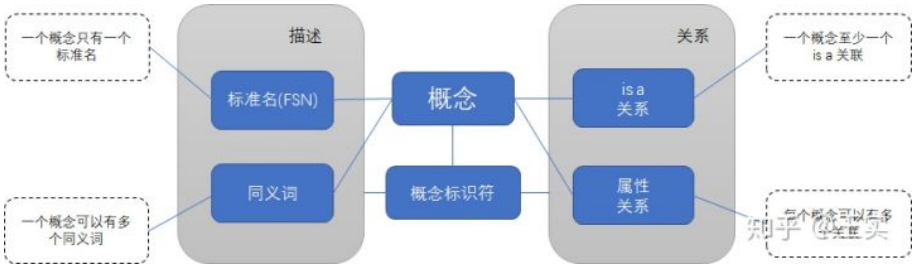



图 1–1 SNOMED CT 逻辑模型

Figure 1–1 The logic model of SNOMED CT

### 1.2.1.2 BMJ Best Practice

BMJ Best Practice 临床实践（简称 BP 是英国医学杂志 (BMJ) 出版集团于 2009 年 2 月出版的循证医学数据库资源的升级版。它在 BMJ Clinical Evidence（临床证据）中的治疗研究证据的基础上，增添了由全球知名学者和临床专家执笔撰写的，以个体疾病为单位，涵盖基础、预防、诊断、治疗和随访等各个关键环节的内容，尤其像鉴别诊断、实验室检查、诊断和治疗的方法和步骤等 (如图 1–3所示)。此外，BP 还提供国际权威指南，收录大量病症的彩色图像和证据表格等资料，嵌入了国际权威的药物处方指南以及患者教育内容。涵盖了 32 个临床专科的 1000 余个疾病（组）和症状主题，其中 80% 以上为临床常见疾病（组）。BP 还收录了 3000 余个诊断分组和 12500 余个细分的诊疗方案，包括 10000 余种诊断方法和 3000 余种诊断性检测；收录了 6000 余篇国际指南，中文版还特别收录了 250 余篇中国指南和中国专家共

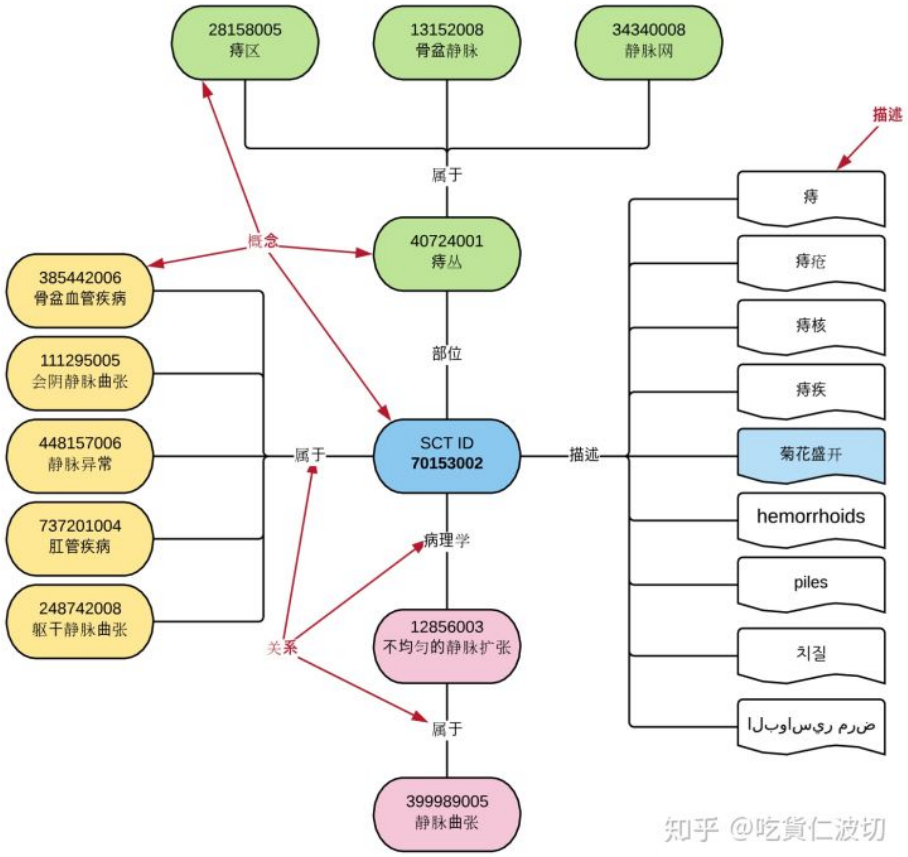



图 1–2 SNOMED CT 中”痔疮“概念的结构关系示例

Figure 1–2 An example of the relationship structure of the concept ”Hemorrhoids” in SNOMED CT

识。另外，BP 也囊括了 700 余个关联的 Cochrane Clinical Answers、4000 余个临床图像、250 余个医学计算器。综上所述，BP 的主要特点有：

a) 收录上千种的临床疾病，包括了临床常见疾病和非常见病。
b) 每一种疾病指南都由世界顶尖临床专家撰写，并经过同行评审完成。增加了权威专家总结的经验和建议。
c) 收录上万种诊断方法。包括临床鉴别诊断、实验室检查、病史检查、诊断步骤和方法等核心内容。
d) 收录了数千项的国际治疗指南和诊断标准的全文内容；并可定制中文的临床指南和标准；此外还提供了大量的彩色病例图片和图像。
e) 提供最新的药物副反应和多种药物相互作用的最新证据。

然而，虽然 BMJ Best Practice 有中文版，但是其翻译质量却较为一般，很多叙述的翻译甚至类似于机器翻译，当然这个问题在不断的改进当中。除此之外，翻译的速度也无法跟上英文原版的更新速度，导致很多条目只有英文版却没有中文版。另外很关键的一点是，BP 中的知识虽然已经分成了”病史和查体”，”诊断步骤“，”检查“等结构化的条目，但是在每个


图 1–3 BMJ Best Practice ”高钠血症”条目截图

Figure 1–3 Screenshot of entry ”Hypernatremia” in BMJ Best Practice

条目内依然是大篇的叙述，这种非结构化的知识无疑增加了自动化处理的难度。

#### 1.2.1.3 CMeKG

CMeKG参考文献：《Preliminary Study on the Construction of Chinese Medical Knowledge Graph》（Chinese Medical Knowledge Graph）是由北京大学计算语言研究所，郑州大学自然语言处理实验室和鹏程实验室在 2019 年合作推出的中文医学知识图谱。CMeKG 利用自然语言处理与文本挖掘技术，基于大规模医学文本数据，以人机结合的方式研发的中文医学知识图谱。CMeKG 的构建参考了 ICD、ATC、SNOMED、MeSH 等权威的国际医学标准以及规模庞大、多源异构的临床指南、行业标准、诊疗规范与医学百科等医学文本信息。CMeKG 1.0 包括：6310 种疾病、19853 种药物（西药、中成药、中草药）、1237 种诊疗技术及设备的结构化知识描述，涵盖疾病的临床症状、发病部位、药物治疗、手术治疗、鉴别诊断、影像学检查、高危因素、传播途径、多发群体、就诊科室等以及药物的成分、适应症、用法用量、有效期、禁忌证等 30 余种常见关系类型，CMeKG 描述的概念关系实例及属性三元组达 100 余万。

如图 1–4所示，CMeKG 中的实体包含“病史”、“发病部位”、“检查”和“临床症状及体征”等多种医学常用关系，但是其采用的 schema 仍然是传统知识图谱扁平的 RDF 结构（主述，谓语，宾语）。这种对知识的表示方式虽然简单，也利于机器的自动学习，但是其在医学这样的专业领域却有一定的局限性，很关键的一点是这种扁平的 schema 无法表达嵌套的结构。比如“甲状腺功能亢进症”的临床表现有“心悸”，“失眠”等，但如果患者是女性，其症状还可能有“月经失调”，所以这条知识就需要嵌套的表达方式，如图所示。而在传统 RDF 表示方式中是很难表达这条知识的。除了表达方式的缺陷，CMeKG 的数据大部分来源于百科数据，其数据质量也难以保证。

当然，除了上述的医学知识库，还有很多通用的知识库中也包含了很多医学的内容。比如从维基百科中撷取出结构化的资料而构建的 DBpedia参考文献，国内从事知识图谱工作的研究者共同开发的开放中文知识图谱 OpenKG参考文献，复旦大学知识工场实验室研发的中文通用百科知识图谱 CN-DBpedia参考文献，罗马大学计算语言学实验室创建的 BabelNet参考文献等等，这些通用的知识库中包含了很多的医学知识，但是其专业性和规模要逊色许多。此外，国内也有一些做从事医疗的企业会构建自己的医学知识库，比如 OMAHA 构建的七巧板医学术语集参考文献，其构建方式参考 SNOMED CT，具有很强的专业性。但是因为不开源且收费高昂，本文没有对其做过多的探索。

### 1.2.2 辅助诊断

诊断是医疗中的一个核心环节，当前医疗给出诊断依赖于：1. 患者体征 2. 患者描述 3. 检查数据，其中 1，3 是主要判断依据。而这两项数据完全可以通过机器自动获取，辅助诊断就是自动获取患者体征以及检查数据，再根据其底层的知识库做自动诊断的过程。辅助诊断如需被医生接受使用，需要满足两个条件：准确以及可解释。而市面上的所有辅助诊断工具都无法满足这两个条件，因此现在所有辅助诊断工具的结果都无法成为患者的确诊结果，而只是作为参考数据提供给医生使用，从而减轻医生的负担。

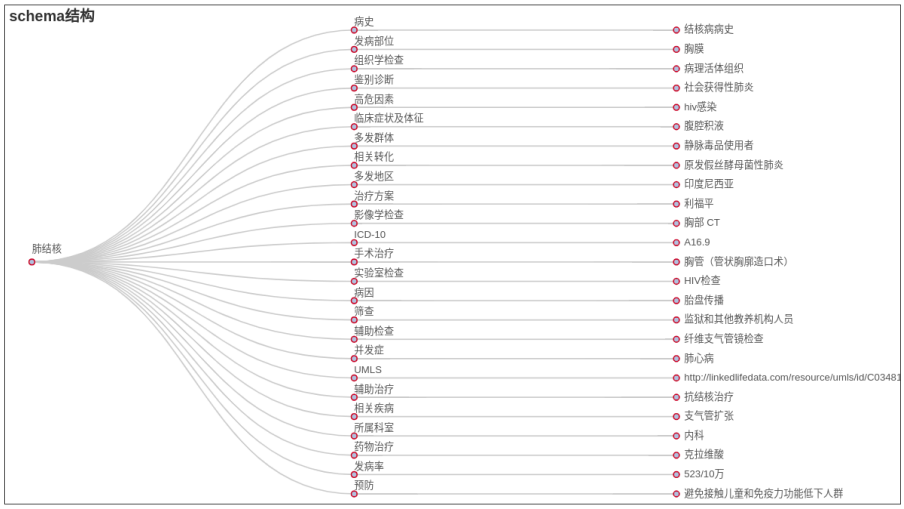



图 1–4 CMeKG schema 示例

Figure 1–4 Example of CMeKG schema

基于底层数据的类型，可以将辅助诊断模型分为：图像诊断类型，文本诊断类型。图像诊断类型的实现逻辑大体是从医院影像资料库获取医疗影像，然后通过机器学习方法训练出诊断模型，继而可以根据已有模型进行诊断。而文本类型的实现逻辑是从医学文献，病例，医书中获取知识构建医学知识库，然后根据知识库中的知识做出诊断。而知识库中的知识如何表示以及如何根据这些知识做出诊断依赖特定的算法，目前学术界和工业界已经尝试了各种方案，本节会对他们做系统的介绍。

#### 1.2.2.1 基于文本匹配

文本匹配是很多智能问答系统采用的技术手段，通过用户的问题从底层文档库中搜索与之匹配度最高的文档，然后根据文档中的实体进行回答。IBM 发布的 Watson 统就是这样的一个智能问答系统，它曾在电视问答比赛节目击败人类玩家从而引发关注。现在 IBM Watson 已经将重心转移到医疗领域，发布了 IBM Watson Health 主要用于肿瘤的诊断及治疗。它从互联网资源（例如 Wikipedia，DBPedia，WordNet 和 Yago参考文献：AI Magazine. “The ai behind watsonthe technical arti- cle”. In: AI Magazine）以及电子病历中收集并结构化医学知识，构建了一个超过 15TB 的庞大知识库。它的诊断策略是首先收集患者的体征和症状信息，然后根据这些患者信息去匹配相似度最高的文档。比如患者的症状是“咳嗽，头晕”，Watson 会使用这两个词条去匹配相似度最高的几篇文档，可能就能匹配到某些医书中的“肺结核”以及“感冒”等章节，然后根据特定的排序策略（比如 TF-IDF，DSSM参考文献：Learning Deep Structured Semantic Models for Web Search using Clickthrough Data等）对结果进行排序筛选，选出相似度最高的文档得出诊断。这种文本匹配方案的优势在于自动化程度高，人类设定好相似度策略以及阈值后就完全可以交给机器自动完成其余的工作，但是其在辅助诊断上的应用缺点也很明显：

- 文本需求量大，从而需要大量计算资源。因为文本匹配只考虑词条在文档中是否出现，而忽视了语义这种包含更多知识量的信息，所以文本匹配算法需要大量的文本来补全这些知识（当然有些缺失掉的知识是无法通过大量文本来补全的）。
- 不适配辅助诊断的流程。辅助诊断需要首先根据患者的主述信息，继而通过与患者的交互获得更多的患者信息，最后根据所有的患者信息进行诊断。然而文本匹配算法需要获得所有的患者信息去匹配最相似的医学文本，而患者很难一下子说出所有的体征信息，这就对文本匹配方法提出了挑战。
- 文本匹配可能获得错误的医学知识。因为文本匹配只考虑词条在文档中是否出现，而不考虑具体的语义信息，这就可能引来一些错误。比如某本医学文档中记录着“高铁血红蛋白血症虽有明显发绀，但一般无呼吸困难”，如果患者的症状信息为“呼吸困难”，文本匹配算法仍然会匹配到“高铁血红蛋白血症”。但其实文本中“高铁血红蛋白血症”是排斥”呼吸困难“的。本质是因为文本匹配丢失了”一般无“这样的语义信息，所以引入了错误。

