3专家知识抽取与规则化描述

在上一章中将知识划分为不同类型，根据知识类型规范了知识规则化描述方法的选用。本章介绍确定知识类型后，如何从源知识中抽取所需知识并使用规则化方法存入数据库。首先，根据知识抽取的自动化程度，将抽取分为人工、半自动化和自动化3种方式，介绍了对应的主流方法、工具。其次，以人工抽取方式为主，给出了知识规则化的录入模板，对于规则化结果，阐述了图知识和表知识的存储规范。

3.1知识抽取方式

知识抽取（Knowledge Extraction）研究如何根据给定本体从无语义标注的信息中识别并抽取与本体匹配的事实知识，即从无语义信息的文档内容中抽取与本体匹配的事实知识，进而实现数据充分、有效的利用。该技术可以抽取出事实知识用于构建基于知识的服务。信息抽取的完整度、准确度直接显性影响后续知识图谱构建步骤的质量和效率以及最终知识图谱的质量。

可以将知识抽取描述为这样的一个过程：首先，第一个阶段对大量孤立、模糊、复杂的动态非结构化数据进行初步处理和计算；然后，第二个阶段对数据进行深层语义分析、用户隐私保护问题分析以及应用领域知识的结合分析；最后，第三个阶段选择合适的挖掘算法和抽取技术进行数据抽取和融合。通过将抽取得到的碎片化知识存入知识库的数据层和模式层，我们最终可以对数据形成本体化表达。

抽取技术按照抽取过程可以分为实体抽取（Entity Extraction）、关系抽取（Relation Extraction）、属性抽取（Attribute Extraction）以及实体链接（Entity Linking）等。其中，实体抽取用于发现文本或者网页中的命名实体，并将其加入现有知识库中。关系抽取用于自动抽取实体之间存在的语义关系。属性抽取属于一种特殊的关系抽取。信息抽取的目标是自动化知识获取，即实现自动地从异构数据源中抽取实体、关系、属性等信息进而得到候选知识单元。

从知识抽取过程的自动化程度来看，知识抽取又可以分为人工抽取、半自动抽取和自动抽取三种方式。自动化程度的不同反映了知识抽取系统所具有的推理能力的不同，以下将对这三种抽取方式进行介绍。

3.1.1人工抽取

人工抽取又称为基于规则的方法，主要是由知识工程师与领域专家共同工作、交流来提炼知识、对所需要的知识内容及类型进行定义，即人工编写规则。在提炼出规则之后，知识工程师就可以根据提炼出的规则对大量的结构化数据、半结构化数据和非结构化数据进行处理，最终将其转变为统一格式的结构化数据。最终的结构化数据可以以图或表的形式方便地存储。这种方式首先要先构建大量的实体及关系抽取规则，然后再将规则与文本字符串进行匹配，识别出命名实体及它们之间的关系。

在知识工程师与领域专家交流协作时，可以通过下面几种方法获取到需要的知识信息并编写规则：

（1）面谈法：与专家进行面对面交谈是一种广泛使用的知识获取方式，但是不拘形式的会谈不易获得详细的知识，因此，可采用专题面谈的形式，即向专家提出事先拟好的问题，由专家任意回答。问题可大致分为两种类型，即“在XXX情况下将怎样处理”和“为什么这样做”等。

（2）模拟法：模拟法可分为静态模拟和动态模拟。静态模拟是提出某一实例的情况，请专家谈其求解过程。这一方法肯定会得到一套资料，说明其求解过程所用的知识和步骤。它的优点是，因系静态条件下进行，可以集中到我们最感兴趣的方面讨论。它的缺点是，工作压力与时间压力与实际情况不同，且可能忽略某些细节。动态模拟是在专家处理某真实问题时，知识工程师观察并录下其实际求解步骤，然后再进行分析。这种方法的优点是能够观察到专家在自然状态下的工作过程，缺点是较为费时费力。

