



自然语言处理导论

张奇 桂韬 黄萱菁

2023 年 2 月 12 日

数学符号

数与数组

α	标量
α	向量
A	矩阵
\mathbf{A}	张量
I_n	n 行 n 列单位矩阵
v_w	单词 w 的分布式向量表示
e_w	单词 w 的独热向量表示: $[0,0,\dots,1,0,\dots,0]$, w 下标处元素为 1

索引 |

α_i	向量 α 中索引 i 处的元素
α_{-i}	向量 α 中除索引 i 之外的元素
$w_{i:j}$	序列 w 中从第 i 个元素到第 j 个元素组成的片段或子序列
A_{ij}	矩阵 A 中第 i 行、第 j 列处的元素
$A_{i:}$	矩阵 A 中第 i 行
$A_{:j}$	矩阵 A 中第 j 列
A_{ijk}	三维张量 \mathbf{A} 中索引为 (i, j, k) 处元素
$\mathbf{A}_{::i}$	三维张量 \mathbf{A} 中的一个二维切片

集合

\mathbb{A}	集合
\mathbb{R}	实数集
\mathbb{C}	复数集
$\{0, 1, \dots, n\}$	含 0 和 n 的正整数的集合
$[a, b]$	a 到 b 的实数闭区间
$(a, b]$	a 到 b 的实数左开右闭区间

线性代数

\mathbf{A}^\top	矩阵 \mathbf{A} 的转置
$\mathbf{A} \odot \mathbf{B}$	矩阵 \mathbf{A} 与矩阵 \mathbf{B} 的 Hadamard 乘积
$\det \mathbf{A}^\top$	矩阵 \mathbf{A} 的行列式
$[x; y]$	向量 x 与 y 的拼接
$[\mathbf{U}; \mathbf{V}]$	矩阵 \mathbf{A} 与 \mathbf{V} 沿行向量拼接
$x \cdot y$ 或 $x^\top y$	向量 x 与 y 的点积

微积分

$\frac{dy}{dx}$	y 对 x 的导数
$\frac{\partial y}{\partial x}$	y 对 x 的偏导数
$\nabla_{\mathbf{x}} y$	y 对向量 \mathbf{x} 的梯度
$\nabla_{\mathbf{X}} y$	y 对矩阵 \mathbf{X} 的梯度
$\nabla_{\mathbf{X}} y$	y 对张量 \mathbf{X} 的梯度

概率与信息论

$a \perp b$	随机变量 a 与 b 独立
$a \perp b \mid c$	随机变量 a 与 b 关于 c 条件独立
$P(a)$	离散变量概率分布
$p(a)$	连续变量概率分布
$a \sim P$	随机变量 a 服从分布 P
$\mathbb{E}_{x \sim P}(f(x))$ 或 $\mathbb{E}(f(x))$	$f(x)$ 在分布 $P(x)$ 下的期望
$\text{Var}(f(x))$	$f(x)$ 在分布 $P(x)$ 下的方差
$\text{Cov}(f(x), g(x))$	$f(x)$ 与 $g(x)$ 在分布 $P(x)$ 下的协方差
$H(f(x))$	随机变量 x 的信息熵
$D_{KL}(P \parallel Q)$	概率分布 P 与 Q 的 KL 散度
$\mathcal{N}(\mu, \Sigma)$	均值为 μ 、协方差为 Σ 的高斯分布

数据与概率分布

\mathbb{X} 或 \mathbb{D}	数据集
$x^{(i)}$	数据集中第 i 个样本（输入）
$y^{(i)}$ 或 $y^{(i)}$	第 i 个样本 $x^{(i)}$ 的标签（输出）

函数

$f : \mathcal{A} \rightarrow \mathcal{B}$	由定义域 \mathcal{A} 到值域 \mathcal{B} 的函数（映射） f
$f \circ g$	f 与 g 的复合函数
$f(\mathbf{x}; \boldsymbol{\theta})$	由参数 $\boldsymbol{\theta}$ 定义的关于 \mathbf{x} 的函数（也可以直接写作 $f(\mathbf{x})$ ，省略 $\boldsymbol{\theta}$ ）
$\log x$	x 的自然对数函数
$\sigma(x)$	Sigmoid 函数 $\frac{1}{1 + \exp(-x)}$
$\ \mathbf{x}\ _p$	\mathbf{x} 的 L^p 范数
$\ \mathbf{x}\ $	\mathbf{x} 的 L^2 范数
$\mathbf{1}^{\text{condition}}$	条件指示函数：如果 condition 为真，则值为 1；否则值为 0

本书中常用写法

- 给定词表 \mathbb{V} ，其大小为 $|\mathbb{V}|$
- 序列 $\mathbf{x} = x_1, x_2, \dots, x_n$ 中第 i 个单词 x_i 的词向量 \mathbf{v}_{x_i}
- 损失函数 \mathcal{L} 为负对数似然函数： $\mathcal{L}(\boldsymbol{\theta}) = -\sum_{(x,y)} \log P(y|x_1 \dots x_n)$
- 算法的空间复杂度为 $\mathcal{O}(mn)$

目 录

12 知识图谱	1
12.1 知识图谱概述	1
12.1.1 知识图谱发展历程	3
12.1.2 知识图谱研究内容	4
12.2 知识图谱表示与存储	6
12.2.1 知识图谱的符号表示	6
12.2.2 知识图谱的向量表示	9
12.2.3 基于表的知识谱图谱存储	12
12.2.4 基于图的知识谱图谱存储	15
12.3 知识图谱获取与构建	18
12.3.1 属性补全	19
12.3.2 实体链接	22
12.3.3 实体对齐	25
12.4 知识图谱推理	31
12.4.1 基于符号逻辑的知识图谱推理	32
12.4.2 基于表示学习的知识图谱推理	35
12.5 知识图谱问答	39
12.5.1 基于语义解析的知识图谱问答	40
12.5.2 基于信息检索的知识图谱问答	42
12.5.3 基于深度神经网络的知识图谱问答	46
12.5.4 知识图谱问答语料库	50
12.6 延伸阅读	51
12.7 习题	52

12. 知识图谱

知识图谱（Knowledge Graph）是谷歌公司于 2012 首次提出的概念，但是其研究历史可以追溯到费根鲍姆教授（B.A.Feigenbaum）在 1977 年提出的知识工程（Knowledge Engineering）以及 20 世纪 90 年代后期开始的语义网（Semantic Web）。知识工程、语义网以及知识图谱都属于人工智能中知识这一核心命题。知识图谱并不是单一技术的研究，而是一个系统工程，其研究内容涵盖知识表示、知识存储、知识推理、图谱构建、图谱问答等方面，还涉及自然语言处理、机器学习、图数据库、逻辑推理等多个交叉领域。自然语言和知识密切关联，知识是实现计算机对自然语言真正理解必不可少的关键部分。知识图谱在自然语言处理中也发挥着越来越重要的作用，是智能问答、语义检索、机器翻译、语义表示等任务的重要基础。

本章首先介绍知识图谱的基本概念和发展历程，在此基础上介绍知识图谱构建，知识图谱推理、基于知识图谱的问答以及知识图谱存储。

12.1 知识图谱概述

知识图谱（Knowledge Graph, KG）是指采用图结构表示实体（包括物体、事件或抽象概念）及其之间关系的知识库（Knowledge Base）。在知识图谱中，图的节点表示实体，图的边表示实体之间的关系。图12.1给出了以金庸为中心的知识图谱示例，图中的节点包含人名等实体，也包含职业等抽象概念，节点之间的边表示了实体之间的各类关系。利用知识图谱可以构建医学、生物、金融、化学等各类型领域知识，一些常识知识也可以使用知识图谱进行表示和利用。知识图谱的目标是利用图结构对知识进行表示，在此基础上识别和推断事物之间的复杂关系并沉淀各类型知识。

相比于直接利用自然语言文本中所包含的知识，计算机算法更容易利用基于图结构所表示的知识。2012 年谷歌公司在提出知识图谱概念的同时，发布了其知识搜索产品。目前绝大部分搜索引擎都将知识图谱作为重要的底层支撑技术，对于很多用户查询，也采用了基于知识图谱的问答算法直接回答用户问题。如图12.2所示，针对“金庸和徐志摩是什么关系？”的问题，搜索引擎直接通过知识图谱给出了“表兄弟”关系，并通过生日计算得到了他们的年纪差。这种基于知识图谱的问答大幅度提升了用户使用搜索引擎的体验。

知识图谱按照其所描述的主要内容可以分为四类：事实知识图谱，概念知识图谱，语言知识

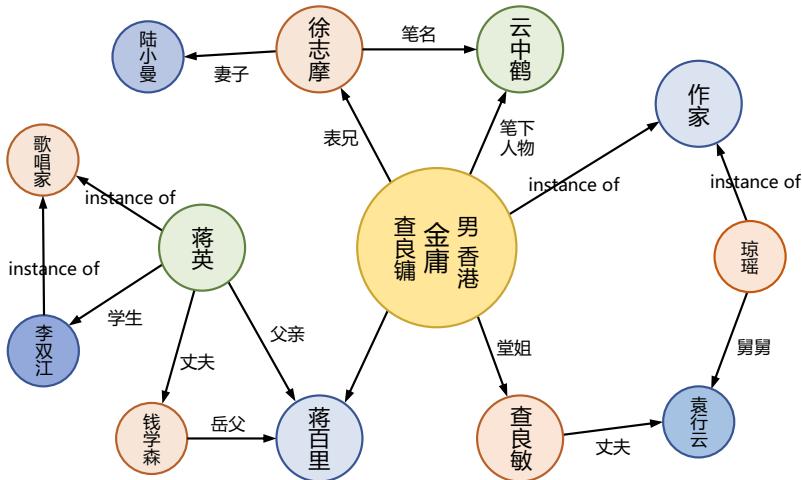


图 12.1 知识图谱样例



图 12.2 基于知识图谱的问题回答样例（来源：搜狗搜索）

图谱和常识知识图谱。

- (1) 事实知识图谱包含了现实世界中各实体之间的关系，比如：“梅西”出生于“阿根廷”。
- (2) 概念知识图谱描述概念之间的子类关系，比如：“老虎”是“哺乳动物”，“足球运动员”属于“运动员”。
- (3) 语言知识图谱是人类语言中蕴含的词法、句法、语义和语用等知识，比如：WordNet、Hownet都可以认为是一种词法知识图谱。
- (4) 常识知识图谱描述人类与世界交互积累的经验与知识，比如：“猫爱吃鱼”、“冬天可能会下雪”。

知识图谱一般由<头实体，关系，尾实体>组成的三元组为基本元素构成。实体一般为世界上的具象事物或者抽象概念，而实体之间的联系则定义为关系。以“刘备是刘禅的父亲”的世界知

识为例，知识图谱将其存储为三元组 \langle 刘备，父亲，刘禅 \rangle ，其中“刘备”为头实体，“刘禅”为尾实体，而“父亲”则为关系。通过扩展这样的三元组，最终会形成一张巨大的知识网络，网络的节点为实体，边则为实体之间的关系，这样的知识网络即为知识图谱。知识图谱可以形式化的定义为：

$$\mathcal{G} = \{\mathcal{E}, \mathcal{R}, \mathcal{F}\} \quad (12.1)$$

其中 \mathcal{E} 表示为实体集合， \mathcal{R} 表示为关系集合，而 \mathcal{F} 表示为事实集合。知识图谱中的一个事实即为上文中提到的三元组 $\langle h, r, t \rangle \in \mathcal{F}$ ， h 表示头实体， r 表示关系， t 表示尾实体， $h, t \in \mathcal{E}$ ， $r \in \mathcal{R}$ 。