#### 1.2.2.2 基于 IF-THEN-ELSE 产生式规则

早期的专家系统都是采用于 IF-THEN-ELSE 产生式规则进行推理。比如 1970 年代初期在斯坦福大学开发的辅助诊断工具 Mycin参考文献：Computer-based medical consultations: MYCIN，它主要用于帮助医生对住院的血液感染患者进行诊断和用抗菌素类药物进行治疗。其采用的知识表示方式就是这样的产生式规则，比如“if 化脓菌感染 then 脑膜炎 “，“if 脑膜炎 then 脑损伤”，这样就可以根据“化脓菌感染”得出到“脑损伤”的推理路径。这种知识表示的好处在于推理结果的可解释性，因为所有的知识已经表达成了固定的 IF-THEN-ELSE 规则，所以最终的结果一定可以根据这些规则回溯到原因，因此 Mycin 系统中专门有独立的“Explanation System” 来为诊断结果做出解释。可以看出，如果这些产生式规则足够多且准确，那么这种方案完全可以胜任医生的角色。但是其弊端就在于这些规则严重依赖专家手动编写，Mycin 刚发布时不过 600 余条规则，导致其只能在很小的领域内使用，很难进行扩展。

#### 1.2.2.3 基于贝叶斯网络

贝叶斯网络（Bayesian network）又称信念网络（belief network），是目前不确定知识表达和推理领域最有效的理论模型之一参考文献：维基百科。一个贝叶斯网络是一个有向无环图 (Directed Acyclic Graph,DAG)。结点代表随机变量，有向边代表了结点间的因果关系 (由父结点指向其子结点)，并用条件概率来表达关系强度。贝叶斯网络主要用于在已知发生了某些结果，推断出造成结果发生的原因以及发生的概率。这种特性非常适用于医学诊断，在已知患者症状信息（结果）的情况下，推断出其所患疾病（原因）。比如图一参考文献：画图肺结核，肺气肿。所以现在很多的辅助诊断系统的底层技术都是基于贝叶斯网络，比如参考文献：贝叶斯相关的辅诊工具。但是绝大多数都是用于专科疾病诊断（比如乳腺癌诊断，肺癌诊断等），因为专科疾病诊断用到的贝叶斯网规模较小。如果需要分类的疾病增多（达到几百个），贝叶斯的结构就会变得复杂。有很多边的权重（即疾病与症状之间的条件概率）数据很难获取，在图较小时可以人工赋权，图较大时就很难获得所有的条件概率。其次在网络

规模较大时，基于贝叶斯网的推理也会变得很低效。在最坏情况下，贝叶斯网的学习和推理是 NP-难（NP-hard）的参考文献：The computational complexity of probabilistic inference using bayesian belief networks，Learning Bayesian networks is NP-hard。

#### 1.2.2.4 基于深度学习参考文献：Deep Learning and Medical Diagnosis: A Review of Literature

深度学习（deep learning）是近年来最为火热的研究领域，将深度学习技术应用到智能诊断来非常直观。其具体方案是：首先从大量的电子病历中提取相互关联的疾病和症状，然后将其作为输入训练出一个适配训练集的特定神经网络（CNN参考文献,RBM参考文献等），最后根据神经网络来对疾病做分类预测参考文献:Deep learning for healthcare decision making with EMRs。这种方案的优势在于数据量大，几乎可以完全覆盖常见的疾病和症状。但是其问题也很明显，首先是严重依赖电子病历，病历中一般都是常见疾病，对于罕见疾病难以覆盖。其次就是其诊断结果的可解释性，大数据 + 深度学习方案旨在找到数据之间的关联性，而没法获取数据之间的因果联系。比如比如根据“低热，盗汗，咯血”给出诊断结果是“肺结核”，但是为什么给出这样的诊断缺乏医学知识的支持，可能增强患者的不信任感。最后是数据的可靠性问题，我国的电子病历尚处于起步阶段，很多医生不太习惯使用电子病历，不少医生填写病历就像在写“八股文”，这样就造成电子病历中可能存在错误，如何甄别并修改这些错误是大数据方案无法解决的。

除了上述的基于文本匹配，规则，贝叶斯网以及深度学习的辅助诊断方案，学术界还提出了一些其他技术做智能诊断，比如模糊逻辑（Fuzzy logic参考文献：Fuzzy Logic and its Applications in Medicine，Fuzzy Set Theory in Medical Diagnosis），马尔科夫链逻辑网络（Markov logic network参考文献：Markov logic networks）等，但它们都只停留在理论阶段而没有实际落地的项目。它们的实际落地都都受限于数据的获取，比如模糊逻辑模型难以获取模糊度函数，马尔科夫链逻辑网络难以获取规则的权重。

## 1.3 研究内容

本文针对大数据 + 深度学习方案所面临的问题，提出了一个新的辅助诊断解决方案：通过从医书等可信医学知识源中提取医学知识，然后将其表达成逻辑公式，通过逻辑推理来进行辅助诊断。具体的研究内容如下：

(1) 知识库的构建

为了实现本文的辅助诊断系统，我们首先设计了更符合医学知识表示并且便于自然语言处理的 schema，包括实体的类型以及实体之间的关系。然后根据设定好的 schema，从可信知识源中（主要是医书教材）中采用人工标记以及自然语言处理的方法抽取出了大量的医学知识，通过这些医学知识构建了知识库，供后续的推理算法使用。

(2) 知识纠错方案

由于知识源的数据可能存在错误，以及知识抽取的过程中也可能引入错误，所以最终获取的知识可能存在错误。因为本文采用的是基于医学知识进行诊断而非大数据方案，

知识的错误会很大程度影响最终诊断的结果。所以必须对抽取的知识进行校对以及纠错，本文就从逻辑的角度，提出了纠正知识库中错误的方案，可以对知识库中的部位错误知识（主要是互相矛盾的知识）进行自动检测。

(3) 疾病诊断算法

基于已经构建好的知识库，本文设计了可以根据知识库以及患者的症状信息进行疾病诊断的算法。主要思路是首先根据患者主述信息从知识库中提取相关知识，然后将相关知识编码成逻辑公式，然后组合变换逻辑公式构建可满足性问题的实例，最后通过 SMT 求解器求解可满足性问题来完成诊断。

(4) 实际应用的系统

本文的所有技术方案不仅仅是停留在理论层面，而是实现出了可用的系统，并且已经在合作医院中落地测试。

## 1.4 章节安排

本文共有七个章节组成，其中主要内容如下所示：

第一章为绪论部分。绪论中主要阐述了本文的研究背景及意义、国内外的研究现状以及本文主要的研究内容。在章节的最后姐好啊了本文的组织结构。

第二章是对本文所用到的背景知识以及相关工作的介绍。具体包括知识表示方法，知识存储所用到的数据库以及逻辑推理中的可满足性问题、SMT 求解器以及 unsat core 等知识。

第三章主要阐述了构建医学知识库的流程。我们设计了两版建模医学文本的 schema，然后根据 schema 从医书中抽取医学知识，最后将这些医学知识存储到 Neo4j 数据库中。

第四章介绍了辅助诊断内核的设计与实现。辅助诊断内核是本文系统的核心，我们详细介绍了其内部各个模并且通过举例更加形象的说明各模块的实现机制和交互流程。

第五章讲解我们如何使用逻辑推理来完成对知识的纠错，这是本文的创新点之一。我们通过逻辑推理出 unsat core，以及特定结构的子图来完成对错误知识的检测。

第六章是实验部分。在实验章节中，我们会简要介绍整个系统的架构。然后通过实验说明我们知识库的规模，辅助诊断的准确率以及修正错误知识对准确率的提升。

第七章是总结和展望，对本文的工作进行了总结和未来工作的展望。

# 第二章　背景知识及相关工作

本文实现的辅助诊断系统使用到了包括知识表示、知识存储，逻辑推理等方面的技术，本章会我们的辅诊系统所依赖的技术做背景以及相关工作的介绍。

## 2.1　知识表示

根据Knowledge Representation and Reasoning的定义，知识表示是将知识表示成符号化、便于计算机自动处理的数据。更直观的来讲，知识的表示就是对知识的一种描述，或者说是对知识的一组约定，一种计算机可以接受的用于描述知识的数据结构，是对知识的符号化、形式化或模型化。它是机器通往智能的基础，使得机器可以像人一样运用知识。经过多年的研究，目前知识表示方法得到了深入的发展，目前使用较多的知识表示方法主要有以下几种：

### 2.1.1　逻辑表示法

符号主义学派认为人的认知基元是符号，而认知过程就是符号的操作过程，人通过自己的眼睛观察客观事物，用符号的形式表示出来，而计算机也是一个对逻辑符号表示的知识进行演绎的物理符号系统。因此早期的知识表示方法都是采用逻辑（主要是一阶谓词逻辑）表示法，以谓词形式来表示动作的主体、客体，是一种叙述性知识表示方法。比如“所有阔叶植物是落叶植物“这条知识可以 2–1谓词逻辑公式来表示，其中谓词 F(x) 表示 x 是阔叶植物，谓词 G(x) 表示 x 是落叶植物。利用逻辑公式，人们能描述对象、性质、状况和关系。它主要用于自动定理的证明，比如 ACL2，Prover9 和 Metis参考文献等。

$$(\forall x)(F(x) \rightarrow G(x)) \tag{2–1}$$

虽然一阶谓词逻辑可以精确、无二义的表示知识，但是由于量词的存在以及其不可判定性参考文献，导致很多给予一阶谓词逻辑的推理系统性能低效甚至无法判定。因此有很多推理系统采用一些一阶谓词逻辑的子集来表示知识：

(1) Horn 逻辑参考文献。Horn 逻辑是一阶谓词逻辑的子集，它的表达形式简单，复杂度低。著名的 Prolog参考文献语言就是基于 Horn 逻辑设计实现的。

(2) 描述逻辑参考文献。描述逻辑是一阶谓词逻辑的可判定子集，用于描述概念，属性。对于术语知识库的构建提供了便捷的表达形式，比如 1.2.1.1提到的 SNOMED CT 就是采用描述逻辑来表示知识。

### 2.1.2　产生式规则表示法

产生式规则表示，又称 IF-THEN-ElSE 表示，它表示一种条件-结果形式，是一种比较简单表示知识的方法。IF 后面部分描述了规则的先决条件，而 THEN 后面部分描述了规则的结论。正如 1.2.2.2所说，早期的专家系统（比如 CADUCEUS，MYCIN，SMH.PAL参考文献）都是基于产生式规则来表示知识并推理。

### 2.1.3 框架表示法参考文献：A framework for representing knowledge

框架 (Framework) 是把某一特殊事件或对象的所有知识储存在一起的一种复杂的数据结构。其主体是固定的，表示某个固定的概念、对象或事件，其下层由一些槽 (Slot) 组成，表示主体每个方面的属性。框架是一种层次的数据结构，框架下层的槽可以看成一种子框架，子框架本身还可以进一步分层次为侧面。槽和侧面所具有的属性值分别称为槽值和侧面值，如图 2–1所示。框架结构表达能力强，层次结构丰富，提供了有效的组织知识的手段，其数据结构与人类的思维和问题求解过程相似。但是其无法表达过程性的知识，而且许多实际情况与框架原型不匹配。

| | |
|---|---|
| 框架名： | 小A抢劫杀人案 |
| 犯罪意图： | 抢劫 |
| 犯罪结果： | 杀人 |
| 被杀者： | 小B |
| 知情人： | 小C |
| 罪犯： | 小A |
| 条件一： | 有小C指控小A |
| 条件二： | 小A招认 |

图 2–1 框架知识表示法示例

Figure 2–1 Example of framework representation

### 2.1.4 语义网络表示法

语义网络（Semantic Network）是知识表示中最重要的方法之一，是一种表达能力强而且灵活的知识表示方法。从图论的观点看，它是一个“带标识的有向图”。语义网络利用节点和带标记的边构成的有向图描述事件、概念、状况、动作及客体之间的关系。具体来说，语义网络用节点来表示对象、概念等实体，用边表示实体之间的关系，如图 2–2所示。本文就是使用语义网络来表示医学知识。语义网络的优势在于便于人类理解和展示，但是其也有一些缺点：1) 无法像逻辑或者规则表示法那样直接进行知识推理，所以本文的辅诊系统会把语义网络转换成逻辑公式进行推理。2) 相关概念与关系容易混淆。因为语义网络的中的节点和边可以代表任何的对象、概念、属性和关系。所以在构建语义网络时很容易混淆它们之间的隶属关系。比如”熊“可以是”哺乳动物“的一个子类（subclassof），也可以是”哺乳动物“的一个实例（is-a），这样类似的关系很容易混淆。因此本文在构建医学语义网络的时候设计了较

为简单的 schema，没有太多的层级结构。当然语义网络只是一个知识表示的概念，其具体的实现方式大多都是采用 RDF 三元组的形式。

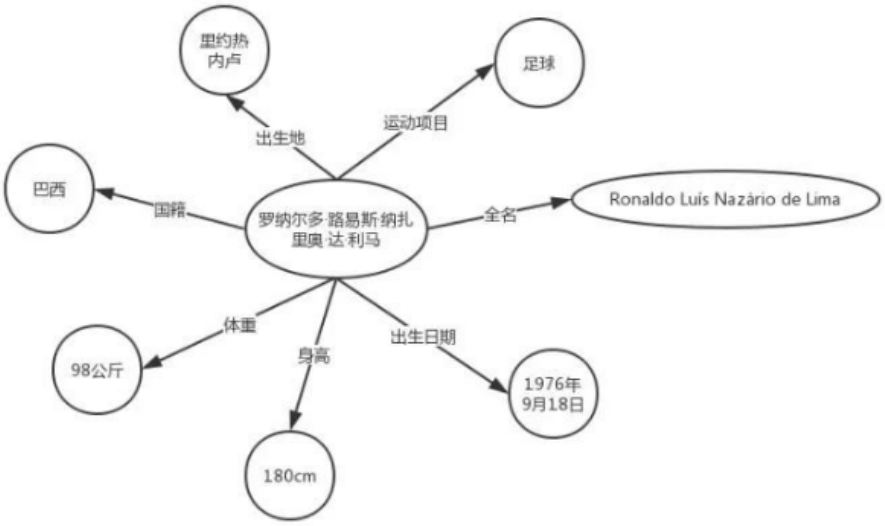



图 2–2 语义网络示例

Figure 2–2 Example of semantic network

#### 2.1.4.1 RDF

RDF(Resource Description Framework) 是一种用来描述网络资源的框架。RDF 数据集是形如 < 主体 (subject)，谓语 (predict)，客体 (object)> 的三元组构成的数据集，每个三元组可以描述一条知识，比如 < 肺结核，导致，咳嗽 > 就可以描述 “肺结核导致发烧” 这条知识。RDF 数据集可以看成是很大的有向图（即语义网络）G=(V,E), 其中 V 是所有顶点的集合，即所有的主体和客体，E 是边的集合，不同的谓语表示不同类型的边。此外 RDF 的标准还提供了一些预定义的规则，比如用 type 这种谓语描述实体间的类属关系。现在主流的知识库大多采用 RDF 三元组的方式来实现，比如 WordNet，BabelNet，HowNet参考文献。本文的医学知识库其实就是一个 RDF 图，如画图，通过预先定义的一些实体和关系类型，然后从知识源中抽取 RDF 三元组集构建知识库。