（3）口语记录分析：当专家解决问题时，让专家自己叙述自己想些什么，并加以笔录或录音，然后对记录加以分析，即对专家的思维活动做出叙述性的记录。在口语记录分析中主要处理两类问题：一种是分析表达明显的部分，即从口语记录明显看出专家思维过程的部分；一种是隐含表达的部分，即对专家没有明确表达的思维过程，需要分析者根据记录加以推断，他的某些语句时基于什么背景得出的。只要专家充分合作，知识工程师有足够的经验和知识，口语记录分析就是一种可靠的知识获取方法。当然在关键地方还可以使用面谈法加以考核，以防失真。

三种方法的特点及其适用范围见下表3-1：

表3-1 获取知识信息的三种方法对比

|  |  |  |
| --- | --- | --- |
| 方式 | 特点 | 适用范围 |
| 面谈法 | 知识工程师与专家面对面交流以获取想要了解的信息 | 适用于知识工程师与专家一起工作或不受时间和空间的限制时 |
| 模拟法 | 知识工程师提出问题让专家求解或是知识工程师观察专家的工作过程 | 适用于知识工程师能够贴近真实工作情况时 |
| 口语记录分析 | 知识工程师记录下专家解决问题时的思路 | 知识工程师有充足的经验和知识，专家充分合作 |

在构建出规则之后，可以按照以下步骤进行知识抽取：

（1）对构建出的规则进行解析；

（2）对句子进行分词、词性标注、命名实体识别、依存分析等处理，并构建出依存树；

（3）在句子依存语法树上匹配规则，每匹配一条规则就生成一个三元组；

（4）根据扩展规则对抽取到的三元组进行扩展；

（5）对三元组实体和触发词进一步抽取出各个实体之间的关系。

其流程图如下图3-1所示：



图3-1 人工知识抽取流程图

早期的知识抽取大都采用这种方式，在小数据集上可以达到很高的准确率和召回率，所得到的知识大多可以直接用于解决问题，但是不适合用于大数据集。对于风洞一体化平台中的数据即可采用此种抽取方式，能够取得较好的效果。

3.1.2半自动抽取

半自动知识抽取指利用一些知识获取工具来辅助知识工程师把知识原材料或专家描述的知识内容经过识别、理解、筛选、格式化后以一定的形式存入知识库中。其主要包括以下6个步骤，流程图如下图3-2所示：

（1）用一组信息模式描述感兴趣的知识，如竞争情报采集与监测系统中常用到的“推出<公司，产品>”；

（2）对数据进行预处理，包括格式清洗与转换；

（3）对数据进行词法分析、浅层句法分析以及简单的语义分析，识别名词短语并标注语义信息；

（4）使用模式匹配方法实现事件模板的构造，建立实体之间的关系；

（5）采用语段分析技术实现句子相关性分析，进行上下文关联、指代消解、引用等分析和推理，构造一个完整的实体事件；

（6）格式化分析结果，把抽取的信息按预定义的模板输出。

形状

低可信度描述已自动生成

图3-2 半自动知识抽取流程图

在半自动知识抽取中，需要要对海量的数据进行预处理，这是一项人力所不能完成的任务，因此需要依靠计算机的协助。自从科学计量学分析引入辅助可视化以来，科学知识图谱领域也诞生了一批有价值的科学知识图谱工具。表3-2列出了一些常用的知识图谱辅助工具。

表3-2 常用知识图谱辅助工具

|  |  |  |
| --- | --- | --- |
| 工具名称 | 特点 | 适用范围 |
| BibExcel | 可对数据格式转换及去噪，并进行BCAD、CAAA、CAAC、ACA、DCA、CWA等分析 | 适合用于简单的计量分析以及为一些可视化工具提供辅助， 如统计、构建共现矩阵等 |
| CiteSpace | 可对数据进行去重和时间切片，并进行BCAD、CAAA、CCAA、CAAI、ACA、DCA、JCA等分析 | 适合用于时间序列分析、突发监测 |
| SPSS | 可用于相关分析、因子分析、主成分分析、多维尺度分析、聚类分析等 | 适合用于分析偏小、分布均匀的数据集 |
| TDA | 可用于聚类分析、关联分析、时间序列、突发监测等 | 适合用于数据清洗、构建矩阵，为SPSS和Ucinet等一些可视化工具提供支持 |
| Ucinet | 可用于中心性分析、子群分析、角色分析、聚类分析等 | 适合用于分析较大的数据集、分析核心关键词 |