12.1.1 知识图谱发展历程

虽然知识图谱的概念在 2012 年才被首次提出，但是关于知识表示和知识推理的研究一直贯穿于整个自然语言处理和人工智能的发展过程中。自 20 世纪 70 年代中期开始，人工智能领域开始逐渐认识到知识在智能系统的重要性，并提出知识工程^[1]的概念，自此知识表示、知识获取、知识推理等知识相关研究就成为了人工智能领域的研究重点和难点。

20 世纪 80 年代开始哲学领域的本体（Ontology）概念引入人工智能^[2]，是指用规范化方法描述概念、实体、术语及其相互关系。同一时期，研究人员们也提出了语义网络（Semantic Network 或 Semantic Net）^[3] 用于形式化地表示知识。语义网使用有向或者无向图中的顶点表示概念，边表示概念之间的语义关系。1993 年 Gruber 关于本体给出了更通用定义，将本体定义为对某一智能体（Agent）或智能体群体中存在的概念和关系的一种描述^[4]。在这期间，企业和学术界构造了包括 CYC、WordNet、BFO（Basic Formal Ontology）、HowNet 等在内的大量通用和领域本体库。

1998 年 Tim Berners-Lee 提出了语义网（Semantic Web）概念^[5]，其核心是通过定义标准的标志语言（Markup Language）和构造相关处理工具，对互联网上的文档添加能够被计算机所理解的元数据（Meta Data），从而扩展互联网，使之成为一个通用的信息交换媒介。语义网通过可扩展标记语言（XML）、资源描述框架（RDF）以及本体等规范化的描述体系和结构定义来构建语义表达框架。从大数据的视角，语义网也可以称为关联数据（Linked Data），通过可链接的 URI 来发布、共享、连接互联网中各类数据，构建语义数据网络，如图12.3所示。在语义网提出后的十几年里，出现了大量的语义网项目，包括 Freebase、LinkingOpenData、WikiData 等。我国在语义网方面也开展了大量研究，OpenKG 项目也收录了大量中文语义网开放数据集。

2012 年谷歌公司以“Things, Not Strings!”为主题发布了知识图谱搜索产品，希望解决用户使用传统搜索引擎需要阅读文章并自行寻找答案，即字符串（Strings）级别搜索。试图构建事物（Things）对象级别搜索，通过构建大规模结构化的事物精确描述以及事物之间的关联，使得用户可以直接得到答案或对象级精准搜索。近十年来，知识图谱快速发展，大量超大规模的知识图谱陆续发布。百度在 2020 年发布的多源异构中文知识图谱，覆盖超过 50 亿实体和 5500 亿事实。阿里巴巴构建的数字商业知识图谱 AliOpenKG，包含 1600 万实体、67 万的核心概念、2681 类关系以及 18 亿三元组。美团构建的餐饮娱乐知识图谱包含 23 类概念、16 亿实体、486 亿三元组。谷歌的知识图谱

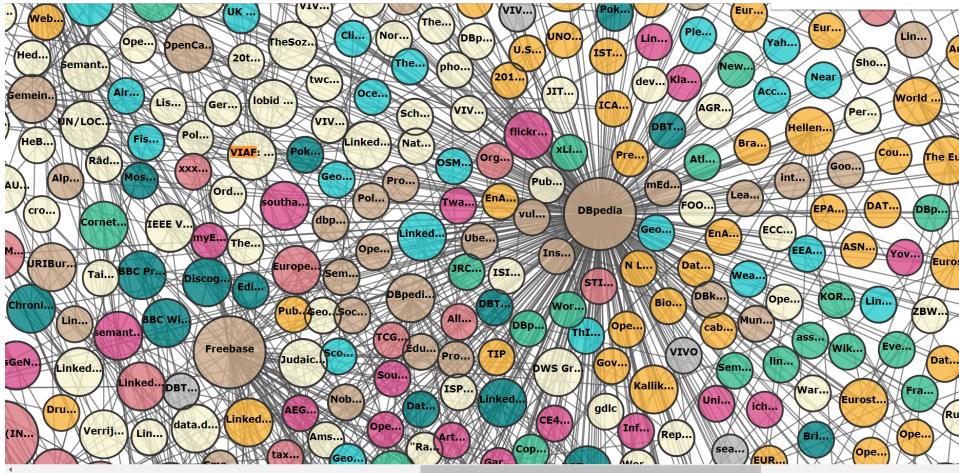


图 12.3 数据关联网络样例

也从 2021 年发布时的 5 亿实体，35 亿关系，增长到 2020 年包含 50 亿实体，5000 亿关系^[6]。知识图谱也已经成为了搜索引擎、推荐系统、智能问答等应用系统中不可获取的基础组件。

12.1.2 知识图谱研究内容

知识图谱研究涉及多个领域，是典型的交叉领域研究。如图12.4所示，知识图谱涉及到机器学习、自然语言处理、数据库、图算法等领域。知识图谱不仅在构建、推理等阶段需要使用机器学习算法，同时知识谱图也可以与机器学习算法结合，在基于知识的特征、基于知识图谱的嵌入表示、图神经网络等方面与基于特征的机器学习方法以及神经网络方法紧密结合。在自然语言处理方面，知识图谱的构建离不开实体识别、关系抽取、事件抽取等自然语言处理中的信息抽取技术，同时知识图谱作为重要的底层支撑，也是自然语言处理中智能问答、机器翻译等任务不可或缺的组成部分。近年来，知识图谱在预训练语言模型方面也发挥了越来越重要的作用，包括清华大学 ERNIE^[7]、百度 ERNIE^[8]、LUKE^[9] 等在内的预训练语言模型都在不同层面使用了知识图谱。大规模知识图谱的广泛应用，知识图谱的存储和检索问题也与数据库研究的产生了交叉，推动了能够存储和快速检索包含数百亿节点和数千亿边规模的图的分布式图数据库的发展。知识图谱使用图结构进行知识存储，因此包括最短路径识别、子图识别、中心度分析等在内的图算法也在知识谱图谱构建和使用中发挥着重要作用。

由于知识图谱涉及到多个交叉研究领域，相关知识点繁多，我们从知识图谱表示、存储、构建、推理、应用等几个技术维度对知识图谱所涉及的研究内容进行介绍。

1. 知识图谱表示: 知识图谱根据使用场景和应用需求，可以采用有向图、有向标记图 (Directed Labelled Graph)、本体描述语言等方式在逻辑层面进行表示。从物理层面研究上述逻辑表示的数据模型，主要包含属性图、RDF 图模型、OWL 本体语言等。近年来，随着神经网络研究的兴起，

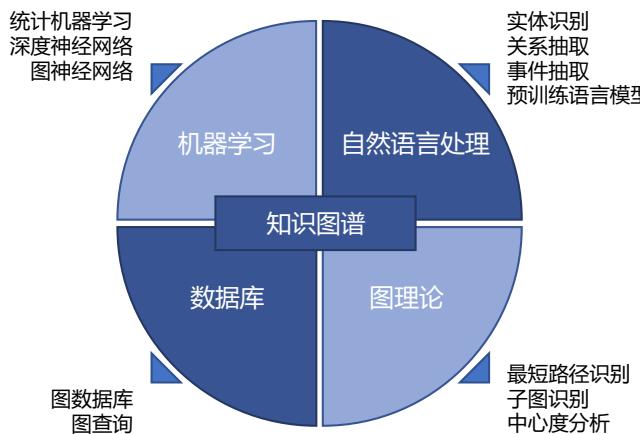


图 12.4 知识图谱研究涉及领域

基于向量的表示方法因为计算高效等特点也被广泛使用。

2. 知识图谱存储：随着知识图谱规模不断扩大，如何能够高效存储和检索包含百亿级顶点千亿级边规模的图是知识图谱应用必不可少的基础工作。在知识图谱表示的基础上，如何利用传统关系型数据库和原生图数据库实现知识图谱存储和检索，以及各种方案的优缺点是知识图谱存储所要研究的重点内容。

3. 知识图谱构建：知识广泛存在于自然语言文本、半结构化以及结构化的数据中，知识图谱构建主要目标就是研究如何利用实体识别、关系抽取等信息抽取技术，从自然语言文本中抽取实体、属性、关系、事件等知识图谱要素的方法，以及属性补全、实体链接、实体对齐等知识图谱扩展和融合方法。

4. 知识图谱推理：知识图谱不仅可以提供知识的存储和检索，更重要的是可以根据构建的已知知识进行归纳、推断和预测未知的事实。知识图谱推理的研究主要基于符号逻辑和表示学习，实现演绎、归纳、溯因、类比等类型推理。

5. 知识图谱应用：在构建了大规模高质量知识图谱后，如何将知识与不同的自然语言处理任务进行深度融合，构建基于知识图谱的智能推荐系统、基于知识图谱的智能问答以及知识增强的自然语言处理算法是知识图谱应用所重点研究的内容。

在本章中，我们将分别针对上述知识图谱研究内容进行具体介绍，针对知识图谱应用以知识图谱问答为例进行介绍。

12.2 知识图谱表示与存储

知识图谱应用需要使用一种合理的知识表达方式，保证结构化的知识可以被计算机正确处理。从某种意义上可以说，知识表示是贯穿知识库的构建与应用全过程的关键问题。传统的知识图谱表示方法一般以符号式为主，包括属性图、RDF、OWL 本体语言等。符号表示方法的特点是可解释性强，但是依赖知识描述的准确性。近年来，随着深度神经网络研究的兴起，基于向量式的表示方法逐渐引起人们重视。向量式的方法容易捕获隐式知识，计算效率高，但是可解释性差。此外，向量式表示本质上属于统计模型，对没见过的实体表示质量较差，而自然界的实体一般呈长尾分布，这也在一定程度上限制了向量式表示方法的应用。

知识图谱的表示属于人对知识图谱定义的数据描述，而知识图谱存储则对应知识在物理层面上如何被计算机组织存放。显然，想要在下游任务方便地利用知识图谱，只有其逻辑描述方法还不够，还需要将其在物理介质上进行存储。设计知识图谱的存储方法，要结合知识图谱的图结构模型，即图的结构信息，还要考虑节点与边的属性所包含的语义信息。在此基础上，也要针对知识存储的空间利用率以及检索查询效率等问题进行合计。

本节将对符号式的知识图谱表示方法和向量式的表示方法进行介绍，并将介绍两类知识图谱存储方法：基于表的知识图谱存储和基于图的知识图谱存储。

12.2.1 知识图谱的符号表示

知识图谱需要建模各种实体、概念之间的关系，图因为其直观性和扩展性，自然地成为知识图谱描述数据的首选方法。图由点和边构成，图上的点可以建模实体，边则可以对应实体对之间的关系。图的表示方法除了降低了知识的理解难度之外，也便于机器存储。

在简单的应用场景下，无向图即可满足需求。若要进一步增强知识图谱的表达能力，给实体和边添加属性，则可以选择有向标记图。常用的有向图模型有两种：属性图和 RDF 图模型。在更加复杂的场景下，比如建模传递关系、自反关系，有向标记图仍然不能满足需要，这时候可以选用 RDFS/OWL 本体语言作为描述语言对 RDF 进行扩展。本节中，我们会分别对这几种常见的知识图谱表示方法进行介绍。