## 2.2 知识存储

知识存储就是如何把抽取出来的知识存储到持久化设备中，本文采用 Neo4j 和 Redis参考文献两种数据库来存储知识。本节会对这两种数据库做详细的介绍。

### 2.2.1 Neo4j

Neo4j 是一个高性能的,NOSQL 图形数据库，它将结构化数据存储在网络上而不是表中。它是一个嵌入式的、基于磁盘的、具备完全的事务特性的 Java 持久化引擎，但是它将结构化数据存储在网络 (从数学角度叫做图) 上而不是表中。Neo4j 也可以被看作是一个高性能的图引擎，该引擎具有成熟数据库的所有特性。程序员工作在一个面向对象的、灵活的网络结构下，而不是严格、静态的表中——但是他们可以享受到具备完全的事务特性、企业级的数据库的所有好处。因为我的知识采用语义网络的方式来表示，而且经常会在知识网络上一些查询（比如子图匹配，最短路径等），所以 Neo4j 的图存储方式非常适合存储我们的医学知识。同时，如图 2–3所示，Neo4j 自带的数据展示工具也非常有助于数据的可视化及调试。

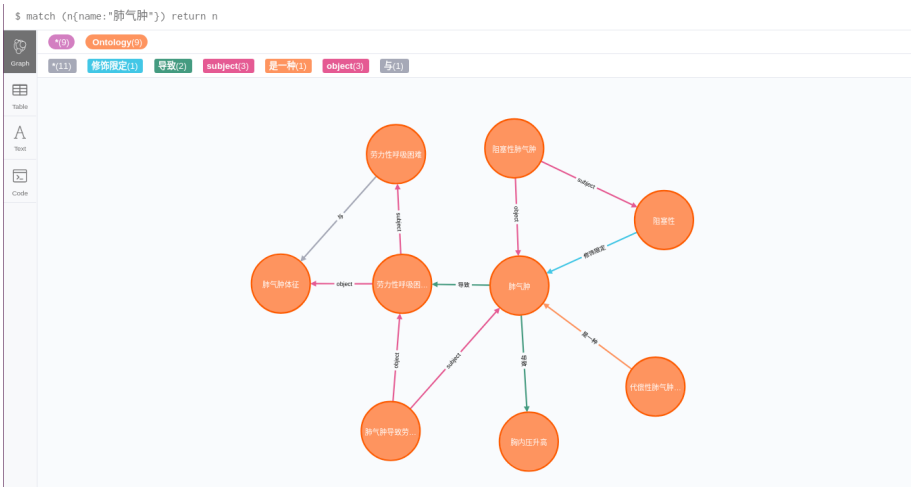



图 2–3 Neo4j 查询界面截图

Figure 2–3 Screenshot of Neo4j query interface

### 2.2.2 Redis

在我们的实际使用中，我们需要从全体的医学知识网络中提取出一张特定的子图。而从全图中提取一张特定子图在 Neo4j 中性能一般，所以我们又在 Neo4j 上加了一层 Redis 缓存来提高查询性能。Redis（Remote Dictionary Server）是一个开源的使用 ANSI C 语言编写、支持网络、可基于内存亦可持久化的日志型、Key-Value 数据库。Redis 本质上是一个 Key-Value 类型的内存数据库，很像 memcached，整个数据库统统加载在内存当中进行操作，定期通过异步操作把数据库数据 flush 到硬盘上进行保存。因为是纯内存操作，Redis 的性能非常出色，每秒可以处理超过 10 万次读写操作，是已知性能最快的 Key-Value DB。正是因为这些特性，我们会把底层的 Neo4j 中的数据缓存在 Redis 中供上层调用查询，弥补了 Neo4j 在子图查询性能上的不足。

## 2.3 逻辑推理

本文为了保证辅助诊断的严谨以及可解释性，采用了逻辑推理的方式来进行诊断。主要思想就是从知识库中提取知识编码成逻辑公式，然后将逻辑公式组合变换成可满足性问题的实例，最后通过 SMT 求解器求解可满足性问题实例完成诊断。之所以采用求解可满足性问题的方式进行辅助诊断，很重要的一个原因是可满足性问题中的一个重要概念 unsat core，它可以完成部分知识纠错的功能，这也是本文的一个重点所在。所以本节会对可满足性问题、SMT 求解器以及 unsat core 做详细的介绍。

### 2.3.1 可满足性问题

布尔可满足性问题（Boolean Satisfiability Problem），有时简称为可满足性问题或者 SAT 问题，其是第一个被证明的 NP-完全（complete）问题参考文献。给定一个命题逻辑公式参考文献，可满足性问题是确定是否存在一组公式中变量的真值指派，使得公式为真。举例来说，比如公式 $f_1 = a \wedge b$ 就是可满足的（Staisfiable），因为存在一组变量的真值指派 $\{a = True, b = True\}$ 使得 $f_1$ 为真。而公式 $f_2 = a \wedge \neg a$ 就是不可满足的（Unstaisfiable），因为无论变量 $a$ 如何取值，公式 $f_2$ 都不可能为真。在我们的辅助诊断系统中，就可以使用求解可满足性问题来进行诊断，比如我们考虑两个疾病”膀胱炎“和”急性肾盂肾炎“，根据人卫教材《内科学》对两种疾病的描述，它们都会有泌尿系统症状，比如”尿频、尿急、尿痛“，但是一般”膀胱炎“不会伴有高烧，而”急性肾盂肾炎“会伴有”高烧“症状。我们可以获取医学知识”膀胱炎会导致尿频、尿急、尿痛，但无高烧“，”急性肾盂肾炎会导致尿频、尿急、尿痛、高烧“。如果患者的症状是”尿频、尿痛、高烧“，我们可以编码成 2–2的命题逻辑公式（膀胱炎：$d_1$，肾盂肾炎：$d_2$，尿频：$s_1$，尿急：$s_2$，尿痛：$s_3$，高烧：$s_4$）。通过求解 $f$ 的可满足性，我们得出解为 $\{d_1 = False, d_2 = True\}$。根据解我们就可以做出患者可能患有”急性肾盂肾炎“但不患有”膀胱炎“的诊断。

$$f = (d_1 \rightarrow s_1 \wedge s_2 \wedge s_3 \wedge \neg s_4)(d_2 \rightarrow s_1 \wedge s_2 \wedge s_3 \wedge s_4)(s_1 \wedge s_3 \wedge s_4)(d_1 \vee d_2) \tag{2–2}$$

### 2.3.2 SMT 求解器

作为 SAT 问题的扩展，SMT 问题（Satisfiability Modulo Theories Problem）处理的对象是一阶逻辑公式，相比于命题逻辑，增加了谓词和量词，其中谓词可以用特定的理论来解释，比如数组理论，整数算数理论参考文献等，很大程度增强了 SMT 公式的表达能力. 用以求解 SMT 问题的自动化工具称为 SMT 求解器（SMT Solver）.SMT 求解技术在有界模型检测、基于符号执行的程序分析、线性规划和调度、测试用例生成以及电路设计和验证等领域有非常广泛的应用. 很多科研机构以及公司都在致力研发正确率高性能优异的 SMT 求解器，并且已经成功应用到了具体的领域. 目前流行的 SMT 求解器有: Barcelogic[10], Beaver[11],Yices[12] 以及 Z3[13] 等. 其中，由微软主导开发的 Z3 SMT 求解器所支持的理论最多，性能也最好，因此本文使用了 Z3 SMT 求解器作为我们的求解引擎。

### 2.3.3 Unsat Core

**定义 2.1** (Unsatisfiable Core).参考文献：On Computing Minimum Unsatisfiable Cores 给定一个公式 $\varphi$, $\varphi_c$ 是 $\varphi$ 的一个 unsatisfiable core 当且仅当 $\varphi_c$ 不可满足且 $\varphi_c \subseteq \varphi$

**定义 2.2** (Minimal Unsatisfiable Core). 考虑一个公式 $\varphi$ 和它的所有 unsatisfiable core:$\{\varphi_{c_1}, ..., \varphi_{c_j}\}$。则 $\varphi_{c_k}$ 是一个 minimal unsatisfiable core 当且仅当 $\forall \varphi_{c_i} \in \{\varphi_{c_1}, ..., \varphi_{c_j}\}, 0 < i \leq j : |\varphi_{c_i}| \geq |\varphi_{c_k}|$

Unsatisfiable core，简称为 unsat core。根据 2.1中的定义，一个命题逻辑公式的 unsat core 是该公式的任意一个不可满足子集。直观来讲，unsat core 就是原公式中互相矛盾的子句集，这对知识纠错很有帮助：如果我们把医学知识表达成一个个命题逻辑子句，然后将它们合取成命题逻辑公式求解其可满足性。如果是不可满足的，那么求解其 unsat core 就可以获得互相矛盾的知识，而互相矛盾的知识中必定有错误的知识，只要修改 unsat core 中的错误知识就完成了部分的知识纠错。因此 unsat core 的功能就是帮助我们快速定位错误知识。当然，因为一个不可满足的公式可能存在很多个 unsat core，而且它们之间可能存在包含关系，所以如果在一个不可满足公式的所有 unsat core 上都做纠错就会变得低效。如果我们能找到最小的 unsat core（Minimum Unsatisfiable Core），也就是包含子句数量最少的 unsat core（严格定义见 2.2），然后在最小 unsat core 上做纠错，就能提高知识纠错的效率。SMT 求解器就有获取不可满足公式最小 unsat core 的功能，这也是我们选择 SMT 求解器作为我们求解引擎的原因之一。

## 2.4 本章小结

本章主要介绍了知识表示、知识存储以及逻辑推理的相关背景与相关工作。在知识表示中，虽然有逻辑、规则、框架和语义网络等多种方案，但考虑到各自的优劣势，现在用语义网络来表示知识已经成为了主流。在语义网络的基础上，我们还介绍了其实现方式之一–RDF，RDF 三元组不仅可以简单方便的表示知识，而且嵌套的 RDF 也可以表示很复杂的知识，这就是本文选择使用 RDF 三元组表示知识的原因。在知识存储中，我们介绍了十分适合储存语义网络这种图结构数据的 Neo4j, 同时为了提高算法性能，我们又用 Redis 对 Neo4j 的数据做了一层缓存。最后，在逻辑推理中，我们介绍了可满足性问题并举例说明如何通过求解可满足性问题来进行辅助诊断，其次又对可满足性问题的求解器 SMT 求解器做了介绍，最后我们通过介绍 unsat core 来说明如何使用它来进行快速的知识纠错。

# 第三章　医学知识库的构建

本章主要介绍我们构建知识库的主要流程和相关方法。如图 3–1所示，我们构建医学知识库的流程主要分为三块：1) 知识表示：即根据医学知识源中文本数据特点为医学文本建模，设计一套适合表达医学知识的 schema。2) 知识抽取：利用我们自己开发的数据标记平台，采用自然语言处理加人工审核的办法从医学文本中抽取出符合 schema 的知识，生成 RDF 三元组文件。3) 知识存储：将 RDF 三元组数据存储到 Neo4j 中。因为本文的重点在于知识表示与推理，所以本章会着重介绍知识表示的实现方式，尤其是如何设计合理的知识 schema 来表达医学知识。而对于知识抽取和知识存储模块只会简略介绍。

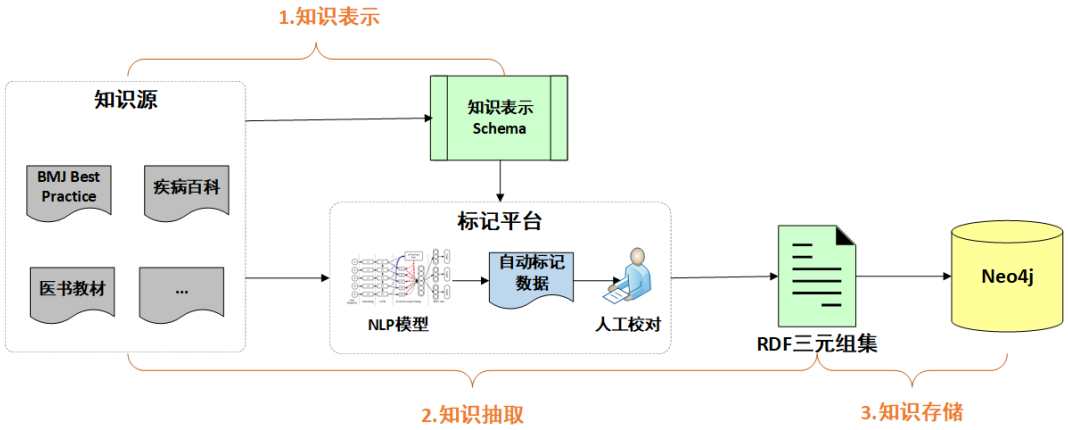



图 3–1 医学知识库构建流程

Figure 3–1 Medical knowledge base construction process

## 3.1　知识表示

本节主要介绍知识表示的相关细节。具体包括医学数据来自哪些知识源，以及如何设计知识的 schema 来给知识源中的医学知识建模。

### 3.1.1　知识源

我们的医学数据主要有以下几个来源：

- BMJ Best Practice：我们爬取了 BMJ Best Practice 中约 1000 种疾病主题下的诊断步骤的文本数据，然后基于这些文本做医学知识抽取。因为 BMJ Best Practice 中的数据按主题组织，每个主题下有比如诊断、治疗、随访等模块的内容（如图 1–3所示），而只有诊断模块中的诊断步骤子模块包含了较多的症状以及检查信息。而辅助诊断比较关心疾病的症状以及检查指标，因此我们主要使用每个疾病主题的诊断步骤模块的数据。