使用相关工具来生成文献分析的可视化图谱固然重要，但认识所分析的数据内容及其结构是我们进行图谱绘制的前提。因此，还可以使用Notepad++和Sublime Text这两个工具快速打开所分析文本，并查看其内容和结构。

采用半自动抽取时可以把文档当作字符串进行训练，如果文档是比较结构化的还可以提取出结构化文档中的树结构去训练。

使用半自动抽取在学习到抽取规则后可以有较好的效果，但是手工标注训练数据代价高昂，不仅要标注一定数量的训练数据，而且要选择适当的训练数据，会比较繁琐。因此这种方式只适合在数据量较小或数据格式统一时使用，在数据量较大且数据格式不统一时较为麻烦。对于风洞一体化平台，如果要使用半自动抽取方式需要预先对数据进行处理，较为麻烦且易出错，因此可避开这种方式。

3.1.3自动抽取

自动抽取利用数据挖掘、机器学习、深度学习等人工智能技术通过对应用实例与实际问题进行建模从而发现一些专家尚未形式化甚至未发现的新知识和新规律，从信息源中自动提取出实体、属性和关系。

近年来，基于预训练模型通过微调进行学习的方法在自然语言处理领域取得了巨大成功，Google的BERT模型就是其中的重要代表。

BERT（Bidirectional Encoder Representations from Transformers）是一种基于转换器（Transformer）的双向编码表示模型，它的拓扑结构是一个多层的双向转换器网络。BERT模型是基于微调学习的典型应用，也就是说他的构建包含有预训练和微调两个步骤。

首先在预训练阶段，对大量不同训练任务的未标记语料数据进行训练，将语料中的知识迁移进了预训练模型的文本嵌入（Embedding）中。这样，在微调阶段，只需要在神经网络中增加一个额外的输出层，就可以对预训练模型进行调整了。具体而言，微调就是用预训练参数初始化BERT模型，然后，使用来自下游任务的标记数据对模型进行微调。

如图3-3所示是使用BERT从保险文档中抽取知识点的模型流程。



图3-3 使用BERT从保险文档中抽取知识点

在自动抽取知识的时候，首先要使用大量的训练数据即训练语料对机器学习或深度学习模型进行训练，在训练完成之后再去使用训练好的模型抽取新的知识。

这种方式构建出模型的代价较小，鲁棒性更好，但是训练语料的数量和质量直接影响着这种方法的抽取性能，知识畸变（言语描述和具体行动间形成的错位现象）也影响着知识获取的精准性，自动抽取方式还未成熟。基于以上种种原因，自动抽取方式还不适合实际应用。

以上三种抽取方式的对比见表3-3。

表3-3 三种抽取方式对比

|  |  |  |  |
| --- | --- | --- | --- |
| 抽取方式 | 方式 | 特点 | 适用条件 |
| 人工抽取 | 人工编写规则，手动分析、整理 | 不需要训练模型，简单方便 | 适用于小数据集 |
| 半自动抽取 | 手工标注数据后使用机器学习方法学习规则 | 节省人力 | 适用于中等规模数据集，能够手工标注数据且选择合适的训练数据 |
| 自动抽取 | 使用人工智能技术进行建模从而抽取知识 | 自动化程度高、建模代价较小、鲁棒性高但是技术还不成熟 | 适用于大数据集且自动化程度要求较高时 |

3.3存储规范

知识的规则化输出结果主要是以图和表的形式，图知识和表知识有着不同的特性及结构，因此需要对它们分别设置不同的存储方案。以下对这两种形式知识的存储进行说明。

3.3.1 图知识的存储

图知识中有以下几个基本概念：图（Graph）指关系图。比如：同学及朋友关系图、银行转账图等；顶点（Vertex）一般指实体。比如：人、账户等；边（Edge）一般指关系。比如：朋友关系、转账动作等；属性（Property）顶点或边可以包含属性，比如：人的姓名、人的年龄、转账的时间。其逻辑形状如下图3-10所示。