1. 属性图

属性图（Property Graph）^[10] 是一种有向标记图，包含三个要素：节点（Vertex）、边（Edge）、属性（Property）。节点对应知识图谱中的实体，边则是实体对之间的关系描述（需要注意的是边是有向的且有类型区别），其出发的节点为源节点，到达节点为目标节点，相应的边的类型则对应实体之间的关系标签。属性本质上是一个键值对，属性图可以灵活地为节点和边添加任意属性。图12.5给出了一个属性图示例，其描述了公司内员工、部门主管以及项目之间的关系信息。以员工和部门主管为例，图中的员工、部门主管都是用节点表示，员工节点和主管节点之间通过被任用的关系边建立连接。此外，在员工节点，还为其添加了很多属性，比如员工编号、生日等信息。当

然边也可以添加属性，比如在边“任命”的属性中添加时间信息。

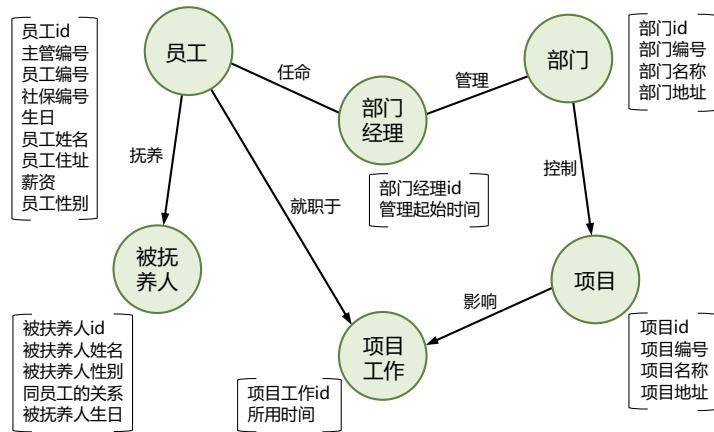


图 12.5 属性图示例

属性图模型在工业界广泛使用，一般通过图数据库 Neo4J^[11] 等实现。属性图的优点是直观、灵活，可以根据使用需求任意给节点或者边添加属性。此外，Neo4J 的图数据库中有大量针对图结构进行的优化工作，使得其查询效率有显著提升，具体信息将在知识图谱存储章节进行详细介绍。属性图与无向图表示的不同之处在于，在属性图的模型中，关系被提升到了与实体一样重要的程度。属性图提供了一个丰富的视角，即不同种类的数据如何相关，而这些数据依赖关系在普通的关系型数据库中难以直接展示。属性图的缺点则是无法建模深层次的语义信息，这直接限制了其语义推理能力。

2. RDF 图模型

资源描述框架（Resource Description Framework, RDF）^[12]，是由国际万维网联盟 W3C 制定，用于描述实体/资源的数据模型。RDF 的基本组成单元为一个 SPO 三元组 <主体 (Subject, 谓词 (Predicate), 客体 (Object) >，用来表示一条客观世界的逻辑描述或客观事实。知识图谱正是由一些相互连接的实体和属性构成，换言之，一条三元组就对应了知识图谱中的一条知识，例如：“<梅西，是，足球运动员>”。多个三元组首尾相连，就构成了一个 RDF 图，如图12.6所示。

通过 RDF 的描述框架，可以将 <梅西，是，足球运动员> 的知识形式化表达为：“梅西 rdf:type 足球运动员”。RDF 也存在一些问题，首先就是无法区分概念和对象。概念和对象可以类比于面向对象程序语言中的 Class 和 Object，还是以图12.6为例，“梅西”描述的是一个对象，而“足球运动员”描述的是一个概念，而 <足球运动员，是，运动员> 描述的是 subClassOf 关系。如果不能正确区分概念和对象，不仅会引起理解上的困难，还会给后续的知识图谱融合以及下游任务的应用带

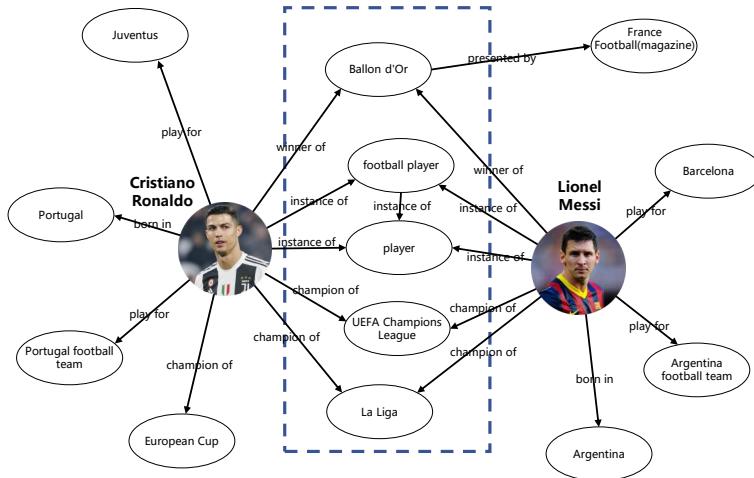


图 12.6 RDF 示例图

来障碍。

鉴于 RDF 表达能力有限，无法区分概念和对象，也无法描述类之间的关系和属性。W3C 提出了资源描述框架模式（Resource Description Framework Schema, RDFS）语言用来描述 RDF。RDFS 中定义了 Class 用来描述类，Domain 表示属性属于何种类别，Range 用来限制属性的取值，subClassOf 用作描述类的父类，subProperty 则描述属性的父属性。基于这些基础的表达构件，RDFS 可以实现一些简单的符号推理。例如，基于“梅西 rdf:type 足球运动员”，“足球运动员 rdfs:subClassOf 运动员”，可以推断出“梅西 rdf:type 运动员”。

3. OWL 本体语言

RDFS 是对 RDF 的一次成功扩展，但其表达能力仍然不足，无法完整表达复杂概念以及复杂概念间关系。为此，W3C 又开发了网络本体语言（Web Ontology Language, OWL）^[13]。OWL 本体语言与 RDFS 类似，本质上是一些预定义的表达构件集合，都是用来描述 RDF 数据。但是，OWL 本体语言相比 RDFS 添加了额外的定义词汇，因此可以当做是 RDFS 的扩展版。在这里介绍几种比较典型的 OWL 本体语言表达构件。

OWL 本体语言引入了本体映射表达构件，使用 owl:equivalentClass 表示两个类是相同的，例如，可以定义“公司”和“企业”是相同的概念；使用 owl:equivalentProperty 表示两个属性是相同的，例如，可以定义“年龄”和“春秋”为含义相同的属性；使用 owl:sameAs 表示两个实体是同一实体，例如，“利昂内尔·梅西”和“梅西”指代同一个足球运动员。

OWL 本体语言中也引入了关于属性的复杂描述词汇，使用 owl:TransitiveProperty 表示属性具有传递性质，例如，“属于”是具有传递性的属性，若 A 属于 B，B 属于 C，那么 A 肯定属于 C；使

用 owl:SymmetricProperty 表示属性具有对称性，例如，“夫妻关系”是具有对称性的属性，若 A 与 B 是夫妻关系，那么 B 与 A 是夫妻关系；使用 owl:inverseOf 定义两个属性的相反关系，例如，“咨询”和“指导”是相反关系，若 A 咨询 B，则 B 指导 A。类似地，还可以定义属性的等价性、唯一性和属性间的互逆性，甚至可以约束属性满足一定的函数约束。

OWL 本地语言中还引入了 owl:unionOf、owl:intersectionOf 和 owl:complementOf 等布尔算子，分别表示集合的并集、交集和补集的运算。更多的 OWL 本体语言表达构件和特性请参考 W3C 官方文档。

12.2.2 知识图谱的向量表示

基于符号表示的大规模知识图谱在实际应用中面临两个问题：(1) 知识表达能力差，因为知识图谱中的实体一般符合长尾分布，许多实体仅有少数关系相连，对于这些稀疏的实体与关系，很难做到充分、完整的知识表达；(2) 计算效率低，基于图结构的知识表示方便人类理解，但是将其应用于下游任务时需要设计相应的图算法，这些算法计算复杂度往往较高，使得大规模知识图谱的应用门槛大大提高。

随着深度学习的广泛应用，越来越多的研究人员开始关注知识表示学习（Knowledge Representation Learning），即如何构建高质量的向量表示。知识表示学习通过将三元组中的语义信息投影到稠密的低维向量空间，构造实体和关系的分布式表示向量。这种表示向量单独地看每一维度并没有明确含义，但是综合各维度形成的向量却能够表示对象的语义信息。这种知识表示方法相对符号式表示有如下几个优点：(1) 知识表达能力强，分布式向量可以更好地建模对象之间的关系，语义相似的对象往往其表示向量也更接近，可以缓解长尾分布的知识表达问题；(2) 计算效率更高，对于计算机来说，已经提前计算好的蕴含语义知识的低维数值向量显然比复杂的知识图谱更高效；(3) 适用于深度学习算法，通过将知识映射到语义空间中，使得不同来源的知识可以很方便地互相融合。

研究人员们先后提出了多种模型学习知识库中实体和关系的表示向量，其中最常见的是基于距离度量的知识表示学习方法。基于距离度量的方法表示通过计算实体之间的距离来评价事实三元组的置信度。距离模型（Distance Model）又可称为平移模型（Translational Model），该类模型将知识图谱中的每个关系看作从头实体向量到尾实体向量的一个平移变换。通过最小化平移转化的误差，将知识图谱中的实体和关系类型映射到低维空间。

1. 知识图谱向量表示学习 TransE 算法

本书第 4 章介绍了 CBOW 和 Skip-Gram 等分布式单词向量表示^[14]，在其研究中发现词向量空间存在平移不变现象，例如：

$$\mathbf{v}(\text{man}) - \mathbf{v}(\text{woman}) \approx \mathbf{v}(\text{king}) - \mathbf{v}(\text{queen}) \quad (12.2)$$

这里 $v(w)$ 表示单词 w 的词向量。换言之，词向量捕获到了 man 和 woman、king 和 queen 之间的近似关系。

受该现象的启发，文献 [15] 提出了 TransE 模型用于学习实体和关系的向量表示。TransE 模型将关系看做是表示空间中的平移(Translation)。如图12.7所示，在表示空间中，对于三元组 $\langle h, l, t \rangle$ ，头实体 h 的向量加上关系 r 的向量等于尾实体 t 的向量。

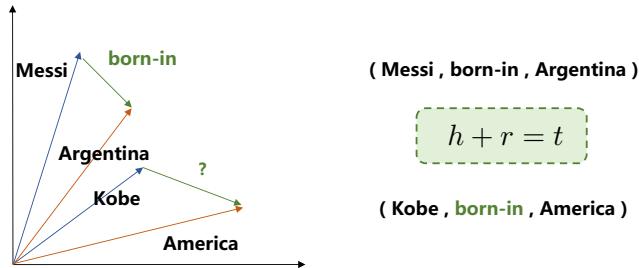


图 12.7 TransE 模型^[14]