- 人卫版医书教材：除了 BMJ Best Practice 中数据以外，我们还预计抽取约 20 本人卫版医书教材中的医学知识。包括《诊断学》、《内科学》、《妇产科学》等等。这个过程一直在进行当中，到目前为止，我们一共抽取完了《诊断学》、《内科学》和《耳鼻咽喉头颈外科学》中的医学知识。
- 疾病百科数据：除了直接从上述较为权威的医学资源中抽取医学知识外，我们另外也爬取一些了医学百科数据，包括百度百科、BabelNet、CMeSH、春雨医生和医脉通等。因为这些数据并不算权威的医学数据源，因此我们没有直接在这些数据上做知识抽取，而是根据这些数据构建了医学同义词库。同义词库在我们系统中的医学实体对齐参考文献，患者意图理解等功能上很有帮助。

### 3.1.2 医学文本建模

本节主要介绍如何设计医学知识表示的 schema。schema 主要包含实体类型以及实体之间的关系，当然这些实体类型都是医学上的类型（比如病症、检查、病史等），关系也都是医学上一些因果关系（比如导致、检查发现等）。在我们设计知识的 schema 并且实际做知识抽取的过程中发现：医学文本语义复杂，如果要非常准确的给医学文本建模就要设计很复杂的 schema。在我们最初的版本中，我们设计了 30 个实体类型，以及 25 个实体关系。这种复杂的 schema 无论对标记还是 NLP 来说都是很大的负担。我们手工标记的数据都漏洞百出，NLP 学出来的数据自然错误更多。而如果设计简单的 schema，很多医学知识难以表达，导致很多医学文本中的知识丢失。在最终的权衡之下，我们设计了相对较为简单，也可以表达大部分问诊层面医学知识的 schema。前后一共经历两个大版本，多个小版本，我们会主要介绍两个大版本的 schema。

### 3.1.3 知识表示 schema v1.0

#### 3.1.3.1 实体类型

知识表示 schema v1.0 中的实体类型如表 3–1所示。为了较为准确的表达医学实体，我们设计了 23 种实体类型。其中有一些实体类型可能比较令人困惑，我们对其做具体说明：

- 六种修饰语：分别修饰医学实体的状态、性质、程度、频率、方位、数量。我们区分这六种修饰语的目的是为了提高问诊的友好程度以及智能程度。比如我们想要询问患者是否有“持续咳嗽”，我们可以首先询问患者是否有“咳嗽”症状，其次根据“持续”是“性质修饰语”的类型来询问患者咳嗽的频率是多少。这样的询问方式可以很好的优化用户体验。
- 病症：我们在知识表示层不区分疾病和症状，因为这两个概念的界限很模糊，标记人员很容易标错。而在标记平台获得病症实体后，我们将病症实体与 ICD-10参考文献疾病编码匹配来区分疾病和症状。
- 代词：医学文本中有非常多的省略以及指代。比如在《耳鼻咽喉头颈外科》中的“耳气压伤”章节，其只有标题中提到了“耳气压伤”，在之后所有的文本中都是用“本病”指代。比如“本病症状有耳闷、耳鸣等”，所以我们在标记时用“本病”指代“耳气压

伤”，然后“本病”会导致“耳闷、耳鸣”。如果直接标记标题中的“耳气压伤”导致正文中的“耳闷、耳鸣”，这对 NLP 自动学习会是很大的负担。

- 否定词：表示对实体的否定。有一些医书中明确说明某种疾病不会有某种症状。比如“膀胱炎不会伴有高烧”，其中“不”就是否定词来修饰“高烧”。

表 3–1 实体类型–schema v1.0

Table 3–1 Entity type–schema v1.0

| 实体名称 | 含义 | 例子 |
|---|---|---|
| 状态修饰语 | 表示医学实体状态的修饰语 | 异常（红细胞膜），正常 (体重) |
| 性质修饰语 | 表示医学实体性质的修饰语 | 功能性（梗阻），溶血性（黄疸） |
| 程度修饰语 | 表示医学实体程度的修饰语 | 中度（肥胖），深度（昏迷） |
| 频率修饰语 | 表示医学实体频率的修饰语 | 持续（咳嗽），反复（皮疹） |
| 方位修饰语 | 表示医学实体位置的修饰语 | 后（腰椎），下（呼吸道） |
| 数量修饰语 | 表示医学实体数量的修饰语 | 大量（脓血），较少（骨关节破坏） |
| 病症 | 疾病和症状 | 咳嗽，肺结核 |
| 代词 | 指代其他医学实体的词 | 者，该病 |
| 部位 | 解剖学部位 | 腰椎，喉咙 |
| 否定词 | 表示对被修饰实体的否定 | 否，非，没有 |
| 检查 | 包括体检和化验等所有理化检查 | 结肠镜检查，内镜检查 |
| 病史 | 包括家族史、既往史、现病史 | 吸烟史，饮酒史 |
| 事件 | 不可归结为理化因素的外部因素 | 摔倒，手术 |
| 颜色 | 颜色 | 红色，绿色 |
| 生物 | 生物 | 毒蕈 |
| 时间 | 时间 | 数分钟，数小时 |
| 化学物质 | 表示具体的某一种或某一类化学物质 | 乙醇，皮质醇 |
| 食品 | 食品 | 酒，浓茶 |
| 年龄 | 年龄的数值 | 35 岁 |
| 人群分类 | 区分人群的实体 | 青壮年，老年人 |
| 数据 | 具体的数据数值（带单位） | 80%，10μmol/dl |
| 生理概念 | 描述生理构造，生理过程 | 红细胞，骨髓 |
| 病理概念 | 描述病理构造，病理过程 | 坏死组织，病变瓣膜 |

#### 3.1.3.2 关系类型

知识表示 schema v1.0 中的实体关系类型如表 3–2所示。在最初的版本中，我们设计了共 19 种实体间的关系类型，这里节选出其中的一些关系做具体说明：

- 两种导致关系：强导致、弱导致。强导致表示实体之间很强的因果关系，比如如果患有疾病“肺结核”大概率就有“咳嗽”症状，那么就可以表示为（肺结核，强导致，咳

嗽)。强导致可以作为推理的依据，具体来说，如果患者患有“肺结核”，我们就断定他有“咳嗽”症状。而如果患者没有“咳嗽”症状，我们就认为其不患有“肺结核”。而弱导致表示较弱的因果关系，比如“吃辣可诱发胃溃疡”，其实“吃辣”不一定就会导致“胃溃疡”，所以他们之间没有很强的因果关系。弱导致无法作为推理的依据，我们无法根据患者“吃辣”推理出“胃溃疡”，只能将“吃辣”作为“胃溃疡”的加分项。4

- 两种并列关系：与、或。与表示实体之间的合取关系，它们必须都要满足，缺一不可。比如“急性溶血时可有发热、寒战“就表示患有“急性溶血”时会同时有“发热”和“寒战”两个症状，所以可以表示为（急性溶血，强导致，（发热，与，寒战））。而或表示实体之间的析取关系，表示实体之间无须都满足，只要满足其一即可。比如“干咳或刺激性咳嗽常见于急性咽喉炎”表示如果患有“急性咽喉炎”，则会有“干咳”或者“刺激性咳嗽”，可能两者都有，也可能只有两者其一，所以就可以用“或”来表示它们之间的关系。即（急性咽喉炎，强导致，（干咳，或，刺激性咳嗽））。
- 条件为关系：表示某种关系发生的条件，常作为嵌套关系使用。比如“痛风性肾病晚期会出现高血压、水肿等症状”，其中“痛风性肾病”可能有“早期，中期，晚期”等多个阶段，每个阶段的症状都不一样，所以要用条件为表示不同的阶段，这句话可以用（（痛风性肾病，条件为，晚期），强导致，（高血压，与，水肿））这样的嵌套关系来表达。
- 检查发现关系：表示某项检查发现的结果。因为检查的结果都是比如“阳性，阴性”这样通用的用词，其实这里的“阳性，阴性”都是做了省略，其真正的含义是某项检查阳性，某项检查阴性。所以我们对检查发现关系做了规定，如果疾病 X 要执行检查 A,A 的检查结果是 B。则必须标记为 (X, 强导致,(A, 检查发现,B))，我们在后台会将检查结果 B 加上其检查项 A 作为前缀合并成 A 检查发现 B。整个关系可以处理为 (X, 强导致，A 检查发现 B)。
- 不导致：表示对因果关系的否定。如“多数哮喘患者并不会出现发热症状”可以标记为：（哮喘, 不导致, 发热），其和（哮喘，强导致，（不，修饰限定，发热））等价。增加不导致关系是为了便于标记人员理解。
- 然后：表示医学实体发生的先后顺序。医学生有一些症状确实是有先后发生顺序的，比如“偏头痛的头痛症状在呕吐后减轻”，这句话可以标记为（偏头痛，弱导致，（呕吐，然后，头痛减轻））。

我们设计的关系 schema 中最大的特点是嵌套关系的存在，比如“条件为”，“并列”等关系基本都会在嵌套关系中出现。嵌套关系可以较为准确的表达医学知识，比如“痛风性肾病”可能有“早期，中期，晚期”等多个阶段，每个阶段的症状都不一样，这些知识用“条件为”这样的嵌套关系就可以很好的表达出来。当然，嵌套关系的存在也增加了人工标记和 NLP 自动处理的难度。

表 3–2 关系类型–schema v1.0

Table 3–2 Relation type–schema v1.0

| 关系名称 | 含义 | 例子 |
|---|---|---|
| 强导致 | 表示很强的因果关系 | 如“凡能引发溶血的疾病都可引发溶血性黄疸”这句话可以表示为：(溶血, 强导致, 溶血性黄疸) |
| 弱导致 | 表示较弱的因果关系 | 如“吃辣可诱发胃溃疡”这句话可以提取：(吃辣, 弱导致, 胃溃疡) |
| 修饰限定 | 表示某词语对另一个词语的修饰和约束。 | 如“粪胆原随之增加”可以提取：(增加, 修饰限定, 粪胆原)；如“后天性获得性溶血性贫血”中，可以通过“修饰限定”关系标记为 (后天性, 限定修饰,(获得性, 限定修饰,(溶血性, 限定修饰, 贫血)))。 |
| 与 | 表示实体之间的与关系 | 如“急性溶血时可有发热、寒战”可以标记为（急性溶血，强导致，（发热，与，寒战）） |
| 或 | 表示实体之间的或关系 | 如“干咳或刺激性咳嗽常见于急性咽喉炎”可以标记为（（急性，修饰限定，咽喉炎），强导致，（干咳，或，（刺激性，修饰限定，咳嗽）） |
| 条件为 | 表示某种关系的发生条件，因此常嵌套使用，在标记时注意标记成树状结构 | 如“血红蛋白在组织蛋白酶的作用下形成血红素和珠蛋白”可以标记为：((血红蛋白, 条件为, 组织蛋白酶), 转变为,(血红素, 并列, 珠蛋白)) |
| 调查病史 | 表示确诊某种病症需要询问某病史 | 如“肥胖多余家族肥胖史相关”可以标记为：(肥胖，调查病史，家族肥胖史) |
| 执行检查 | 表示确诊某种病症需要进行的检查，包括查体和化验 | 如“腹泻患者请执行粪便检查”可以标记为：(腹泻，执行检查，粪便检查) |
| 检查发现 | 表示某种检查得出特定的结果，如果疾病 X 导致检查 A 的结果是 B，则应该标记为:(X, 导致,(A, 检查发现,B)) | 如“隐血试验阳性”应标记为：(隐血试验, 检查发现, 阳性)；“血液检查除贫血外尚有网织红细胞增加”应标记为：(血液检查, 检查发现,(贫血, 并列,(增加, 修饰限定, 网织红细胞))) |
| 是一种 | 表示概念的从属关系 | 如“常见病因有先天性溶血性贫血，如海洋性贫血”应标记为 ((海洋性, 限定修饰, 贫血), 是一种,(先天性, 限定修饰,(溶血性, 修饰限定, 贫血))) |
| 等价 | 表示数量上的相等，或概念上的相同 | 如“水肿（edema）”可以标记为（水肿，等价，edema） |

续下页

续表 3–2

| 关系名称 | 含义 | 例子 |
| --- | --- | --- |
| 不导致 | 表示对因果关系的否定 | 如“多数哮喘患者并不会出现发热症状”可以标记为: (哮喘, 不导致, 发热) |
| 转变为 | 表示物质之间的转化 | 如“血红蛋白在组织蛋白酶的作用下形成血红素和珠蛋白”可以标记为: ((血红蛋白, 条件为, 组织蛋白酶), 转变为,(血红素, 并列, 珠蛋白)) |
| 定义为 | 表示显示或隐式的定义说明 | 如“有不同程度的贫血和血红蛋白尿（尿呈酱油色或茶色）”可以标记为 (血红蛋白尿, 定义为,((酱油色, 并列, 茶色), 修饰限定, 尿)) |
| 指代 | 表示代词指代的具体实体 | 如（本病，指代，耳气压伤） |
| 作用于 | 表示某实体对另一个实体施加作用 | 如“肝脏处理尿胆原的能力降低”可以标记为: (降低, 修饰限定,(肝脏, 作用于, 尿胆原)) |
| 表现为 | 表示某个实体的具体特点。 | 如“腹泻表现为每天排便次数增多”可以标记为（腹泻，表现为，排便次数增多） |
| 比例为 | 表示某实体占另一实体的比例 | 如“循环血容量 30% 以上...”可以标记为（循环血容量，比例为，30% 以上） |
| 然后 | 表示时间先后顺序 | 如“偏头痛的头痛症状在呕吐后减轻”可以标记为（偏头痛，弱导致，（呕吐，然后，头痛减轻）） |

#### 3.1.3.3 关系约束

在我们用设计好的 schema 做实际的知识抽取过程中发现，三元组（实体 A，关系 R，实体 B）中关系 R 其实对实体 A 和实体 B 的类型有约束。比如（功能性，强导致，梗阻）这种类型的三元组就不是合法的三元组，因为“功能性”属于性质修饰语，而修饰语在语法上都是作为名词的定语出现，而强导致关系的主体必须是名词，所以“功能性”就不能作为强导致关系的主体。又比如（肺结核，检查发现，肺部阴影）也是非法三元组，因为检查发现要求主体的类型必须是检查，而“肺结核”的类型是病症，不能成为检查发现关系的主体。所以对于三元组（实体 A，关系 R，实体 B），我们制作了关系 R 对 A 和 B 类型的约束表，主要分为两种:

- 白名单: 规定实体 A，B 合法的类型范围。详细数据见表 3–3。
- 黑名单: 规定实体 A，B 非法的类型范围。详细数据见表 3–4。

### 3.1.4 知识表示 schema v2.0

在我们使用知识表示 schema v1.0 抽取知识的过程中，我们发现标记以及 NLP 抽取的知识错误率很高。究其原因，还是因为 schema 过于复杂，23 种实体类型以及 19 种关系类型对于标记人员已经是很大的负担，对于 NLP 来说更是难上加难。考虑到我们是辅诊系统，主要

表 3–3 关系约束白名单–schema v1.0

Table 3–3 Whitelist of relationship constraints–schema v1.0

| 关系类型 | 主体类型 | 客体类型 |
|---|---|---|
| 条件为 | 代词，病史，生理概念，病理概念 | 病症，代词，病史，事件，时间，数据 |
| 调查病史 | 病症，代词 | 病史，代词 |
| 执行检查 | 病症，代词 | 检查，代词 |
| 检查发现 | 检查，代词 | 病症，代词，化学物质，生理概念，病理概念 |
| 指代 | 代词 | * |
| 比例为 | 病症，代词，病史，生物，化学物质，食品，人群分类，生理概念，病理概念 | 数据 |
| 转变为 | 化学物质，生物，生理概念，病理概念，代词 | 化学物质，生物，生理概念，病理概念，代词 |
| 作用于 | 部位，生物，化学物质，食品，生理概念，病理概念，代词 | 部位，生物，化学物质，食品，生理概念，病理概念，代词 |
| 然后 | 病症，代词 | 病症，代词 |

* 为通配符，表示所有实体类型.