图3-10 图知识逻辑形状

在图知识中最重要的信息就是图的节点、边的信息和边之间的关系，可以使用关系型数据库或图数据库来存储这些信息，具体如下。

图数据库是以图论为基础的一种非关系型数据库，其数据库存储结构和数据的查询方式都是以图论为基础的。图论中图的基本元素为节点和边，在图数据库中对应的就是节点和关系。图数据库的关注点是“关联关系”形成的图，其目标是对现实世界中的实体与实体之间的关联关系进行存储与分析：将实体抽象为顶点、将实体之间的关联关系抽象为边。通过顶点和边形成的图谱结构，直观自然地表达万物关联的世界，同时解决了复杂关联关系深层检索的性能问题。

图数据库仅使用一个模式和一组起点，就可以围绕这些初始起点探索相邻数据，收集和汇总来自数百万个节点和关系的信息，并保持搜索范围之外的任何数据不变。图数据库在处理规模大、关联度高的数据时优势明显。与传统的关系型数据库相比，图数据库具有如下优势：

（1）可以很自然地表达现实世界中的实体及其关联关系（对应图的顶点及边）；

（2）灵活的数据模型可以适应不断变化的业务需求；

（3）灵活的图查询语言，轻松实现复杂关系网络的分析；

（4）关系型数据库在遍历关系网络并抽取信息的能力非常弱，图数据库则为此而生；

（5）关系型数据库在规模庞大时很难做多层关联关系分析（Join操作往往消耗过长时间而失败），图数据库则天然把关联数据连接在一起，无需耗时耗内存的Join操作，可以保持常数级时间复杂度；

（6）在关系型数据库的查询中，关系的增多意味着需要遍历更多的关联表与实体表，其中存在着大量需要遍历的数据。并且，关系型数据库在数据查询时遍历的大量内容都与查询内容毫无关系，因此这也会导致关系查询时效率的大大降低；

（7）在进行数据查询时，图数据库会从查询目标的临近节点开始，基于图结构进行查询，而不是对整个类别的数据进行遍历。因此，在查询内容不变的情况下，数据规模的不断增大并不会对其查询性能产生明显影响；

（8）图数据库多用于网络安全、金融风控、知识图谱、广告推荐、社交网络、智能机器人、物联网等领域以及关联分析、路径搜索、子图挖掘、特征发现、社区聚类等应用。其主流的查询语言包括：Gremlin、Cypher、SPARQL，典型的存储结构有边集数组、邻接矩阵、邻接表和十字链表等。

综上所述，使用图数据库存储图数据能够产生更高的效率及性能。在使用图数据库存储图数据时，可以使用当前流行的图数据库Neo4j。Neo4j是一个由Java实现的开源图数据库，有效地将属性图模型实现到了存储级别，还提供了完整的数据库特性。它包括如下几个显著特点：完整的ACID支持、高可用性、轻易扩展到上亿级别的节点和关系、通过遍历工具高速检索数据。Neo4j确保了在一个事务里面的多个操作同时发生，保证数据的一致性。Neo4j把其数据库文件分为四大类来分类存储，分别是标签、节点、属性和关系。除此之外，Neo4j还提供了一个用户友好的Web页面，可以进行各项配置、写入、查询等操作，并且提供了可视化功能。Cypher是Neo4j的声明式图形查询语言，允许用户不必编写图形结构的遍历代码，就可以对图形数据进行高效的查询。Cypher的设计目的类似SQL，适合于开发者以及在数据库上做点对点模式（ad-hoc）查询的专业操作人员。其具备的能力包括：创建、更新、删除节点和关系；通过模式匹配来查询和修改节点和关系；管理索引和约束等。以上种种原因使得Neo4j最适合用于完整的企业部署或者用于一个轻量级项目中完整服务器的一个子集存在。

使用Neo4j图数据库存储图知识的步骤也非常简单方便：可以直接使用Cypher load csv语句将数据转换成CSV格式后通过LOAD CSV读取数据，也可以使用官方提供的neo4j-import工具进行批量导入存储。除了这两种方式之外，还可以使用官方提供的API或是一些别的导入工具进行存储。