对于给定的由三元组集合 S 组成的知识图谱，其中三元组 $\langle h, l, t \rangle$ 中 h 和 t 是头实体和尾实体， l 是关系， $h, \ell, t \in \mathbb{R}^k$ 是对应的向量表示。根据平移假设，对于图谱中存在的三元组应该符合 $h + \ell \approx t$ 。为了保证不同三元组之间的区分度，TransE 除了知识库中存在的三元组 $(h + \ell, t)$ 之外，还构造了知识库中不存在的三元组 $(h' + \ell, t')$ ，采用合页损失 (Hinge Loss) 作为模型的损失函数：

$$\mathcal{L} = \sum_{\langle h, \ell, t \rangle \in S} \sum_{\langle h', \ell, t' \rangle \in S'_{\langle h, \ell, t \rangle}} [\gamma + d(h + \ell, t) - d(h' + \ell, t')]_+ \quad (12.3)$$

其中 γ 为正负例的得分间隔距离， S 为正例集合， S' 为负例集合。为了选取有代表性的错误三元组，TransE 算法将 S 中每个三元组的头实体、关系和尾实体其中之一随机替换成其他实体或关系来得到 S' ，即：

$$S'_{\langle h, \ell, t \rangle} = \{\langle h', \ell, t \rangle | h' \in \mathcal{E}\} \cup \{\langle h, \ell, t' \rangle | t' \in \mathcal{E}\} \cup \{\langle h, \ell', t \rangle | \ell' \in \mathcal{R}\} \quad (12.4)$$

TransE 算法的优化过程如算法12.1所示，首先，将所有实体和关系的嵌入向量随即初始化。在每次迭代开始之前，会对实体的向量进行归一化操作。之后，在每次批量训练时，从训练集中随机采样一些三元组。对于每个三元组，会构造一个负例三元组，用于计算12.3损失函数。根据损失函数产生的梯度，对实体和关系的表示向量进行更新。迭代上述过程，直至算法在验证集上的效果收敛。

代码 12.1: TransE 模型训练算法

输入: 训练集合 $S = \langle \mathbf{h} + \boldsymbol{\ell}, \mathbf{t} \rangle$, 实体集合 E , 关系集合 L , 超参数间隔 γ , 表示向量维度 k

初始化 $\boldsymbol{\ell} \leftarrow \text{uniform}\left(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}}\right)$ 对每个 $\ell \in L$;

$\ell \leftarrow \ell / \|\ell\|$ 对每个关系 $\ell \in L$;

$e \leftarrow \text{uniform}\left(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}}\right)$ 对每个实体 $e \in E$;

repeat

$e \leftarrow e / \|e\|$ 对每个实体 $e \in E$;

$S_{\text{batch}} \leftarrow \text{sample}(S, b)$; // 采样大小为 b 的批次样本;

$T_{\text{batch}} \leftarrow \emptyset$; // 初始化三元组集合;

for $\langle h, \ell, t \rangle \in S_{\text{batch}}$ **do**

$\langle h', \ell, t' \rangle \leftarrow \text{sample}\left(S'_{\langle h, \ell, t \rangle}\right)$; // 构造冲突三元组;

$T_{\text{batch}} \leftarrow T_{\text{batch}} \cup \{(\langle h, \ell, t \rangle, \langle h', \ell, t' \rangle)\}$;

end

根据以下损失函数更新表示向量

$\sum_{(\langle h, \ell, t \rangle, \langle h', \ell, t' \rangle) \in T_{\text{batch}}} \nabla [\gamma + d(\mathbf{h} + \boldsymbol{\ell}, \mathbf{t}) - d(\mathbf{h}' + \boldsymbol{\ell}, \mathbf{t}')]_+$;

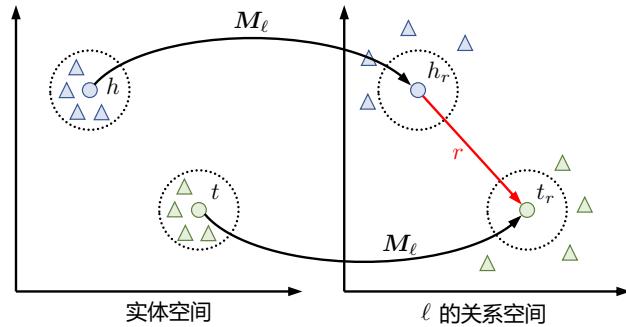
until 验证集效果收敛;

2. 知识图谱向量表示学习 TransR 算法

TransE 算法存在不能正确建模知识图谱中存在的一对多、多对一以及多对多的复杂关系的问题。例如，知识图谱中存在 $<\text{曹操}, \text{出生于}, \text{东汉末年}>$ 和 $<\text{刘备}, \text{出生于}, \text{东汉末年}>$ 两个事实，根据通过 TransE 算法所训练出来的“曹操”和“刘备”的表示很相似，无法区分二者。又比如“曹操”在不同语境下可能是“军事家”，也可能是“诗人”。此外，TransE 算法将实体和关系映射到相同的空间中，但是关系和实体是完全不同的对象，共同的语义空间可能不足以表示它们。因此，这些复杂的场景下，TransE 算法过于理想化的平移假设显然不能满足需求。

针对上述问题，文献 [16] 提出了 TransR 算法，为每种关系 ℓ 定义了单独地语义空间。给定三元组 $\langle h, \ell, t \rangle$ ，头尾实体表示分别是 $\mathbf{h}, \mathbf{t} \in \mathbb{R}^k$ ，关系表示向量 $\boldsymbol{\ell} \in \mathbb{R}^d$ ，实体表示和关系表示的维度不需要相等。TransR 算法通过针对每种关系的不同的映射矩阵 M_ℓ 将实体映射到关系空间中，要求实体在关系空间中满足平移关系即可。如图12.8所示，特定关系投影可以使实际具有该关系的头/尾实体（图中表示为彩色圆圈）彼此靠近，也可以使得不具有该关系的实体（图中表示为彩色三角形）则彼此远离。TransR 本质上是放宽了 TransE 的 $\mathbf{h} + \boldsymbol{\ell} = \mathbf{t}$ 假设，将原来在一个空间中满足的平移关系，改为在不同的关系空间中满足。

具体来说，给定三元组 $\langle h, \ell, t \rangle$ ，TransR 算法通过关系 ℓ 对应的映射矩阵 M_ℓ ，将头实体和

图 12.8 TransR 模型^[16]

尾实体映射到关系空间中，得到实体表示 h_ℓ 和 t_ℓ :

$$h_\ell = hM_\ell, \quad t_\ell = tM_\ell \quad (12.5)$$

在得到对应关系空间的表示 h_ℓ 和 t_ℓ 后，就可以使用与 TransE 类似的评分函数进行训练:

$$f_\ell(h, t) = \|h_\ell + \ell - t_\ell\|_2^2 \quad (12.6)$$

在实现时，TransR 算法还对 h , ℓ , t 的表示向量以及关系空间中映射的表示向量的范数添加约束。

$$\forall h, \ell, t, \|h\|_2 \leq 1, \|\ell\|_2 \leq 1, \|t\|_2 \leq 1, \|hM_\ell\|_2 \leq 1, \|tM_\ell\|_2 \leq 1 \quad (12.7)$$

12.2.3 基于表的知识谱图谱存储

尽管知识图谱是用图的形式描述，但图数据库并不是存储知识图谱的唯一方案。在工业界，很多成熟的数据库都是基于关系模型，知识图谱的数据可以通过一定的设计，从而可以利用关系型数据库存储。基于关系型数据库的知识图谱存储方案主要可以分为四种：基于三元组的知识图谱存储、基于属性表的知识图谱存储、基于垂直表的知识图谱存储以及基于全索引的知识图谱存储。

1. 基于三元组的知识图谱存储

利用关系型数据存储知识图谱最简单的办法就是构建一个由主体、谓词和客体三列构成的表，将知识图谱中的每个 SPO 三元组看做该表中的一条记录，直接存储到关系型数据库中，如表12.1所示。

当用户输入一个查询请求时，需要将其转换为对应的查询 SQL 语句。但是稍微复杂一点的查询，需要大量的自连接（Self-Join）操作。例如，用户需要查询“获得金球奖的足球运动员出生地”，这需要将整张表复制三次，然后依次扫描，基于查询结果再做连接（Join）操作，查询代价过于昂贵。

表 12.1 基于三元组的知识图谱存储

主体	谓词	客体
利昂内尔·梅西	出生地	阿根廷
利昂内尔·梅西	isA	足球运动员
利昂内尔·梅西	生日	1987-06-24
利昂内尔·梅西	效力	巴黎圣日耳曼足球俱乐部
利昂内尔·梅西	荣誉	金球奖
杰西·林加德	出生地	英格兰
杰西·林加德	isA	足球运动员
杰西·林加德	效力	曼彻斯特联足球俱乐部
科比·布莱恩特	出生地	美国
科比·布莱恩特	isA	篮球运动员
科比·布莱恩特	效力	洛杉矶湖人队
科比·布莱恩特	荣誉	NBA 总决赛
足球运动员	subClassOf	运动员
篮球运动员	subClassOf	运动员
金球奖	英文名	Ballon d'Or
NBA 总决赛 MVP	英文名	NBA Finals MVP

2. 基于属性表的知识图谱存储

为了解决基于三元组的知识图谱存储查询效率过低问题，还可以按照属性表（Property Tables）^[17]的方式存储知识图谱。其存储方案仍然是基于关系型数据库，不同之处在于，属性表根据实体类型对三元组进行分类，不同类别的三元组存储于不同的表中。例如，针对上一节中的三元组存储表案例转换为属性表后，其结构如表12.2所示。将运动员、类型、生日、出生地、荣誉等信息合并到一张表中进行保存。对于上例中用户查询“获得金球奖的足球运动员出生地”，只需要检索“荣誉”对应的属性表就可以完成。

表 12.2 基于属性表的知识图谱存储示例

主体	isA	生日	出生地	效力	荣誉
利昂内尔·梅西	足球运动员	1987-06-24	阿根廷	巴黎圣日耳曼足球俱乐部	金球奖
杰西·林加德	足球运动员		英格兰	曼彻斯特联足球俱乐部	
科比·布莱恩特	篮球运动员		美国	洛杉矶湖人队	NBA 总决赛 MVP

主体	英文名
金球奖	Ballon d'Or
NBA 总决赛 MVP	NBA Finals MVP

主体	subClassOf
足球运动员	运动员
篮球运动员	运动员

虽然属性表的效率相较于三元组表在一定程度上有所提升，但是知识图谱的数据往往具备稀

疏性，因此基于属性表的存储方案可能会包含大量空值，造成存储空间的极大浪费。此外，属性表的存储方案还依赖于人的设计，哪些属性需要合并到一张表中，需要根据知识图谱需求和实际关系联合考虑。

3. 基于垂直表的知识图谱存储

除了按照实体对三元组表格分类外，还可以选择按照谓词构造表格，这种方式称为基于垂直表（Vertical Tables）^[17] 的存储方式。将不同的谓词对应不同的表格，这种方法不仅可以避免查询过程中大量的自连接操作，还不会在数据库中产生大量的空值。表12.2中三元组存储表案例被转换为垂直表后，其结构如表12.3所示。根据谓词构建了 isA、出生地、效力等多个表。

表 12.3 基于垂直表的知识图谱存储

isA		出生地		荣誉		生日	
主体	客体	主体	客体	主体	客体	主体	客体
利昂内尔·梅西	足球运动员	利昂内尔·梅西	阿根廷	利昂内尔·梅西	金球奖	利昂内尔·梅西	1987-06-24
杰西·林加德	足球运动员	科比·布莱恩特	美国	科比·布莱恩特	NBA 总决赛 MVP		
科比·布莱恩特	篮球运动员	杰西·林加德	英格兰				