表 3–4 关系约束黑名单–schema v1.0

Table 3–4 Blacklist of relationship constraints–schema v1.0

| 关系类型 | 主体类型 | 客体类型 |
|---|---|---|
| 强导致 | 性质修饰语，状态修饰语，程度修饰语，频率修饰语，方位修饰语，数量修饰语，部位，否定词，颜色，时间，数据 | * |
| 弱导致 | 性质修饰语，状态修饰语，程度修饰语，频率修饰语，方位修饰语，数量修饰语，部位，否定词，颜色，时间，数据 | * |

* 为通配符，表示所有实体类型.

涉及到的是问诊层面的知识，这个层次的知识从本质上来讲只需要关注疾病和症状之间的相关关系即可，配合上一些基本的患者信息，如性别、年龄、病史等。而对于复杂的检查指标（比如“影像检查发现病灶 >2cm,AFP $>$ 400ng/ml “），深层的病理发生机制（比如“当失水量达体重的 2% ~ 3% 时，渴感中枢兴奋，刺激抗利尿激素释放，水重吸收增加，尿量减少，尿比重增高”）其实无需关注，所以我们简化了我们知识表示的 schema，虽然有一些复杂的医学知识无法表达，但是优化了知识结构，减少了知识抽取的复杂度。

#### 3.1.4.1 实体类型

知识表示 schema v1.0 中的实体类型如表 3–5所示, 我们主要做了以下修改：

- 合并六种修饰语。在实际标记过程中，标记人员很难区分这六种修饰语，而且修饰语的类别除了优化用户体验之外没有太大的作用。所以我们将其直接合并成一种“修饰语”类型，不再区分。
- 删除病理概念实体类型。因为问诊层不会涉及到太多深层的病理机制，所以我们删除了“病理概念”类型。
- 增加“阶段”实体类型。医学文本中有很多疾病的分型以及阶段区分，之间一直没有很好的类型来建模这种实体（用“时间”来表示不太妥当），所以我们增加了“阶段”实体类型。

表 3–5 实体类型–schema v2.0

Table 3–5 Entity type–schema v2.0

| 实体名称 | 含义 | 例子 |
|---|---|---|
| 修饰语 | 六种修饰语合并，不再区分 | 先天性，升高 |
| 病症 | 疾病和症状。这里症状只包含患者可以主观感受的那些，其他的不算在内。 | 咳嗽，肺结核 |
| 代词 | 指代其他医学实体的词 | 者，该病 |
| 部位 | 解剖学部位 | 腰椎，喉咙 |
| 否定词 | 表示对被修饰实体的否定 | 否，非，没有 |
| 病史 | 包括家族史、既往史、现病史 | 吸烟史，饮酒史 |
| 事件 | 不可归结为理化因素的外部因素 | 摔倒，手术，高温环境，月经前 |
| 颜色 | 颜色 | 红色，绿色 |
| 生物 | 生物 | 毒蕈 |
| 时间 | 时间 | 数分钟，数小时 |
| 化学物质 | 表示具体的某一种或某一类化学物质 | 乙醇，皮质醇 |
| 食品 | 食品 | 酒，浓茶 |
| 性别 | 男女性别 | 男性，女性 |
| 年龄 | 年龄的数值 | 35 岁 |
| 生理概念 | 仅用于拼接实体的时。当短实体类型不能用以上类型表达的时候使用 | 尿，血压 |
| 阶段 | 用以表示病症的阶段或分型 | 早期，一型 |

#### 3.1.4.2 关系类型

知识表示 schema v1.0 中的实体类型如表 3–6所示，我们主要做了以下修改：

- 去除“执行检查”以及“检查发现”关系，因为我们决定只关注问诊层的知识，所以

对复杂的检查指标知识不在抽取。当然，由于我们采用 schema v1.0 抽取过很多医学知识，所以我们现在的医学知识库中依然存在不少“执行检查”和“检查发现”关系。

- 将“修饰限定”修改为“拼接”关系。修饰限定本质上就是将修饰语和主语拼接，所以修改为“拼接关系”。而且像“后天性获得性溶血性贫血”这种实体之间会把他们标记成（后天性，修饰限定，（获得性，修饰限定，溶血）），现在不在强调修饰关系，直接标记为一个“病症”实体。
- 将“等价”修改为“同义词”关系，便于理解。
- 增加“并发症”关系。医书中有很多关于并发症的描述，所以我们增加这个关系。

其实我们不在强调修饰语，且只关心问诊层的知识之后，虽然会遗失部分知识，但这些知识对于辅诊来说是无关紧要的，但是知识抽取的复杂度降低了很多。减轻了标记人员的负担，也提高了 NLP 的准确率。

#### 3.1.4.3 关系约束

schema v2.0 中关系约束的设计思路和 schema v1.0 没有太大区别，参考表 3–3和表 3–4，这里不再赘述。

## 3.2 知识抽取

如图 3–1所示，我们自己开发了标记平台来进行知识的标记以及自动抽取。具体来说，就是先使用标记平台标记少量文本做训练集训练出 NLP 模型，然后使用 NLP 模型对批量做预标记。在我们的实践过程中，NLP 中命名实体识别 (Named Entity Recognition) 准确率较高，在分词，实体类型判别上都有超过 95% 的准确率，但关系抽取（Relation Extraction）准确率就相对较低，有很多的嵌套关系漏标或者错标，所以 NLP 模型无法完全代替人直接完成数据标记。所以我们采用通过 NLP 模型预标记数据，然后人工审核（增、删、改）预标记数据的方案来完成医学文本中知识的抽取。因为本文的核心在于知识表示与推理，所以偏重于 NLP 技术的知识抽取不是本文的重点，这里不再详细介绍。

## 3.3 知识存储

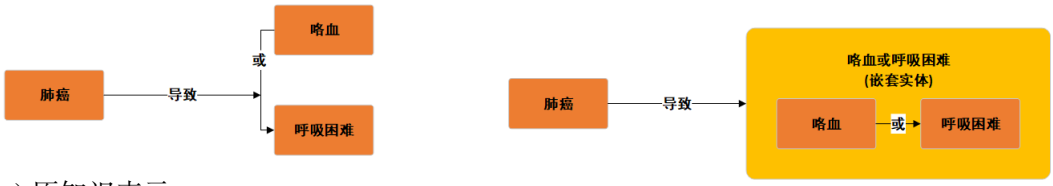
因为 RDF 数据集可以看成一个很大的有向图，即语义网络，所以我们直接使用图的方式来存储 RDF 三元组数据。具体来说，RDF 三元组形如 < 主体 (subject)，谓语 (predict)，客体 (object)>，我们把主体和客体分别当成两个节点，谓语当成主体指向客体的有向边存储到 Neo4j 中。具体形式如 2–3所示。但是在我们的实际使用过程中，我们需要从全体的医学知识网络中提取出一张特定的子图（比如以肺结核为中心的相关知识子图，具体定义见 引用定义）。而从全图中提取一张特定子图在 Neo4j 中性能一般，又因为我们现在的需求中这张子图不会动态更新，所以我们又在 Neo4j 上加了一层 Redis，将这些特定的子图全部缓存在 Redis 中来提高查询性能。这一块也不是本文的重点，不在详细介绍。

在存储时，“拼接”和“指代”关系我们会做特殊处理：1) 会将“拼接”关系的主客体合并成一个实体存储，比如”（典型性，拼接，肺炎）” 合并成一个实体” 典型性肺炎” 存储，因

表 3–6 关系类型–schema v2.0
Table 3–6 Relation type–schema v2.0

| 关系名称 | 含义 | 例子 |
|---|---|---|
| 强导致 | 同 schema v1.0 | 同 schema v1.0 |
| 弱导致 | 同 schema v1.0 | 同 schema v1.0 |
| 不导致 | 同 schema v1.0 | 同 schema v1.0 |
| 拼接 | 表示组成实体的词素的拼接，在词汇不能直接标记为实体的时候使用。原则上能不用这个关系标记实体的时候就不用它。词素将按照“主 + 宾”的顺序拼接成一个实体。拼接以后实体的类型遵从最右侧优先级规则，取主语和宾语实体中优先级较高的类型。 | 如“粪胆原随之增加”可以提取：(粪胆原: 生理概念, 拼接, 增加: 修饰语)，则等价于标记实体“粪胆原增加”，合并实体类型为“病症”；又如“肠内的尿胆原增加”可以表示为：((肠: 部位, 拼接, 尿胆原: 生理概念), 拼接, 增加: 修饰语)，等价于标记实体“肠尿胆原增加”，合并实体类型为“病症”。如“后天性获得性溶血性贫血”这种直接连在一起的实体，则不需要拼接关系，直接标记成一整个实体即可。 |
| 与 | 同 schema v1.0 | 同 schema v1.0 |
| 或 | 同 schema v1.0 | 同 schema v1.0 |
| 条件为 | 同 schema v1.0 | 同 schema v1.0 |
| 调查病史 | 同 schema v1.0 | 同 schema v1.0 |
| 是一种 | 同 schema v1.0 | 同 schema v1.0 |
| 同义词 | 表示实体之间是同义词, 替换 schema v1.0 中的等价关系 | 如“水肿（edema）”可以标记为（水肿，同义词，edema） |
| 指代 | 同 schema v1.0 | 同 schema v1.0 |
| 然后 | 同 schema v1.0 | 同 schema v1.0 |
| 并发症 | 表示病症的并发症 | 如“肠结核的并发症有肠出血”可以标记为 (肠结核，并发症，肠出血) |
| 否定修饰 | 同 schema v1.0 | 同 schema v1.0 |

此知识库中不会存在“拼接”关系。2) 会将“指代关系”的主体替换成客体存储，比如“(本病, 指代，肺炎)，(本病，强导致，咳嗽)”转化成“(肺炎，强导致，咳嗽)”存储，因此知识库中不会存在“指代”关系。此外，我们在知识存储时引入了嵌套实体的概念。当我们需要存储嵌套关系的时候，比如“(肺癌，导致，或（呼吸困难，咯血))”，其知识的表示方式如图 3–2a所示，但是 Neo4j 中边只能由实体连接到实体，无法从实体连到边。所以我们额外增加了一个嵌套实体“咯血或呼吸困难”，然后从“肺癌“连接到这个嵌套实体上，如图 3–2b所示。




a) 原知识表示

a) Origin knowledge representation

b) Neo4j 的存储方式

b) Storage method of Neo4j

图 3–2 嵌套实体在 Neo4j 中的存储示例

Figure 3–2 Storage example of nested entities in Neo4j

## 3.4 本章小结

本章主要介绍了我们医学知识库的构建流程，具体包括知识表示，知识抽取以及知识存储三部分。而知识表示是本文的重点，所以我们花了较大篇幅来介绍。知识表示的核心就是设计可以建模医学文本中知识的 schema，这是一个 trade-off 的过程：schema 简单，很多医学知识无法表达。schema 复杂，知识抽取的准确率低。我们前后经历了两个大版本的迭代，最终设计了 3.1.4中的 schema v2.0。schema v2.0 中包括 16 种实体类型，14 种实体关系，而且只关注问诊层面的医学知识，对于复杂的检查指标和病理机制不在关注，这极大程度降低了知识抽取的复杂度。我们的知识 schema 中最核心的点就是使用嵌套关系来表达复杂的医学知识。最后我们还介绍了关系约束的白名单和黑名单，对特定关系的主客体的类型做了约束，来降低知识抽取的错误率。

# 第四章　辅助诊断内核

本章主要介绍辅助诊断内核的设计与实现。辅助诊断内核的主要工作就是利用知识库中的医学知识结合患者的症状信息完成对患者所患疾病的预测。其具体的模块设计如图 4–1所示，主要分为四个子模块：

(1) KBConverter: 根据患者的主述信息从医学知识库中抽取相关的医学知识，并将这些医学知识编码成逻辑公式交给 ReasoningSolver 使用。

(2) ReasoningSolver: 将 KBConverter 输出的命题逻辑公式组合变换成可满足性问题的实例，并使用 SMT Solver 进行求解。求解过程中会需要更多逻辑变量的赋值信息，ReasoningSolver 会将这些变量抛出交给 QuestionGenerator 处理，QuestionGenerator 处理完后会将这些变量赋完值交给 ReasoningSolver 继续处理。ReasoningSolver 会根据医学知识的逻辑规则结合患者的症状信息排除掉很多无关的疾病，最后输出一个候选症状集。

(3) QuestionGenerator：因为 ReasoningSolver 处理的对象都是逻辑变量及公式，当 ReasoningSolver 需要确定某些变量的赋值时，会将这些变量抛给 QuestionGenerator 处理。QuestionGenerator 会根据这些变量解码生成具体的问题（单选，多选等）与患者交互，获得患者的答案后 QuestionGenerator 会将这些问题答案再编码成赋完值的变量交给 ReasoningSolver 继续处理。

(4) DiagnosisCalculator：因为 ReasoningSolver 只会排除无关的疾病，对于剩余的疾病都会当成候选疾病。一是这些候选疾病可能数量很多；二是这些疾病没有评分的先后关系。DiagnosisCalculator 会根据患者的症状信息给这些疾病评分，并最终将得分最高的 5（5 只是阈值，可以调整）个疾病返回。