3.3.2 表知识的存储

表知识中的各种数据被划分为一个个字段，每条数据对于同一个字段可能会有着不同的值，在存储表知识时可以使用关系型数据库。

关系型数据库（Relational Database）是建立在关系模型基础上的数据库，借助于集合代数等数学概念和方法来处理数据库中的数据。现实世界中的各种实体以及实体之间的各种联系均用关系模型来表示。关系型数据库以行和列的形式存储数据，用户通过查询来检索数据库中的数据，而查询是一个用于限定数据库中某些区域的执行代码。关系模型可以简单理解为二维表格模型，而一个关系型数据库就是由二维表及其之间的关系组成的一个数据组织。

关系型数据库具有如下几个特点：

存储方式：传统的关系型数据库采用表格的储存方式，数据以行和列的方式进行存储，要读取和查询都十分方便。

存储结构：关系型数据库按照结构化的方法存储数据，每个数据表都必须对各个字段定义好（也就是先定义好表的结构），再根据表的结构存入数据，这样做的好处就是由于数据的形式和内容在存入数据之前就已经定义好了，所以整个数据表的可靠性和稳定性都比较高，但带来的问题就是一旦存入数据后，如果需要修改数据表的结构就会十分困难。

存储规范：关系型数据库为了避免重复、规范化数据以及充分利用好存储空间，把数据按照最小关系表的形式进行存储，这样数据的管理就可以变得很清晰、一目了然。

扩展方式：由于关系型数据库将数据存储在数据表中，数据操作的瓶颈出现在多张数据表的操作中，而且数据表越多这个问题越严重，如果要缓解这个问题，只能提高处理能力，也就是选择速度更快性能更高的计算机，这样的方法虽然可以一定的拓展空间，但这样的拓展空间一定有非常有限的，也就是关系型数据库只具备纵向扩展能力。

查询方式：关系型数据库采用结构化查询语言（即SQL）来对数据库进行查询，SQL早已获得了各个数据库厂商的支持，成为数据库行业的标准，它能够支持数据库的CRUD（增加，查询，更新，删除）操作，具有非常强大的功能，SQL可以采用类似索引的方法来加快查询操作。

规范化：在数据库的设计开发过程中开发人员通常会面对同时需要对一个或者多个数据实体（包括数组、列表和嵌套数据）进行操作，这样在关系型数据库中，一个数据实体一般首先要分割成多个部分，然后再对分割的部分进行规范化，规范化以后再分别存入到多张关系型数据表中，这是一个复杂的过程。随着软件技术的发展，相当多的软件开发平台都提供一些简单的解决方法，例如，可以利用ORM层（也就是对象关系映射）来将数据库中对象模型映射到基于SQL的关系型数据库中去以及进行不同类型系统的数据之间的转换。

事务性：关系型数据库强调ACID规则（原子性（Atomicity）、一致性（Consistency）、隔离性（Isolation）、持久性（Durability）），可以满足对事务性要求较高或者需要进行复杂数据查询的数据操作，而且可以充分满足数据库操作的高性能和操作稳定性的要求。并且关系型数据库十分强调数据的强一致性，对于事务的操作有很好的支持。关系型数据库可以控制事务原子性细粒度，并且一旦操作有误或者有需要，可以马上回滚事务。

读写性能：关系型数据库十分强调数据的一致性，并为此降低读写性能付出了巨大的代价，虽然关系型数据库存储数据和处理数据的可靠性很不错，但一旦面对海量数据的处理的时候效率就会变得很差，特别是遇到高并发读写的时候性能就会下降的非常厉害。

在关系型数据库中，数据由不同的表组成，每个表包含有不同的字段，还可以对各个字段施加多种限制，其结构如下表3-9所示。

表3-9 关系型数据表结构

|  |  |  |
| --- | --- | --- |
| 字段1 | 字段2 | 字段3 |
| A | E | I |
| B | F | J |
| C | G | K |

在使用关系型数据库存储图数据时，可以使用当前流行的MySQL数据库管理系统。在Web应用方面，MySQL是最好的RDBMS （Relational Database Management System，关系数据库管理系统） 应用软件之一。MySQL所使用的SQL语言是用于访问数据库的最常用标准化语言。MySQL软件采用了双授权政策，分为社区版和商业版，由于其体积小、速度快、总体拥有成本低，尤其是开放源码这一特点，一般中小型网站的开发都选择MySQL作为网站数据库。