效力		英文名		subClassOf	
主体	客体	主体	客体	主体	客体
利昂内尔·梅西	巴黎圣日耳曼足球俱乐部	金球奖	Ballon d'Or	足球运动员	运动员
杰西·林加德	曼彻斯特联足球俱乐部	NBA 总决赛 MVP	NBA Finals MVP	篮球运动员	运动员
科比·布莱恩特	洛杉矶湖人队				

这种方法缺点是对于具备多个属性的实体，往往需要对多个表进行插入操作，操作效率低。对于属性不确定的查询，垂直表也不能很好的处理，例如，查询“科比·布莱恩特和洛杉矶湖人队的关系”，就需要检索所有表才能得到结果，这类查询的检索性能较差。

4. 基于全索引的知识图谱存储

基于三元组存储的另一种改进优化方法是基于全索引的存储方式，即通过对关系型数据库建立多种索引的方式，达到查询效率优化的目的。全索引表的优化方法，首先会将 SPO 三元组里的每个元素集合映射为一个数字 ID 集合，通过存储 ID 而不是字符串可以节省大量存储空间，如表12.4所示。然后，对三元组中主体、谓词、客体的各种排列情况都分别建立索引，这种组合一共

有六种：SPO、POS、PSO、OSP、OPS、SOP，如图12.9所示。无论是基于谓词查询主体和客体，还是基于客体查询谓词和主体，六种索引都可以满足需求。全索引的方法应对简单的查询效率高，但是其索引的更新与维护同样代价高昂。同时，面对复杂一些的查询，全索引仍然不可避免要做多次自连接操作，效率低下。

表 12.4 基于全索引的知识图谱存储

Value	ID	Value	ID
利昂内尔·梅西	0	1987-06-24	12
杰西·林加德	1	巴黎圣日耳曼足球俱乐部	13
科比·布莱恩特	2	金球奖	14
出生地	3	英格兰	15
isA	4	曼彻斯特联足球俱乐部	16
生日	5	美国	17
效力	6	篮球运动员	18
荣誉	7	洛杉矶湖人队	19
subClassof	8	NBA 总决赛 MVP	20
英文名	9	运动员	21
阿根廷	10	Ballon d'Or	22
足球运动员	11	NBA Finals MVP	23

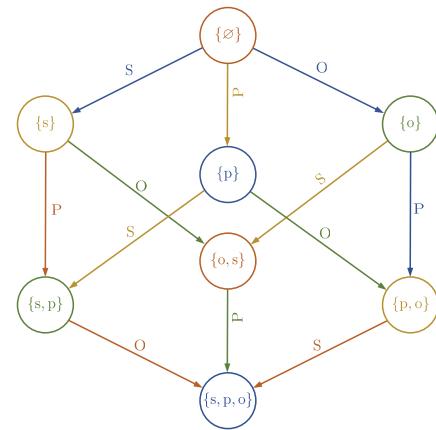


图 12.9 六重索引

12.2.4 基于图的知识谱图谱存储

在图数据库技术还不成熟的初期，关系型数据库因其成熟的技术架构，成为知识图谱存储的首选方案。但是关系型数据库却并不善于处理“关系”，在关系型数据库上进行图数据上的多跳查询，会产生大量的连接操作，其计算复杂度随着查询的跳数增多呈指数级增长。以表12.1为例，如果要查询“获得 NBA Finals MVP 荣誉的球员效力于哪个俱乐部”，关系型数据库至少要三次表连接操作。知识图谱除了需要提供高效的关联推理，还要能刻画丰富的语义表达能力，诸如传递关系、自反关系、对称关系和函数关系等等。

图数据库则充分利用图的结构对数据进行建模，将一张图对应到一个邻接列表，再基于这个邻接列表建立索引，优化关联查询。例如，查询“获得 NBA Finals MVP 荣誉的球员效力于哪个俱乐部”，只要依次对“NBA Finals MVP”、“NBA 总决赛 MVP”、“科比·布莱恩特”三个节点的邻居节点进行检索即可。此外，在图数据库中，关系是被显式描述和刻画的，属性也可以单独定义，这就极大地增加了数据建模的灵活性。随着相关技术和工具的逐渐成熟，图数据库已逐渐成为知识图谱存储和查询引擎搭建的基础设施。

1. 图数据库使用案例

在本章第12.2.1节知识图谱表示部分，我们介绍了两种图模型：属性图模型和RDF图模型。属性图模型是图数据库Neo4J所引导的一种数据模型，使用Cypher^[18]作为查询语言。因为其性能较好，已经在工业界得到广泛应用。RDF图模型严格地说并不是一个存储模型，而是一种规范定义

的数据交换的格式标准，使用 SPARQL 查询语言^[19]。其中，属性图数据库善于处理关联查询，而 RDF 图模型则提供更多的关联推理能力。本节以 Neo4J 为例，对基于图数据库的知识图谱的存储和查询进行简要说明。

在 Neo4J 数据库中，数据以节点（Node）和边（Edge）进行组织。节点可以代表知识图谱中的实体，边可以代表实体间的关系。需要特别注意的是，关系具有类型和方向性。在节点上可以添加标签（Label）以及键值对（Key-Value）来表示实体具备的属性（Property）。以 2022 世界杯为例，其属性图模型如图12.10所示，每一年的世界杯对应一个 WorldCup 节点，每场比赛对应一个 Match 节点，球员节点为 Player。Country 节点通过 HOSTED_BY 关系连接到 WorldCup 节点，同时每个 Country 都会指定一个球员小队 Squad，代表他们参加世界杯比赛。对于 Player 作为首发或替补参加的每场比赛，都连接到 Appearance 节点，如果球员取得分数，则 Appearance 将连接到该得分节点 Goal。

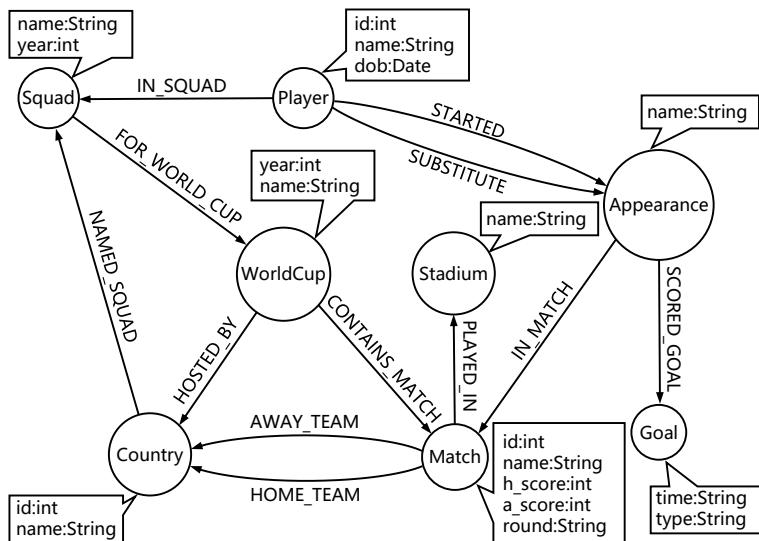


图 12.10 Neo4j 的属性图模型样例

Neo4J 采用自己独有的查询语言 Cypher，这是一种描述性语言，用户只需要声明“查什么”，而不用关心“怎么查”。Cypher 语言与 SQL 很相似，但是 Cypher 关键字不区分大小写，属性值、标签、关系类型和变量则区分大小写。Cypher 中最常用到的关键词是：

- **match:** 相当于 SQL 中的 select，用来说明检索匹配的图模式。
- **where:** 用来限制节点或者边中的属性值，从而过滤掉不需要的返回值。
- **return:** 返回节点或者关系。

Neo4J 支持知识图谱的路径查询，以图12.10的图结构为例，查询“世界杯的举办国”，返回世

世界杯节点和举办国节点之间的路径，可以使用如下 Cypher 语句：

```
MATCH path = (wc:WorldCup)-[:HOSTED_BY]->(country)
RETURN path
```

返回结果如图所示12.11。

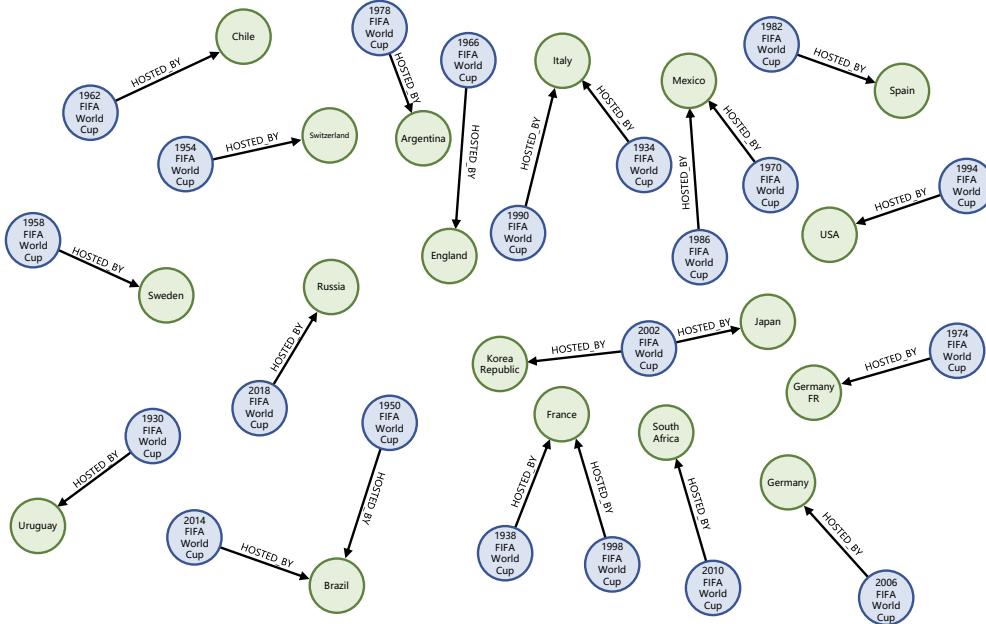


图 12.11 Neo4j 的路径查询样例

如果需要直接返回表格数据，例如查询“举办过世界杯的国家”，可以采用类似 SQL 的语法组织查询：

```
MATCH (host:Country)<-[HOSTED_BY]-(wc)
WITH wc, host ORDER BY wc.year
WITH host, count(*) AS times, collect(wc.year) AS years
WHERE times > 1
RETURN host.name, times, years
```

其结果如表12.5所示。

知识图谱的存储需要综合考量查询性能、扩展性和存储成本等多种因素，使用者需要根据具体的应用场景和数据规模选择合适的存储方式。基于属性图的图数据库、基于 RDF 的存储系统和关系型数据库的特点可以总结如表12.6所示。对于简单查询，传统的关系型数据库在性能方面更

表 12.5 Neo4J 查询结果返回样例

host.name	times	years
Mexico	2	[1970,1986]
France	2	[1938,1998]
Brazil	2	[1950,2014]
Italy	2	[1934,1990]