本章之后的小节会具体介绍每个模块的功能以及实现方式，对于模块之间如何交互也会详细介绍。

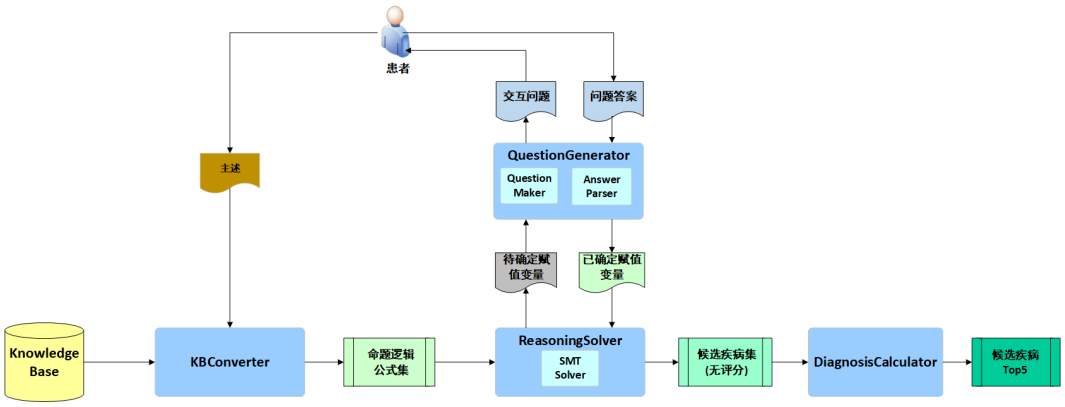



图 4–1 诊断内核模块设计

Figure 4–1 The module design of aided diagnose core

## 4.1 KBConverter

本节会介绍 KBConverter 功能及实现方式，其具体的功能主要有两个：1) 根据患者的主述信息从知识库中提取相关的医学知识 2) 将这些医学知识编码成命题逻辑公式。

### 4.1.1 提取相关医学知识

正如前文 2.1.4提到，我们构建的医学知识库可以看做是一张语义网络图，具体来说，就是一张 RDF 图 G。根据主述信息从医学知识库中提取相关知识就是从 G 中提取出一张子图 G'。

图 4–2就是我们从知识库中提取的以‘咳嗽，恶心’为主述，hop=3[1]的子图。

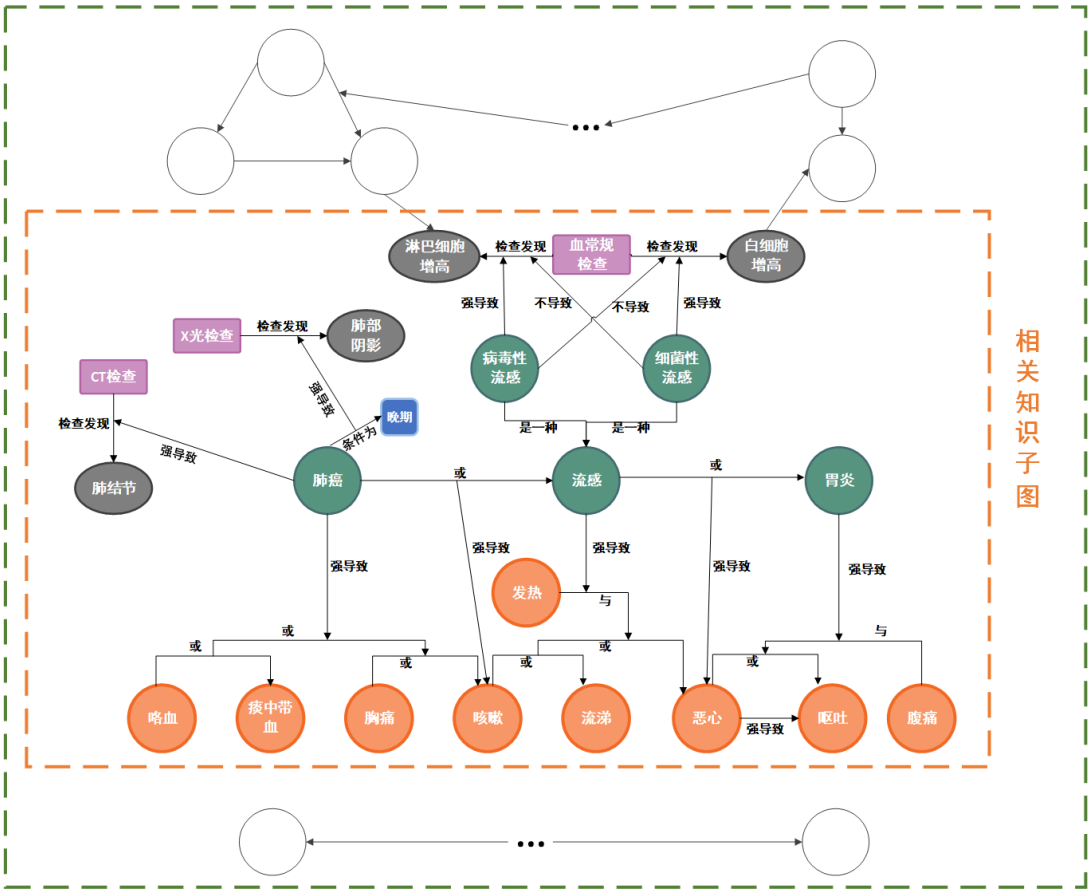



图 4–2 相关医学知识子图示例[2]

Figure 4–2 Example of the related medical knowledge graph

[1]实际应用中，我们一般使用 hop=4 或 hop=5 的子图，其会包含数千个节点及关系，这里为了方便展示取 hop=3。

[2]实际存储中，图中起点或终点在边上的关系都应该使用如 3–2示例的嵌套实体的存储方式，本图为了展示方便没有实际画出嵌套实体。

### 4.1.2 命题逻辑公式编码

提取出子图后，我们要把子图编码成逻辑公式集合，也就是将子图中的 RDF 三元组转换成命题逻辑公式。编码过程分为两步:

(1) 将 RDF 三元组中主客实体编码成逻辑变量。这一步比较简单，就是将不同名称的实体映射成不同的逻辑变量。比如“肺炎, 流感, 胃炎”分别映射成“$d\_1, d\_2, d\_3$”

(2) 将将 RDF 三元组中谓语编码成命题逻辑连接词。具体编码方式见表 4–2，我们节选一些做具体说明:

(a) ”然后”关系。“然后”强调的是实体发生的先后顺序，因为我们采用朴素的命题逻辑编码 RDF 三元组，因此无法表示顺序关系。我们采用将”（a, 然后,b）”合并成一个实体”a 然后 b”实体的方式，然后直接询问患者是否有“a 然后 b”症状，让患者判断先后关系。这样的交互方式显然不太友好，之后我们会考虑加入时序逻辑 参考文献解决这个问题。

(b) 非强因果关系。我们认为“弱导致”，“调查病史”，“并发症”都不是强因果关系，不会对其编码。

(c) ”不导致”和“否定修饰”关系。这两个关系其实本质一样，比如“（膀胱炎，不导致，高烧）”等价于“（膀胱炎，强导致，（不，否定修饰高烧））”。我们只是为了标记人员理解同时使用了两种关系

所以，相关知识子图 G’（图 4–2）可以采用如下方式编码成命题逻辑规则集 $R$。为了处理方便，我们会把所有规则统一转换拆分成 $(v_{l_1} \wedge v_{l_2} \wedge ... \wedge v_{l_m}) \rightarrow (v_{r_1} \vee v_{r_2} \wedge ... \wedge v_{r_n})$ 的形式，比如”$d_1 \vee d_2 \rightarrow s_4$”可以拆分成两条规则”$d_1 \rightarrow s_4, d_2 \rightarrow s_4$”（转换拆分算法比较简单，这里不详细介绍）:

(1) 用逻辑变量编码实体，如表 4–1所示

(2) 用逻辑连接词编码关系。基于 4–2的编码方式，G’ 可以编码成公式集 4–1。

$$
\begin{aligned}
R = \{ & d_1 \rightarrow s_1 \vee s_2 \vee s_3 \vee s_4,\ d_2 \rightarrow s_4 \vee s_5 \vee s_6,\ d_2 \rightarrow \wedge s_7,\ d_3 \rightarrow s_7 \vee s_8, \\
& d_3 \rightarrow \wedge s_9,\ d_1 \rightarrow s_4,\ d_2 \rightarrow s_4,\ d_2 \rightarrow s_7,\ d_3 \rightarrow s_7,\ s_7 \rightarrow s_8,\ d_4 \rightarrow d_2, \\
& d_5 \rightarrow d_2,\ d_1 \rightarrow c_1,\ d_1 \wedge t_1 \rightarrow c_2,\ d_4 \rightarrow c_3,\ d_5 \rightarrow c_4,\ d_4 \rightarrow \neg c_4,\ d_5 \rightarrow \neg c_3 \}
\end{aligned} \tag{4–1}
$$

表 4–1 实体类型–schema v1.0

Table 4–1 Entity type–schema v1.0

| 实体类型 | 实体名称 | 逻辑变量 |
|---|---|---|
| 疾病 | 肺癌、流感、胃炎、病毒性流感、细菌性流感 | d1,d2,d3,d4,d5 |
| 症状 | 咯血、痰中带血、胸痛、咳嗽、流涕、发热、恶心、呕吐、腹痛 | s1,s2,s3,s4,s5,s6,s7,s8,s9 |
| 检验指标 | 肺结节、肺部阴影、淋巴细胞增高、白细胞增高 | c1,c2,c3,c4 |
| 阶段 | 晚期 | t1 |

## 4.2 ReasoningSolver

本节会介绍 ReasoningSolver 的运行机制。ReasoningSolver 的主要功能就是结合 KBConverter 输出的公式集以及患者症状信息编码而成的公式集，将它们组合成可满足性问题的实例，然后去求解可满足性问题从而排除无关的疾病，最后输出候选症状集。

### 4.2.1 构建可满足性问题实例

在我们构建可满足性问题实例时，需要用到三类信息：

(1) 相关子图的知识。根据公式集 4–1我们可以构建相关知识子公式 $f_1 = (d_1 \rightarrow s_1 \vee s_2 \vee s_3 \vee s_4) \wedge (d_2 \rightarrow s_4 \vee s_5 \vee s_6) \wedge (d_2 \rightarrow s_7) \wedge (d_3 \rightarrow s_7 \vee s_8) \wedge (d_3 \rightarrow s_9) \wedge (d_1 \rightarrow s_4) \wedge (d_2 \rightarrow s_4) \wedge (d_2 \rightarrow s_7) \wedge (d_3 \rightarrow s_7) \wedge (s_7 \rightarrow s_8) \wedge (d_4 \rightarrow d_2) \wedge (d_5 \rightarrow d_2) \wedge (d_1 \rightarrow c_1) \wedge (d_1 \wedge t_1 \rightarrow c_2) \wedge (d_4 \rightarrow c_3) \wedge (d_5 \rightarrow c_4) \wedge (d_4 \rightarrow \neg c_4) \wedge (d_5 \rightarrow \neg c_3)$。

(2) 患者的症状信息，包括患者的主述以及之后通过交互获取的患者症状信息。我们依然用图 4–2为例，患者的主述信息为“咳嗽，恶心”。假设通过与患者的交互（至于如何选择症状询问我们会在 4.4.2详细解释）了解到患者“发热，无腹痛”。则可以构建患者症状信息的子公式 $f_2 = (s_4) \wedge (s_7) \wedge (s_6) \wedge (\neg s_9)$。

(3) 患者必须患有一种疾病的假设。我们认为既然患者前来就诊，并且说出自己的主述症状，那么患者必然患有相关知识子图中的某一个或多个疾病。则我们可以构建子公式 $f_3 = (d_1 \vee d_2 \vee d_3 \vee d_4 \vee d_5)$

最终合取三个子公式构建了可满足性问题的实例公式 $f = f_1 \wedge f_2 \wedge f_3$, 即:

$$
\begin{aligned}
f =& (d_1 \rightarrow s_1 \vee s_2 \vee s_3 s_4) \\
& \wedge (d_2 \rightarrow s_4 \vee s_5 \vee s_6) \\
& \wedge (d_2 \rightarrow s_7) \\
& \wedge (d_3 \rightarrow s_7 \vee s_8) \\
& \wedge (d_3 \rightarrow s_9) \\
& \wedge (d_1 \rightarrow s_4) \\
& \wedge (d_2 \rightarrow s_4) \\
& \wedge (d_2 \rightarrow s_7) \\
& \wedge (d_3 \rightarrow s_7) \\
& \wedge (s_7 \rightarrow s_8) \\
& \wedge (d_4 \rightarrow d_2) \\
& \wedge (d_5 \rightarrow d_2) \\
& \wedge (d_1 \rightarrow c_1) \\
& \wedge (d_1 \wedge t_1 \rightarrow c_2) \\
& \wedge (d_4 \rightarrow c_3) \\
& \wedge (d_5 \rightarrow c_4) \\
& \wedge (d_4 \rightarrow \neg c_4) \\
& \wedge (d_5 \rightarrow \neg c_3) \\
& \wedge (s_4) \\
& \wedge (s_7) \\
& \wedge (s_6) \\
& \wedge (\neg s_9) \\
& \wedge (d_1 \vee d_2 \vee d_3 \vee d_4 \vee d_5)
\end{aligned}
\tag{4–2}
$$

### 4.2.2 求解候选疾病集

我们通过使用 SMT 求解器求解可满足性来确定候选疾病。具体来说，如果 $f \wedge d_i$ 是可满足的，那么 $d_i$ 就是候选疾病，反之则不是。因为 $f \wedge d_i$ 可满足，意味着 $f \rightarrow d_i$问豪豪，规范写。我们用伪代码 4–1来表示求解候选症状集的过程。同样本章的例子，最后求得候选症状集为 $\{d_1, d_2, d_4, d_5\}$, 即 {肺癌, 流感, 病毒性流感, 细菌性流感}, 疾病“胃炎”被排除。

## 4.3 QuestionGenerator

QuestionGenerator 即问题生成器，它主要做两件事情（如图 4–3）:

**算法 4–1** 求解候选疾病集伪代码

```
Input: 可满足性问题实例公式 f, 相关知识子图中所有疾病集合 D
Output: 候选疾病集合 D'
D' ← Φ;
for each d in D do
    if f ∧ d is SATISFIABLE then
        add d to D';
    end
end
return D';
```

(1) 生成问题。因为 ReasoningSolver 在求解过程中需要知道更多逻辑变量的取值信息，所以它会将它需要知道的逻辑变量抛给问题生成器来处理，问题生成器会将逻辑变量解码成症状，然后查看症状是主观可感，还是查体可知或者是检验才能得知。比如“发热”就是查体项目，问题生成器会生成一个查体问题，提示外部硬件设备对患者做体温检测。而“腹痛”是主观可感症状，问题生成器会直接抛出一个单选题“请问您是否腹痛”让患者选择。而“白细胞增高”需要专业的血常规检查才能得知，所以问题生成器不会处理整个症状[1]，并反馈给 ReasoningSolver。