使用MySQL数据库存储表知识也是非常方便快速：提前对表知识进行一些规范化处理后将其存储为CSV文件、EXCEL文件等格式，然后可以使用各种各样的数据库管理工具从这些文件中批量导入数据即可。

3.3.3 存储工具推荐

综上所述，图知识和表知识有着不同的特性和结构，其存储方式、存储工具等都不相同，对它们的存储既要符合它们独特的数据结构，又要能够在查询时方便地进行检索。关于图知识和表知识的特点、可使用的存储工具及建议使用的较成熟的数据库工具如下表3-10所示：

表3-10 图知识和表知识特点及建议存储工具

|  |  |  |  |
| --- | --- | --- | --- |
| 数据类型 | 特点 | 可使用的存储工具 | 建议使用 |
| 图知识 | 数据以节点和边的形式展现出来 | 图数据库、关系型数据库 | Neo4j图数据库 |
| 表知识 | 数据以二维表格的形式展现出来 | 关系型数据库 | MySQL数据库 |

3.3.4 存储命名规范

不论是在图数据库中存储图知识还是在关系型数据库中存储表知识，都应该对其中的实体、关系或字段名称进行约束，以保证全局数据的一致性并降低后期对知识的管理难度。本小节对数据库中实体、关系及字段的命名进行规范。

对于使用关系型数据库存储表知识中相关字段的命名可以采取

3.4知识规则化示例

本部分针对风洞基地的一个故障案例——“水泵运转不上水”，基于知识图谱将其中涉及的知识规则化，对知识之间的联系进行分析和描绘。案例主要描述如下：

故障：水泵运转不上水。

时间：2019年6月14日；故障发现人：杨某。

故障/事故描述：试验运行前，打开为热阀供水水泵，水泵运转起来，水泵出口压力不上升。

检查经过：拆解水泵底阀，发现底阀封面有沙粒，底阀密封不严。

发生原因：水泵长期使用，水质变差，水中杂质寄存于底阀密封面，造成底阀无法密封，水泵抽吸为抽空状态，导致无法上水。

依据知识图谱的应用流程，从上述案例中进行知识抽取，得到相关实体、关系、属性以及属性值，即三元组（实体，关系，实体）或（实体，属性，属性值）的元素，如表3-11所示。用箭头将所有三元组连接起来，即可得到如图3-11所示的水泵故障知识图谱。

表3-11 知识抽取结果

|  |  |
| --- | --- |
| 类型 | 结果 |
| 实体 | 水泵、杨某等 |
| 关系 | 故障结果、措施等 |
| 属性 | 发现时间 |
| 属性值 | 2019年6月14日 |



图3-11 水泵故障知识图谱

在得到上图中的知识图谱后，可以使用图数据库Neo4j将其存储起来。Neo4j使用Cypher查询语言对数据库中图数据进行处理，能够方便地构建、查询图知识。上图中的知识图谱可使用下列Cypher语句进行构建：

create

(:杨某{name:"杨某"}),

(:运转不上水{name:"运转不上水"}),

(运转不上水)-[:发现人]->(杨某),

(:更换水体{name:"更换水体"}),

(运转不上水)-[:措施]->(更换水体),

(:清除阀门杂质{name:"清除阀门杂质"}),

(运转不上水)-[:措施]->(清除阀门杂质),

(:时间{name:"2019年6月14日"}),

(运转不上水)-[:发现时间]->(时间),

(:水泵{name:"水泵"}),

(水泵)-[:故障结果]->(运转不上水),

(:出口压力不升{name:"出口压力不升"}),

(水泵)-[:故障描述间]->(出口压力不升),

(:拆解底阀{name:"拆解底阀"}),

(水泵)-[:检查方式]->(拆解底阀),

(:底阀无法密封{name:"底阀无法密封"}),

(出口压力不升)-[:原因]->(底阀无法密封),

(:水中杂质{name:"水中杂质"}),

(底阀无法密封)-[:原因]->(水中杂质)

在Neo4j数据库中使用以上Cypher语句构建之后产生的图知识逻辑结构如下图3-12所示，之后如果需要对知识图谱中内容进行查询或修改即可使用Cypher语句快速完成。

图表

描述已自动生成

图3-12 在Neo4j数据库中的水泵故障知识图谱