有优势，但是在处理多跳查询和建模复杂关系语义的任务上，基于图的知识图谱存储模式更加适配。属性图的图分析性能更强，比如涉及到子图匹配、图结构学习等任务。而 RDF 存储系统的关联推理能力更强，并且由于 RDF 作为一种数据交换的格式标准，具有更好的理论基础。

表 12.6 知识图谱不同存储方案比较

	基于属性图的图数据库	RDF 存储系统	关系型数据库
数据模型	属性图	RDF 三元组	关系数据模型
查询语言	Cypher、Gremlin ^[20]	SPARQL	SQL
应用场景	多为工业界	多为学术界	工业界和学术界均有使用
其他特点	图遍历效率高；图的全局操作效率低	有标准的推理引擎；易于数据发布	简单查询销量高；多跳查询会产生连接操作，效率低

12.3 知识图谱获取与构建

构建知识图谱最简单直接的方式是使用人工直接构建，但是这种方法得到的知识图谱规模和知识的覆盖面往往有限，而且不同专家的知识认知也不尽相同，难以用统一的标准精准地刻画专家知识，使得基于人工构建的知识图谱具有高度的不确定性、不准确性。如何构建大规模、高质量的知识图谱，实现海量知识的准确抽取和有效聚合，是知识图谱领域最重要的问题之一。

知识图谱的数据来源是多种多样的，包括结构化数据、半结构化数据以及非结构化数据等。对于已有的结构化数据，可以通过一定的规则来提取其中的知识，比如关系型数据库，其行（Row）可能代表知识图谱中的实例（Instance），列（Column）可能表示属性（Property），单元（Cell）对应属性值，外键（Foreign Key）则可以表示指代关系。对于半结构化数据，比如维基百科，可以制定解析规则，将页面上的字段映射为知识图谱中的实体或者属性。如图12.12所示，维基百科中 IBM 词条的 InfoBox 区域包含了大量可以直接转化为知识图谱的信息。对于非结构化数据，即纯文本数据，首先需要从文本中挖掘出目标知识图谱所需的实体，可以应用命名实体抽取等技术；基于已知实体信息，还需要从文本中识别出它们之间的关系，这对应了 SPO 三元组中的谓词关系，需要使用关系抽取技术；除了实体和关系信息，还需要抽取属性信息，需要使用属性补全技术；更复杂的场景下需要对事件信息进行抽取，可以看作一组三元组的联合抽取过程。

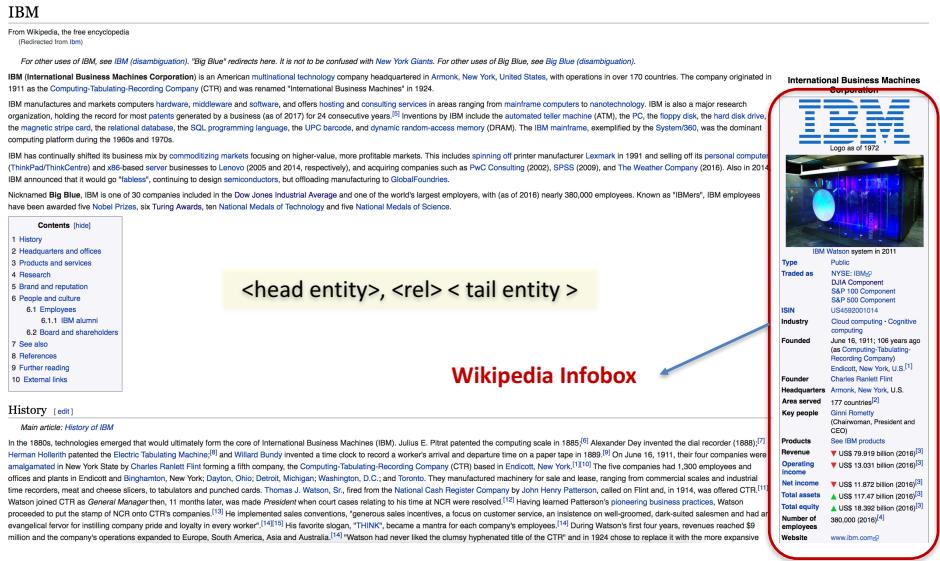


图 12.12 基于维基百科中半结构化数据构建知识图谱

考虑到自然语言的歧义性，比如“苹果”，在某些场景下指的是“苹果公司”，有的时候则表示的是水果，因此还需要引入实体链接技术消除不同文本中实体指称的一词多义问题。此外，自然语言具有表达的多样性，比如“复旦大学”，有的表达中使用“复旦”，这使得在不同的知识图谱中相同的实体会有不同的实体指代，因此需要引入实体对齐技术解决本体或实例的异构问题。知识图谱构建所依赖的实体抽取、关系抽取等方法在本书第 7 章信息抽取部分有详细介绍，这里我们就不赘述，本节将会依次介绍属性补全、实体链接技术和实体对齐常见方法。

12.3.1 属性补全

现实世界中的实体并不总是单一存在，往往伴随许多属性对其进行修饰，例如，复旦大学有在校师生数、校庆日等属性。准确地掌握实体的属性信息，能够多方位地描述实体特征，提升知识的建模能力。属性补全的目的就是自动地从文本中提取出实体所具备的属性，特别是在电子商务等垂直领域下，属性补全有着重要的应用，如图12.13所示，若能自动从产品描述中提取出产品的属性，将能更好的描述产品特征，优化产品搜索和推荐效果。

属性补全方法通常采用抽取式方法，类似于命名实体识别，将属性补全转为序列标注任务，即对句子中的每个字进行分类，在此基础上基于字的分类结果解析得到属性值。但如果简单地将其等价于普通的序列标注任务，将属性类型等价于实体类型，会带来两个问题：(1) 普通的命名实体识别模型一般只建模数个实体类型，而领域知识图谱可能包含成千上万的属性种类，且这些属性往往符合二八法则（Power Laws），直接应用普通的序列标注模型，由于标签数量巨大，会造成性

First Aid Beauty Ultra Repair Cream: Vegan and Gluten-Free Intense Moisturizer for Dry Sensitive Skin. Perfect for Skin Conditions and Eczema. **Pink Grapefruit (14 ounce)**

About this item

- HEAD-TO-TOE: Head-to-toe moisturizer that provides instant relief and long-term hydration for **dry, distressed** skin, even eczema. The beautiful, whipped texture is instantly absorbed with no greasy after-feel. **Grapefruit** has a bright **citrus** fruit scent that is fresh, juicy and sparkling.
- CLINICALLY PROVEN: Formulated with Colloidal Oatmeal, Shea Butter, Ceramide 3 and the FAB Antioxidant Booster, it provides immediate relief and visible improvement for parched skin and it is clinically proven to increase hydration by 169% immediately upon application.

Product description

Banish **dry** skin with First Aid Beauty's Ultra Repair Cream. Suitable for **all** skin types, especially **dry, flaky** skin, this hydration wonder leaves skin feeling smooth, hydrated and comfortable after just a single use.

Mentioned Attributes:	Brand	SkinType	Scent	Quantity

图 12.13 电商场景下的产品属性示例

能的显著下降；(2) 知识图谱的构建需要保证拓展性，保留属性集合扩展的可能，而当前实体识别模型一般是基于封闭世界假设，无法识别新的实体种类，若直接应用，同样无法识别未标注过的新属性。

为了解决这个问题，文献 [21] 提出了一种基于属性理解的属性抽取方法 AVEPT。AVEPT 算法将属性理解问题转化为抽取式阅读理解任务，待抽取的文本作为上下文，目标属性信息作为问句，属性值则作为答案，是模型要抽取出来的目标。由此，将抽取结果和属性类型进行了解偶，并且输入的属性信息可以帮助模型捕获属性的语义信息。因此，该方法在取得较好的效果同时，还保留了模型对新属性识别的拓展性。AVEPT 算法神经网络结构如图 12.14 所示。AVEPT 模型可以分为简单分为四层：词表示层、上下文编码层、注意力层、输出层。

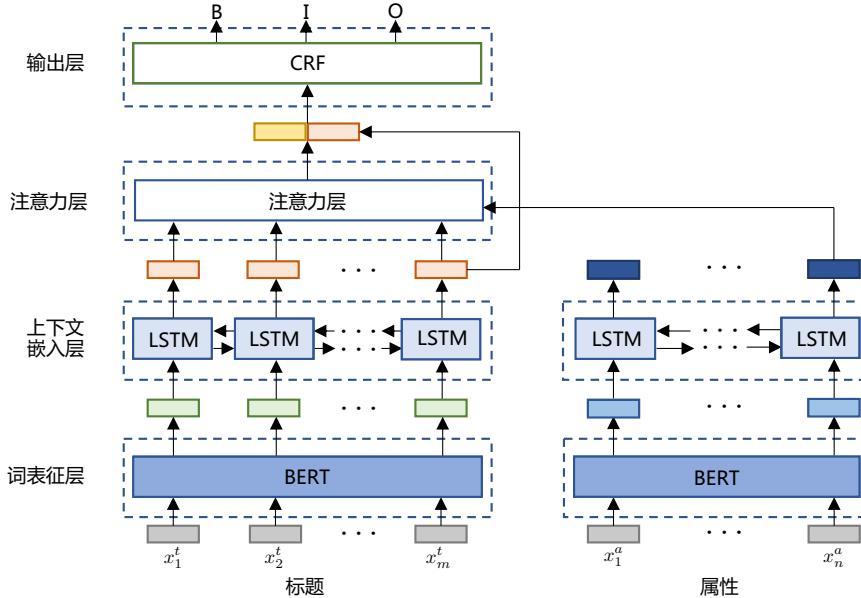
具体来说，待抽取文本为 $T = \{x_1^t, x_2^t, \dots, x_m^t\}$ ，长度为 m 。目标属性记为 $A = \{x_1^a, x_2^a, \dots, x_n^a\}$ ，长度为 n ，模型的目标输出序列为 $y = \{y_1, y_2, \dots, y_m\}$ ， $y_i \in \{B, I, O\}$ ， B 和 I 分别表示属性值的开始和中间， O 表示非目标属性值。

词表示层 (Word Representation Layer)，AVEPT 使用预训练语言模型 BERT 将待抽取文本 T 和属性 A 里的词分别映射为词向量表示，即 $\{\mathbf{w}_1^t, \mathbf{w}_2^t, \dots, \mathbf{w}_m^t\}$ 和 $\{\mathbf{w}_1^a, \mathbf{w}_2^a, \dots, \mathbf{w}_n^a\}$ 。

上下文编码层 (Contextual Embedding Layer)，为了建模长距离的上下文信息，AVEPT 选用两个不同的双向长短时记忆网路 (BiLSTM) 分别编码待抽取文本和目标属性的词向量表示。将待抽取文本对应的 BiLSTM 的隐状态向量作为其上下文表示，即 $\mathbf{H}^t = \{\mathbf{h}_1^t, \mathbf{h}_2^t, \dots, \mathbf{h}_m^t\}$ ，其中每个词的隐状态向量计算方式为：

$$\mathbf{h}_i^t = [\overrightarrow{\mathbf{h}}_i^t; \overleftarrow{\mathbf{h}}_i^t] = \text{BiLSTM}(\overrightarrow{\mathbf{h}}_{i+1}^t, \overleftarrow{\mathbf{h}}_{i-1}^t, \mathbf{w}_i^t) \quad (12.8)$$

与待抽取文本的上下文表示不同，目标属性只使用 BiLSTM 最后时间步的隐状态向量作为表示

图 12.14 AVEPT 算法神经网络结构图^[21]