(2) 分析问题答案。当收到患者的答案（或者是硬件检测的结果），问题生成器会将结果再编码为赋值之后的逻辑变量返回给 ReasoningSolver。比如 ReasoningSolver 需要知道 $s_9$ 的赋值，即需要知道患者是否有“腹痛”症状，问题生成器生成单选题“请问您是否腹痛”让患者作答。假设患者回答没有腹痛，问题生成器则将 $s_9$ 的赋值，即 $\{s_9 : False\}$ 返回给 ReasoingSolver 迭代求解。

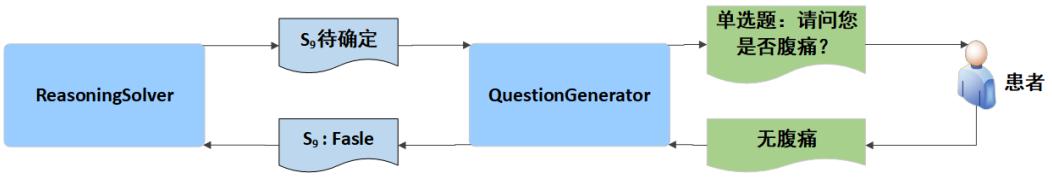



图 4–3 QuestionGenerator 运行流程示例

Figure 4–3 Example of running process of QuestionGenerator

## 4.4 DiagnosisCalculator

本节会介绍 DiagnosisCalculator 的主要功能。DiagnosisCalculator 主要负责症状以及疾病得分的计算：1) 计算症状的得分，帮助 ReasoningSolver 选择交互症状。2) 计算疾病得分，从而为 ReasoningSolver 输出的候选疾病排序，最终选择得分最高的 5 个疾病（5 是阈值，可以调整）。

[1]我们的系统有已经出厂的硬件诊断舱，可以完成体温、体重、血压等基本特征的检测。问题生成器会根据硬件的能力来决定是否处理特定症状。

### 4.4.1 疾病打分

因为 ReasoningSolver 采用逻辑求解的方式，其只会排除无关的疾病，对于候选疾病却无先后之分。举个通俗的例子，比如患者说“我头痛，脚不痛”，根据患者的症状信息逻辑只能排除患者没有脚部疾病，而头部以及腹部疾病都是候选疾病。但是既然患者只说明自己“头痛”，并没有说自己任何腹部的症状，我们其实可以认为患者患有头部疾病的可能性要大于腹部疾病（这里并不能直接排除腹部疾病，因为可能真的有一些腹部疾病通过内部生理机制引起头痛）。所以，我们认为“疾病 $A$ 与患者症状的关联度高于疾病 $B$（患者的症状信息对疾病 $A$ 的贡献度高于 $B$），则患者患有疾病 $A$ 的可能性高于疾病 $B$”。我们采用统计的方式来量化症状对疾病的贡献度，具体就是采用了 TF-IDF（term frequency–inverse document frequency）参考文献:Using TF-IDF to Determine Word Relevance in Document Queries 方法来评价贡献度。TF-IDF 是一种统计方法，用以评估一字词对于一个文件集或一个语料库中的其中一份文件的重要程度。字词的重要性随着它在文件中出现的次数成正比增加，但同时会随着它在语料库中出现的频率成反比下降。我们认为每个疾病都是一篇“文档”，这个疾病的症状就是文档中的“关键词”，如果对于患者具有某个症状 $s$，但是 $s$ 对于疾病 $A$ 的“关键程度”大于 $B$，则疾病 $A$ 的得分高于疾病 $B$。

给定所有症状的集合为 $S$，集合 $S$ 中某个症状 $s$，所有候选疾病集合 $D'$，集合 $D'$ 中的某个候选疾病 $d$，则症状 $s$ 对候选疾病 $d$ 的贡献度可以按如下方式计算：

$$\begin{aligned} w_{s,d} &= tf_{s,d} \times idf_{s,D'} \\ &= \frac{n_{s,d}}{\sum\limits_{s_i \in S}(n_{s_i,d} + n_{\neg s_i,d})} \times \frac{|d' \in D' : s \in d'|}{|D'|} \end{aligned} \tag{4–3}$$

$$\begin{aligned} w_{\neg s,d} &= tf_{\neg s,d} \times idf_{\neg s,D'} \\ &= \frac{n_{\neg s,d}}{\sum\limits_{s_i \in S}(n_{s_i,d} + n_{\neg s_i,d})} \times \frac{|d' \in D' : \neg s \in d'|}{|D'|} \end{aligned} \tag{4–4}$$

其中 $n_{s,d}$ 表示症状 $s$ 在疾病 $d$ 出现的次数，而 $\sum\limits_{s_i \in S} n_{s_i,d}$ 表示 $d$ 中所有症状出现的总次数，$|d' \in D' : s \in d'|$ 表示包含症状 $s$ 的候选疾病数量。$w_{\neg s,d}$ 表示某个症状的负出现对某个疾病的贡献度，比如“膀胱炎不会引起高烧”，则“不高烧”会对“膀胱炎”有贡献度，而“高烧”对“膀胱炎”没有贡献。

接着我们定义 $n_{s',d}$（s' 为文字，即 $s' = s$ 或 $s' = \neg s$），症状 $s'$ 在疾病 $d$ 出现的次数自然不能单纯地使用症状 $s'$ 作为关键词在医书疾病文档 $d$ 出现的次数来定义，因为这种单纯文本匹配的方式（IBM Watson 采用的策略）只考虑了症状关键词是否在疾病文档中出现，而没有考虑它们之间的逻辑关系。比如在《肺结核》文档中有描述，“咳嗽是肺结核的常见症状，咳嗽是一种呼吸道症状”，这里“咳嗽”虽然在“肺结核”文档中出现两次，但是第二次只是对“咳嗽”的描述，其实和肺结核没有任何关系，它的出现不应该对“肺结核”有任何贡献。因此我们采用知识库中的知识来表达症状在疾病中出现的次数，因为知识库中的知识规则只考虑疾病与症状之间的逻辑关系，而没有逻辑关系的无关描述并不会转化成逻辑规则。比如“咳嗽是肺结核的常见症状，咳嗽是一种呼吸道症状”会提取出两条规则“肺结核 → 咳嗽，咳嗽 → 呼吸道症状”，基

于这两条规则，“咳嗽” 只在“肺结核” 中出现了一次。对于相关知识规则集 $R$，其中的某条规则 $r = (s'_{l_1} \wedge ... \wedge' s_{l_m}) \wedge (d_{l_1} \wedge ... \wedge d_{l_n})) \wedge (v'_{l_1} \wedge ... \wedge v'_{l_k}) \rightarrow (s'_{r_1} \wedge ... \wedge s'_{r_m}) \wedge (d_{r_1} \wedge ... \wedge d_{r_n})) \wedge (v'_{r_1} \wedge ... \wedge v'_{r_k})$，规则左边的症状集为 $S_l = \{s'_{l_1}, ..., s'_{l_m}\}$，疾病集 $D_l = \{d_{l_1}, ..., d_{l_n}\}$，右边的症状集为 $S_r = \{s'_{r_1}, ..., s'_{r_m}\}$，疾病集 $D_r = \{d_{r_1}, ..., d_{r_n}\}$。注意这里 $s'$ 表示是文字 (literal)，它可以表示变量的正出现或者负出现，比如 $s'_{l_1}$ 可能是变量正出现 ($s_{l_1}$)，也可能是变量负出现 ($\neg s_{l_1}$)。从逻辑的角度看，是左边的症状会导致右边的疾病，左边的疾病会导致右边的症状，而左边（右边）的症状和疾病之间其实没有任何逻辑关系。我们认为 $S_l(S_r)$ 中的所有症状在 $D_r(D_l)$ 中的每个疾病中一共出现了一次。即给定某条规则公式 $r$，$S_l, S_r$ 分别为规则左右两边的症状集，$D_l, D_r$ 分别为规则左右两边的疾病集，则有

$$\begin{aligned} \forall d \in D_r, \sum_{s' \in S_l} n_{lr\_s',d}(r) = 1 \\ \forall d \in D_l, \sum_{s' \in S_r} n_{rl\_s',d}(r) = 1 \end{aligned} \tag{4–5}$$

其中 $n_{lr'_s,d}(r)$ 表示在规则 $r$ 中，左边症状在右边疾病中出现的次数,$n_{lr'_s,d}(r)$ 表示右边症状在左边疾病中出现的次数，然后我们将 $S_l(S_r)$ 中的症状在 $D_r(D_l)$ 中疾病的出现次数平均，得到：

$$\begin{aligned} \forall d \in D_r, \forall s' \in S_l, n_{lr\_s',d}(r) = \frac{1}{|S_l|} \\ \forall d \in D_l, \forall s' \in S_r, n_{rl\_s',d}(r) = \frac{1}{|S_r|} \end{aligned} \tag{4–6}$$

然后我们可以定义公式 $r$ 中症状在疾病中出现的次数，即：

$$n_{s',d}(r) = n_{lr\_s',d}(r) + n_{rl\_s',d}(r) \tag{4–7}$$

有了公式集中某一条公式中症状在疾病中出现次数的定义，我们可以定义在相关知识规则集中症状在疾病出现的次数。对于相关知识规则集 $R$, 有：

$$n_{s',d} = \sum_{r \in R} n_{s',d}(r) \tag{4–8}$$

我们可以使用伪代码 4–2来表示计算症状在疾病中出现次数的字典。

基于公式 4–3定义的症状对候选疾病的贡献度，我们可以定义候选疾病的得分。设患者的症状信息集合 $S'$(里面可能包含症状的正出现和负出现，比如患者 “发热，无腹痛” )。注意患者的症状信息除了患者明确说出的症状之外（包括患者的主述以及通过与患者交互获得的症状），还包含我们知识库推断出的症状信息。比如如果知识库中有 “咳嗽强导致声嘶”，患者说明自己 “咳嗽”，我们可以推断出患者也具有 “声嘶” 症状。如果我们的知识库构建的全面且准确，这个功能将非常强大，因为它充分发挥了逻辑链条的作用，可以推断出尚未出现但准确的信息。给定某个候选疾病 $d$，则候选疾病的得分可以如下定义:

$$score_d = \sum_{s' \in S'} w_{s',d} \tag{4–9}$$

我们依然用图 4–2 为例，患者的主述信息为 “咳嗽，恶心”。假设通过与患者的交互了解到患者 “发热，无腹痛”，即患者说出的症状信息为 “咳嗽，恶心，发热，无腹痛 “; 而通过相关知识

的规则集，我们可以推断出患者会有“呕吐”症状（因为相关规则中存在规则“恶心 → 呕吐”），即患者的症状信息集合 $S' = \{s_4, s_7, s_6, \neg s_9, s_8\}$，可以计算出候选症状集 $D' = \{d_1, d_2, d_4, d_5\}$ 中每个疾病的得分，最后得出 $score_{d_2} > score_{d_1} > score_{d_4} = score_{d_5}$，即得分先后关系为“流感 > 肺癌 > 病毒性感冒 = 流行性感冒”。

### 4.4.2 症状打分

给症状打分主要是为了选择交互症状询问患者：ReasoingSolver 在推理过程中需要知道更多症状变量的信息，最简单的办法自然是把所有症状都向患者询问一遍，但是这种交互方式显然不友好。所以 ReasoingSolver 需要找到包含信息量最多的症状变量去询问患者。症状打分都是为了量化症状包含的信息量，使得 ReasoingSolve 可以问较少的问题就可以做出诊断。这种量化方式没有统一的标准，我们主要尝试了三种量化方式。

#### 4.4.2.1 朴素方法和大师兄讨论决定

#### 4.4.2.2 最大熵

我们通过计算每个症状变量的熵值，选出熵值最大的变量去询问患者。假设候选疾病集合为 $D' = \{d_1, d_2, ..., d_t\}$，对于某个症状变量 s，患者回答为 $True$ 的概率为 $p(s|d_1 \vee d_2 \vee ... \vee d_t)$，我们设为 $x$。根据香农的信息熵理论参考文献，变量 s 的熵为：

$$H(s) = -(xlogx + (1-x)log(1-x)) \tag{4–10}$$

其中 $x = p(s|d_1 \vee d_2 \vee ... \vee d_t)$，但是这个值我们现在无法获取[1]，所以我们采用如下公式来模拟：

$$p(s|d_1 \vee d_2 \vee ... \vee d_t) \approx \sum_{d_i \in D} p(s \wedge d_i) = \sum_{d_i \in D} p(s|d_i) \times p(d_i) \tag{4–11}$$

其中 $p(s|d_i)$ 表示患者在患有疾病 $d_i$ 的条件下，出现症状 $s$ 的概率，我们可以用 $tf_{s,d_i}$ 来估计，即

$$p(s|d_i) = tf_{s,d_i} = \frac{n_{s,d_i}}{\sum_{s_j \in S} n_{s_j,d_i}} \tag{4–12}$$

而 $p(d_i)$ 表示患者在当前情况下患有疾病 $d_i$ 的概率，我们可以用疾病得分来估计。用 $d_i$ 的得分除以所有候选疾病的总得分来估计患者患有疾病 $d_i$ 的概率，即：

$$p(d_i) = \frac{score_{d_i}}{\sum_{d_j \in D'} score_{d_j}} \tag{4–13}$$

基于公式 4–10- 4–13，我们可以计算每个未确定赋值症状变量的熵值，然后挑选熵值最大的症状询问患者。基于之前的例子，患者的主述“咳嗽，恶心”，之后提取的相关知识子图如图 4–2，基于对剩余未确定赋值变量熵值的计算，我们会选择“发热”症状询问患者。

[1]之后的工作我们会通过大量病历提取这个概率值

#### 4.4.2.3 最大期望疾病贡献度

某个症状 $s$ 的取值会对疾病有不同的贡献度，症状为 $True$ 会对包含 $s$ 正出现的疾病有贡献，，症状为 $False$ 会对包含 $s$ 负出现的疾病有贡献。基于公式 4–3, 症状会对疾病的贡献度可以量化。对于候选疾病集 $D' = \{d_1, d_2, ..., d_t\}$，某个症状 s 对候选疾病集 $D'$ 贡献度的期望：

$$\begin{aligned} E(w_{s,D'}) &= \sum_{d_i \in D'} w_{s,d_i} \times p(s|d_1 \vee d_2 \vee ... \vee d_t) + \sum_{d_i \in D} w_{\neg s,d_i} \times p(\neg s|d_1 \vee d_2 \vee ... \vee d_t) \\ &= \sum_{d_i \in D'} w_{s,d_i} \times p(s|d_1 \vee d_2 \vee ... \vee d_t) + \sum_{d_i \in D} w_{\neg s,d_i} \times (1 - p(s|d_1 \vee d_2 \vee ... \vee d_t)) \end{aligned} \tag{4–14}$$