向量。

$$\mathbf{h}^a = [\overrightarrow{\mathbf{h}}_n^a; \overleftarrow{\mathbf{h}}_n^a] = \text{BiLSTM}(\overrightarrow{\mathbf{h}}_n^a, \overleftarrow{\mathbf{h}}_n^a, \mathbf{w}_n^a) \quad (12.9)$$

注意力层 (Attention Layer)，不同于普通的自注意力方法捕获重要的词汇，AVEPT 考虑了属性和待抽取文本之间隐藏的语义交互。这种方法相比自注意力具有更好的泛化性，即使是训练中没有见过的属性，也能抽取对应属性值。具体来说，AVEPT 首先计算属性和待抽取文本中每个词的相似度，得到注意力向量 $\mathbf{S} = \{\alpha_1, \alpha_2, \dots, \alpha_m\}$ ，向量之间的相似度计算采用余弦函数。

$$\alpha_i = \cos(\mathbf{h}_i^t, \mathbf{h}^a) \quad (12.10)$$

然后使用注意力向量对待抽取文本的上下文表示进行放缩，即 $\mathbf{C} = \mathbf{S} \odot \mathbf{H}^t$ ，其中 \odot 表示按位乘。

输出层 (Output Layer)，为了捕获邻近词的标签依赖关系，AVEPT 采用条件随机场 (CRF) 作为解码器。CRF 的输入 \mathbf{M} 为文本的上下文表示 \mathbf{H}^t 和属性放缩表示 \mathbf{C} 的拼接矩阵，这样可以同时保存上下文信息和上下文对于属性重要性的信息。

$$\mathbf{M} = [\mathbf{H}^t; \mathbf{C}] \quad (12.11)$$

使用 CRF 对标签进行解码，可以得到标签序列的联合概率密度：

$$\Pr(y \mid T; \psi) \propto \prod_{i=1}^m \exp \left(\sum_{k=1}^K \psi_k f_k(y_{i-1}, y_i, M_i) \right) \quad (12.12)$$

其中 ψ_k 表示相应权重， f_k 为特征函数， K 为特征数量。模型最终输出的标签序列对应概率值最大的标签路径：

$$y^* = \operatorname{argmax}_y \Pr(y \mid u; \psi) \quad (12.13)$$

AVEPT 的训练损失使用最大条件似然估计：

$$\mathcal{L}(\psi) = \sum_{i=1}^N \Pr(y_i \mid u_i; \psi) \quad (12.14)$$

12.3.2 实体链接

随着信息抽取和知识图谱的发展，涌现出很多大规模、高质量且机器可读的开放知识图谱，包括 YAGO^[22], DBpedia^[23], Freebase^[24], 和 Probbase^[25] 等。这些知识图谱包含数千万命名实体和数十亿命名实体间关系。实体链接 (Entity Linking) 目标是将文本中的实体指代和它们在知识图谱中的对应实体进行对应。对已有知识图谱扩充的时候，也需要实体链接任务将文本中的实体指代与其知识图谱中真正对应的实体完成映射。除此之外，实体链接任务也是很多下游应用的预处理工作，如问答、关系抽取、内容分析等。

实体链接的任务难点主要有两个：(1) 实体的歧义性，如图12.15所示，“MJ”可能表示篮球运动员“Michael Jordan”，也可能是美国歌手“Mj Rodriguez”；(2) 实体表达的多样性，即一个命名实体有不同的表示形式，如全称、缩写、别名等等，例如：“NYC”和“Big Apple”是命名实体“New York City”的缩写和昵称。歧义性和多样性的消除需要依据上下文语义信息以及实体属性和关系信息。

基于传统机器学习的实体链接算法，一般分为两步，首先提取人工设计的特征，如实体的 TF-IDF 值、局部上下文兼容性和文档级的实体引用全局一致性的特征等等。然后，将特征输入给实体排序模型，进行最后的链接预测。这种方法有两个不足：一是需要进行大量细致而繁琐的特征工程；二是由于特征设计过程对特定知识库或者领域知识强烈依赖，很难将已有的实体链接方法推广到其他知识库或者领域。随着深度学习研究的发展，很多深度学习方法也应用于实体链接算法。深度学习算法通常将实体链接任务分解为实体识别和实体消歧两部分，对它们分别建模，但是这样会导致模型错误传递。文献 [26] 提出一种端到端的实体链接方法 ENEL (End-to-End Neural Entity Linking)，同时建模实体识别和实体消歧任务。主要思想是考虑所有可能的文本片段作为潜在的实体，并学习其候选实体的上下文相似性分数。

实体链接任务输入为文档或句子 $D = \{w_1, \dots, w_n\}$ ，其中的每个词都来源于词典 $w_k \in \mathcal{W}$ 。输出为实体提及-实体对的列表 $\{(m_i, e_i)\}_{i \in [1, T]}$ ，其中每个实体提及为输入文档的子序列 $m =$

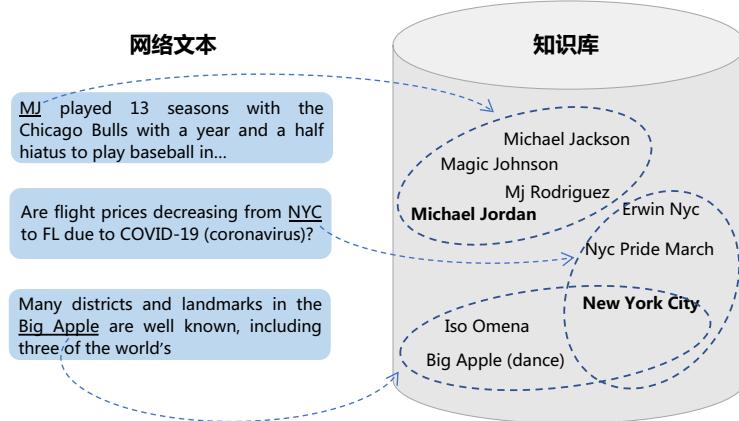


图 12.15 实体链接任务示意图

w_q, \dots, w_r , 每个实体则对应知识库里的一个实体条目 $e \in \mathcal{E}$ 。需要注意的是, ENEL 算法只建模在知识库中存在实体对应的实体提及。ENEL 模型结构图如12.16所示, 主要组成部分包括蕴含上下文的实体提及表示、实体嵌入向量和提及-实体的概念映射。

单词和字符嵌入向量 (Word and Char Embeddings) 用于构词向量表示, 字符的嵌入向量是由一个 BiLSTM 建模得到, 输入为一个单词的所有字符。单词的嵌入向量是预训练好的词向量。最后句子中的单词嵌入向量由其字符向量和词向量拼接得到, 表示为 $\{v_k\}_{k \in [1, n]}$, 这对应了图12.16中的单词-字母嵌入拼接。

实体提及表示 (Mention Representation) 用于建模上下文信息, ENEL 算法中采用用 BiLSTM 进行建模。

$$x_k = BiLSTM(v_k) \quad (12.15)$$

对每个可能的实体提及 $m = w_q, \dots, w_r$, 其软头嵌入向量 (Soft Head Embedding) 为:

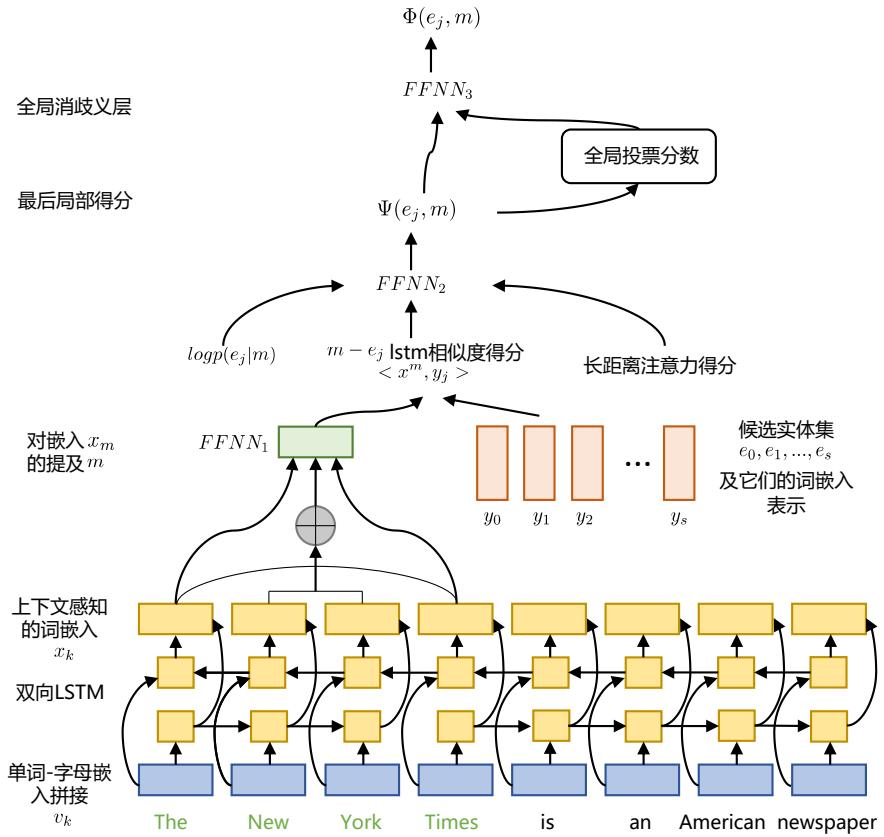
$$\alpha_k = \langle \mathbf{w}_\alpha, x_k \rangle \quad (12.16)$$

$$a_k^m = \frac{\exp(\alpha_k)}{\sum_{t=q}^r \exp(\alpha_t)} \quad (12.17)$$

$$\hat{x}^m = \sum_{k=q}^r a_k^m \cdot v_k \quad (12.18)$$

实体提及的最终表示 \mathbf{g}^m 为序列的首尾词的上下文表示向量和其软头嵌入向量的拼接:

$$\mathbf{g}^m = [x_q; x_r; \hat{x}^m] \quad (12.19)$$

图 12.16 ENEL 模型结构图^[26]

实体嵌入向量 (Entity Embedding) 采用文献 [27] 中基于预训练得到的实体表示向量, 用 y_e 来表示。所采用的方法是将单词和实体名映射到同一个空间中, 首先利用单词的向量表示初始化实体表示, 再根据实体和单词共现关系用于训练实体表示。

候选实体词选择 (Candidate Selection) 用于选择可能的候选实体。ENEL 采用文献 [27] 给出的经验概率映射的方法, 对每个可能的序列 m , ENEL 最多选择 s 个候选实体, 候选实体集合用 $C(m)$ 表示。

实体-提及对得分 (Entity-Mention Score), 对每个可能成为实体指代的序列 m , 也即 $|C(m)| \geq 1$, 对其计算局部得分。首先对实体候选集合里的每个实体与提及序列, 通过点积计算相似度, 再结合实体候选集合计数中得到的候选概念对数, 输入到一个全连接层得到最终得分。

$$\Psi(e_j, m) = FFNN_2([\log p(e_j | m); \langle x^m, y_j \rangle]) \quad (12.20)$$