基于公式 4–14, 我们可以计算每个个未确定赋值症状变量对候疾病集合贡献度的期望值，选出期望值最大的症状来询问患者。

## 4.5 本章小结

本章主要介绍了辅助诊断内核的内部架构和各模块的实现机制。辅助诊断内核主要包含四个模块，分别是 KBConverter，ReasoningSolver，QuestionGenerator 以及 DiagnosisCalculator。KBConverter 主要负责根据患者的主述信息从医学知识库中提取相关的医学知识并把他们转换成逻辑规则，其中我们介绍了如何将知识中的关系编码成命题逻辑连接词 (表 4–2)。ReasoningSolver 是推理引擎，是诊断内核的核心模块，它的主要任务是接收 KBConverter 输出的逻辑规则，根据其构建可满足性问题实例来求解，从而排除无关疾病得到候选疾病集。QuestionGenerator 是问题生成器，因为 ReasoningSolver 在推理过程中会需要知道更多的逻辑变量的赋值信息，它会将这些逻辑变量抛给问题生成器处理，问题生成器处理会将这些变量解码成具体症状然后生成相应的问题去询问患者，得到患者的答案后在编码成赋完值的逻辑变量交给 ReasoningSolver 迭代求解。而 DiagnosisCalculator 可以为症状以及疾病打分，为症状打分可以指导 ReasoningSolver 选择包含信息量最多的症状去询问，从而可以用较少的问题就可以得出诊断。在症状打分中，我们分别介绍了朴素、最大熵、最大期望疾病贡献度三种打分方式。而为疾病打分的目的是增加候选疾病的区分度，因为 ReasoningSolver 得出的候选疾病可能很多，通过疾病打分就可以给它们排序，从而最后只输出得分最高的几个疾病，代表可能性最高的几个疾病。

**算法 4–2** 求解症状在疾病中出现次数伪代码

```
Input: 相关知识规则集合 R
Output: 症状在疾病中出现次数的字典
D 为 R 中所有疾病集合;
S 为 R 中所有症状集合;
T 初始化为空字典;
for each s in S do
    for each d in D do
        T[s][d] = 0;
        T[¬s][d] = 0;
    end
end
for each r in R do
    S_l 为 r 左边症状集;
    S_r 为 r 右边症状集;
    D_l 为 r 左边疾病集;
    D_l 为 r 右边疾病集;
    for each s′ in S_l do
        for each d in D_r do
            if s′ 为 s 正出现 then
                T[s][d] += 1/|S_l|;
            else
                T[¬s][d] += 1/|S_l|;
            end
        end
    end
    for each s′ in S_r do
        for each d in D_l do
            if s′ 为 s 正出现 then
                T[s][d] += 1/|S_r|;
            else
                T[¬s][d] += 1/|S_r|;
            end
        end
    end
end
return T;
```

表 4–2 知识关系编码成命题逻辑连接词

Table 4–2 Encoding knowledge relations as propositional logical connectives

| 关系名称 | 命题逻辑连接词 | 例子 |
|---|---|---|
| 强导致 | 蕴含 (→) | 如“（胃炎，强导致，恶心）”可以编码成“胃炎 → 恶心” |
| 弱导致 | 弱导致不是强因果关系，不会对其编码 | —— |
| 不导致 | 导致否 (→ ¬) | 如“(膀胱炎，不导致，高烧)”可以编码为“膀胱 → ¬高烧” |
| 拼接 | 在知识存储时已经将“a 拼接 b”关系合并成一个实体“ab”，所以此关系在知识库中不存在 | —— |
| 与 | 与 (∧) | 如“(流感，强导致，与（发热，流涕))”可以编码成“流感 → 发热 ∧ 流涕” |
| 或 | 或 (∨) | 如“(肺癌，强导致，或（咯血，痰中有血))”可以编码成“肺癌 → 咯血 ∨ 痰中有血” |
| 条件为 | 与 (∧) | 如“((肺癌，条件为，晚期)，强导致，肺部阴影)”可以编码成“肺癌 ∧ 晚期 → 肺部阴影” |
| 调查病史 | 调查病史不是强因果关系，不会对其编码 | —— |
| 是一种 | 蕴含 (→) | 如“(病毒性流感，是一种，流感)”可以编码成“病毒性流感 → 流感” |
| 同义词 | 等价 (↔) | 如“（水肿，同义词，edema）”可以编码为“水肿 ↔ edema” |
| 指代 | 在知识存储时已经将“a 指代 b”中的 a 用 b 替换，所以此关系在知识库中不存在 | —— |
| 然后 | 会将“a 然后 b”关系合并成一个实体“a 然后 b” | 如“(糖尿病，导致，（尿微量蛋白，然后，大量尿蛋白））”可以编码成“糖尿病 → 微量尿蛋白然后大量尿蛋白” |
| 并发症 | 并发症不是强因果关系，不会对其编码 | —— |
| 否定修饰 | 否定 (¬) | 如“(肺炎，强导致，（不，否定修饰，腹痛）)”可以编码成“肺炎 → ¬腹痛” |

# 第五章　知识纠错

## 5.1　Unsat Core

## 5.2　特定结构子图

# 第六章　实验

## 6.1　系统架构

## 6.2　知识库规模

## 6.3　诊断准确率

### 6.3.1　准确率比较

### 6.3.2　变量选择方式对准确率的影响

## 6.4　知识纠错率

# 第七章　总结与展望

# 全文总结

这里是全文总结内容。

2015 年 2 月 28 日，中央在北京召开全国精神文明建设工作表彰暨学雷锋志愿服务大会，公布全国文明城市（区）、文明村镇、文明单位名单。上海交通大学荣获全国文明单位称号。

全国文明单位这一荣誉是对交大人始终高度重视文明文化工作的肯定，是对交大长期以来文明创建工作成绩的褒奖。在学校党委、文明委的领导下，交大坚持将文明创建工作纳入学校建设世界一流大学的工作中，全体师生医护员工群策群力、积极开拓，落实国家和上海市有关文明创建的各项要求，以改革创新、科学发展为主线，以质量提升为目标，聚焦文明创建工作出现的重点和难点，优化文明创建工作机制，传播学校良好形象，提升社会美誉度，显著增强学校软实力。2007 至 2012 年间，上海交大连续三届荣获“上海市文明单位”称号，成为创建全国文明单位的新起点。

上海交大自启动争创全国文明单位工作以来，凝魂聚气、改革创新，积极培育和践行社会主义核心价值观。坚持统筹兼顾、多措并举，将争创全国文明单位与学校各项中心工作紧密结合，着力构建学校文明创建新格局，不断提升师生医护员工文明素养，以“冲击世界一流大学汇聚强大精神动力”为指导思想，以“聚焦改革、多元推进、以评促建、丰富内涵、彰显特色”为工作原则，并由全体校领导群策领衔“党的建设深化、思想教育深入、办学成绩显著、大学文化丰富、校园环境优化、社会责任担当”六大板块共 28 项重点突破工作，全面展现近年来交大文明创建工作的全貌和成就。

进入新阶段，学校将继续开拓文明创建工作新格局，不断深化工作理念和工作实践，创新工作载体、丰富活动内涵、凸显创建成效，积极服务于学校各项中心工作和改革发展的大局面，在上级党委、文明委的关心下，在学校党委的直接领导下，与时俱进、开拓创新，为深化内涵建设、加快建成世界一流大学、推动国家进步和社会发展而努力奋斗！

上海交通大学医学院附属仁济医院也获得全国文明单位称号。

# 附录 A Maxwell Equations

选择二维情况，有如下的偏振矢量：

$$\mathbf{E} = E_z(r,\theta)\hat{\mathbf{z}} \tag{A–1a}$$

$$\mathbf{H} = H_r(r,\theta)\hat{\mathbf{r}} + H_\theta(r,\theta)\hat{\boldsymbol{\theta}} \tag{A–1b}$$

对上式求旋度：

$$\nabla\times\mathbf{E} = \frac{1}{r}\frac{\partial E_z}{\partial\theta}\hat{\mathbf{r}} - \frac{\partial E_z}{\partial r}\hat{\boldsymbol{\theta}} \tag{A–2a}$$

$$\nabla\times\mathbf{H} = \left[\frac{1}{r}\frac{\partial}{\partial r}(rH_\theta) - \frac{1}{r}\frac{\partial H_r}{\partial\theta}\right]\hat{\mathbf{z}} \tag{A–2b}$$

因为在柱坐标系下，$\overline{\overline{\mu}}$ 是对角的，所以 Maxwell 方程组中电场 $\mathbf{E}$ 的旋度：

$$\nabla\times\mathbf{E} = \mathrm{i}\omega\mathbf{B} \tag{A–3a}$$

$$\frac{1}{r}\frac{\partial E_z}{\partial\theta}\hat{\mathbf{r}} - \frac{\partial E_z}{\partial r}\hat{\boldsymbol{\theta}} = \mathrm{i}\omega\mu_r H_r\hat{\mathbf{r}} + \mathrm{i}\omega\mu_\theta H_\theta\hat{\boldsymbol{\theta}} \tag{A–3b}$$

所以 $\mathbf{H}$ 的各个分量可以写为：

$$H_r = \frac{1}{\mathrm{i}\omega\mu_r}\frac{1}{r}\frac{\partial E_z}{\partial\theta} \tag{A–4a}$$

$$H_\theta = -\frac{1}{\mathrm{i}\omega\mu_\theta}\frac{\partial E_z}{\partial r} \tag{A–4b}$$

同样地，在柱坐标系下，$\overline{\overline{\epsilon}}$ 是对角的，所以 Maxwell 方程组中磁场 $\mathbf{H}$ 的旋度：

$$\nabla\times\mathbf{H} = -\mathrm{i}\omega\mathbf{D} \tag{A–5a}$$

$$\left[\frac{1}{r}\frac{\partial}{\partial r}(rH_\theta) - \frac{1}{r}\frac{\partial H_r}{\partial\theta}\right]\hat{\mathbf{z}} = -\mathrm{i}\omega\overline{\overline{\epsilon}}\mathbf{E} = -\mathrm{i}\omega\epsilon_z E_z\hat{\mathbf{z}} \tag{A–5b}$$

$$\frac{1}{r}\frac{\partial}{\partial r}(rH_\theta) - \frac{1}{r}\frac{\partial H_r}{\partial\theta} = -\mathrm{i}\omega\epsilon_z E_z \tag{A–5c}$$

由此我们可以得到关于 $E_z$ 的波函数方程：

$$\frac{1}{\mu_\theta\epsilon_z}\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial E_z}{\partial r}\right) + \frac{1}{\mu_r\epsilon_z}\frac{1}{r^2}\frac{\partial^2 E_z}{\partial\theta^2} + \omega^2 E_z = 0 \tag{A–6}$$

# 附录 B　绘制流程图

图 B–1 是一张流程图示意。使用 tikz 环境，搭配四种预定义节点（`startstop`、`process`、`decision`和`io`），可以容易地绘制出流程图。

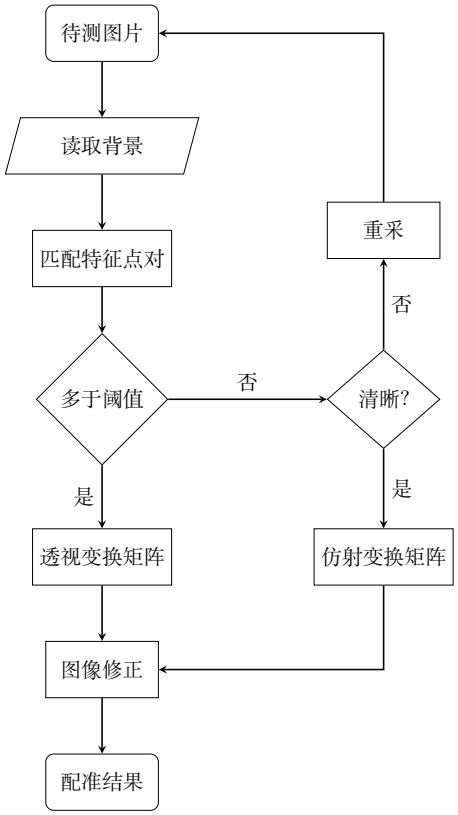



图 B–1 绘制流程图效果

Figure B–1 Flow chart

# 致　谢

感谢那位最先制作出博士学位论文 LaTeX 模板的交大物理系同学！
感谢 William Wang 同学对模板移植做出的巨大贡献！
感谢 @weijianwen 学长一直以来的开发和维护工作！
感谢 @sjtug 以及 @dyweb 对 0.9.5 之后版本的开发和维护工作！
感谢所有为模板贡献过代码的同学们，以及所有测试和使用模板的各位同学！
感谢 LaTeX 和 SJTUThesis，帮我节省了不少时间。

# 攻读硕士学位期间已发表或录用的论文

[1] Chen H, Chan C T. Acoustic cloaking in three dimensions using acoustic metamaterials[J]. Applied Physics Letters, 2007, 91:183518.

[2] Chen H, Wu B I, Zhang B, et al. Electromagnetic Wave Interactions with a Metamaterial Cloak[J]. Physical Review Letters, 2007, 99(6):63903.

# 攻读硕士学位期间参与的项目

[1] 参与 973 项目子课题 (2007 年 6 月–2008 年 5 月)

[2] 参与自然基金项目 (2005 年 5 月–2005 年 8 月)

[3] 参与国防项目 (2005 年 8 月–2005 年 10 月)

# 攻读硕士学位期间申请的专利

[1] 第一发明人，“永动机”，专利申请号 202510149890.0

# 简　历

## 基本情况

某某，yyyy 年 mm 月生于 xxxx。

## 教育背景

- yyyy 年 mm 月至今，上海交通大学，博士研究生，xx 专业
- yyyy 年 mm 月至 yyyy 年 mm 月，上海交通大学，硕士研究生，xx 专业
- yyyy 年 mm 月至 yyyy 年 mm 月，上海交通大学，本科，xx 专业

## 研究兴趣

LaTeX 排版

## 联系方式


- 地址：上海市闵行区东川路 800 号，200240
- E-mail: xxx@sjtu.edu.cn