为了进一步提升性能, ENEL 还可以引入全局依赖, 使用上下文嵌入与候选实体做相似度计算, 将得到的结果加入公式12.20中。

全局消歧 (Global Disambiguation) 对局部得分高的实体-提及对全局的实体-提及得分进行投票。由于实体-提及得分只建模了一对的信息, 为了引入多对实体-提及信息, ENEL 增加了全局消歧层。

$$V_G = \{(m, e) \mid m \in M, e \in C(m), \Psi(e, m) \geq \gamma'\} \quad (12.21)$$

$G(e_j, m)$ 表示由实体嵌入与所有其他投票实体嵌入的平均值之间的余弦相似度, $\Phi(e_j, m)$ 表示最终的全局得分。

$$V_G^m = \{e \mid (m', e) \in V_G \wedge m' \neq m\} \quad (12.22)$$

$$y_G^m = \sum_{e \in V_G^m} y_e \quad (12.23)$$

$$G(e_j, m) = \cos(y_{e_j}, y_G^m) \quad (12.24)$$

$$\Phi(e_j, m) = \text{FFNN}_3([\Psi(e_j, m); G(e_j, m)]) \quad (12.25)$$

结合实体-提及的局部得分和全局得分, 可以得到最终的训练目标:

$$\theta^* = \arg \min_{\theta} \sum_{d \in D} \sum_{m \in M} \sum_{e \in C(m)} V(\Psi_{\theta}(e, m)) + V(\Phi_{\theta}(e, m)) \quad (12.26)$$

其中 V 用来强制正确的分数与错误对的分数线性可分:

$$V(\Psi(e, m)) = \begin{cases} \max(0, \gamma - \Psi(e, m)), & \text{if } (e, m) \in \mathcal{G} \\ \max(0, \Psi(e, m)), & \text{其他} \end{cases} \quad (12.27)$$

12.3.3 实体对齐

知识图谱包含描述抽象知识的本体层和描述具体事实的实例层。本体层一般为抽象知识, 如概念、属性、公理等。实例层则描述具体实体、实体间的关系, 即事实数据。知识工程早期希望可以构建一个包含所有知识的统一的知识库, 但是在实际情况下面临很多困难。首先就是不同领域的知识存在差异性, 同时不同专家对知识定义的主观性, 使得无法构建统一的本体知识。其次, 知识本身会随着时间发生演变, 在不同时间, 使用者会构建不同的知识本体, 对相同实体的指代称呼也会发生变化, 如图12.17所示。这种现象称之为本体和实例的异构性。

针对知识图谱的异构性问题, 研究人员们提出了知识融合技术, 其目标就是为不同知识图谱的实例或者对象建立关联, 可以高效合并多个不同来源的知识图谱。知识融合技术的两个基本任务是本体匹配和实体对齐, 两个任务的目的类似, 只是匹配对象不同, 前者用来匹配本体层的抽

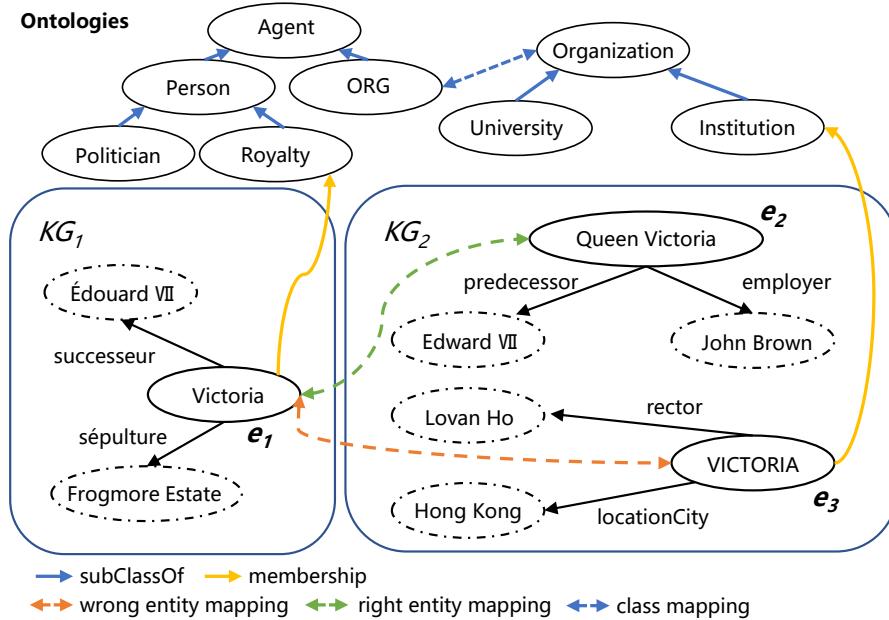


图 12.17 知识图谱融合样例图

象知识，后者用来匹配实例层中相同对象的不同实体指代。两者使用的技术有很多类似之处，鉴于知识图谱的实例规模通常远大于本体规模，任务难度更大，本节将重点介绍实体对齐任务。

实体对齐（Entity Alignment）也被称作实体匹配（Entity Matching），旨在发现多源知识图谱中等价的实体对。比如由几种不同的语言构建的知识库，实现跨语言知识对齐将帮助人们构建一个连贯的知识库，并帮助机器处理跨语言的实体关系的不同表达。早期的实体对齐研究一般基于实体属性的相似度计算。这种方法依赖于用户定义的规则来决定实体之间进行比较的属性。例如，实体的类别、实体的邻居类别等。因为不同的实体对可能需要不同的属性来进行比较，使得依赖用户定义规则的方法很容易出错，并且算法的迁移性较差，已有的规则难以在其他场景直接应用。

随着深度神经网络在自然语言处理领域的研究不断深入，基于表示学习的实体对齐方法逐渐成为目前的主流方法。本章第12.2.2节介绍了知识的向量表示，简单来说就是将三元组中的语义信息投影到稠密的低维向量空间，构造实体和关系的分布式表示向量。基于表示学习的实体对齐，核心思想就是在低维空间中用向量的距离来计算不同来源的实体之间的相似度。下面将介绍两类基于表示学习的实体对齐方法：基于平移的方法和基于图神经网络的方法。

1. 基于平移的实体对齐方法

虽然基于平移的知识表示技术可以帮助提高单语言知识的完整性，但是该技术并不能直接解决跨语言知识的对齐问题。主要有以下几个方面的原因：(1) 跨语言平移比任何单语关系平移的范围都要大得多；(2) 实体和关系在不同语言之间的词汇表达不连贯；(3) 用于训练知识嵌入表示的

已知对齐关系通常只占知识库的一小部分。

为了解决以上问题，文献[28]提出 MTransE 算法，通过在独立的嵌入空间中编码每种语言的实体和关系，为每个嵌入向量提供到其他语言空间中对应向量的过渡，同时保留单语言嵌入的功能。MTransE 的核心假设是相同实体在不同来源的知识图谱里存在类似的分布。MTransE 基本框架如图12.18所示，主要包含两个模块：平移模块和对齐模块。平移模块在特定于语言的知识图谱中编码实体和关系，采用 TransE 作为知识嵌入模型。在此基础上，对齐模块学习跨语言的实体和跨不同嵌入空间的关系对齐。

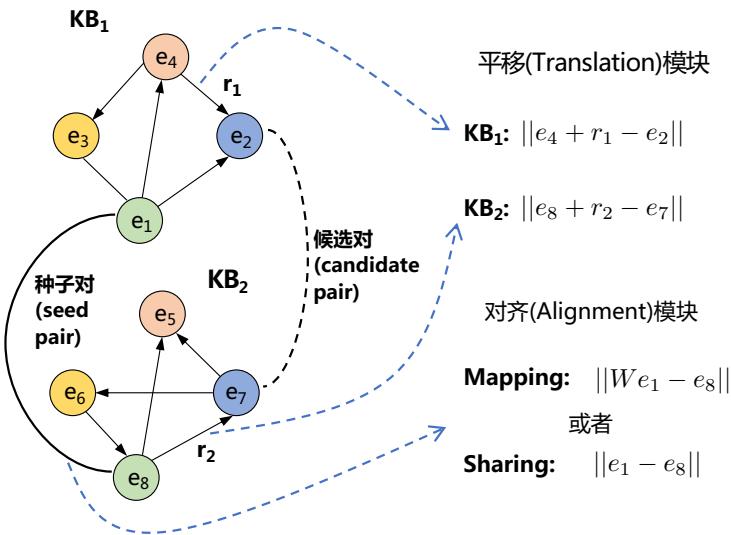


图 12.18 MTransE 算法示意图^[28]

给定知识库，用 \mathcal{L} 表示语言集合， \mathcal{L}^2 表示两种语言对集合。对于某种语言 $L \in \mathcal{L}$ ，其语言特定的知识图谱为 G_L ，其对应的实体和关系集合表示为 E_L 和 R_L 。知识图谱中三元组 $T = < h, r, t >$ 对应的嵌入向量分别为 h, r, t 。 $(L_1, L_2) \in \mathcal{L}^2$ 表示由 L_1 和 L_2 两种语言组成的语言对， $\delta(L_1, L_2)$ 表示语言对中对齐的三元组集合。

平移模块的主要功能是，通过基于平移的知识图谱嵌入模型，将随机初始化的实体嵌入约束为固定分布。TransE^[14] 将关系解释为从头实体到尾实体的平移，因此其实体嵌入向量也具有平移不变性。MTransE 对每一种涉及的语言都采用了基于翻译的 TransE 方法，通过在不同的关系上下文中统一表示嵌入，有利于跨语言任务。因此其损失函数为：

$$S_K = \sum_{L \in \{L_i, L_j\}} \sum_{< h, r, t > \in G_L} \|h + r - t\| \quad (12.28)$$

由于知识库按照语言被划分为互不关联的子集，因此平移模块可以采取并行训练。

在实体对齐阶段，已有的知识库来源不同但是指代相同的实体对会作为初始种子，帮助对齐模块将不同知识图谱的实体嵌入映射到统一的向量空间进行对齐。其损失函数为：

$$S_A = \sum_{(T, T') \in \delta(L_i, L_j)} S_a(T, T') \quad (12.29)$$

对齐得分 S_a 的计算方法有两种，映射对齐和共享表示对齐：

- (1) 映射对齐方法通过线性变化，将来源不同的知识嵌入向量映射到一个统一的向量空间。如图12.18所示，优化的对象就是统一向量空间中的实体对之间的距离，如 $\|We_1 - e_8\|$ 。
- (2) 共享表示对齐是通过让每个预对齐的实体对，直接共享相同的嵌入，将不同的知识图谱嵌入到统一的向量空间中，这比映射方法更加直接。如图12.18所示，优化的对象直接就是初始嵌入空间的向量距离，如 $\|e_1 - e_8\|$ 。

在实际训练时，MTransE 采用 S_K 和 S_A 交替优化的方式实现。

2. 基于图神经网络的实体对齐方法

基于平移的知识嵌入模型实现的实体对齐模型，需要小心调节平移模块和对齐模块的损失权重，比较难优化。而且基于平移的方法的训练目标只包含单独的三元组，而实体的属性信息（例如人的年龄、城市的人口数）并没有被有效利用，所以其并不能从全局视角捕获实体和关系的信息。因此，文献 [29] 提出了 GCNAlign 算法，引入图卷积神经网络来建模全局信息，其模型结构如图12.19所示。GCNAlign 网络包含两个图神经网络，分别将不同源的知识图谱实体编码到一个统一的向量空间，然后通过对比损失或者三元组损失优化它们之间的距离。

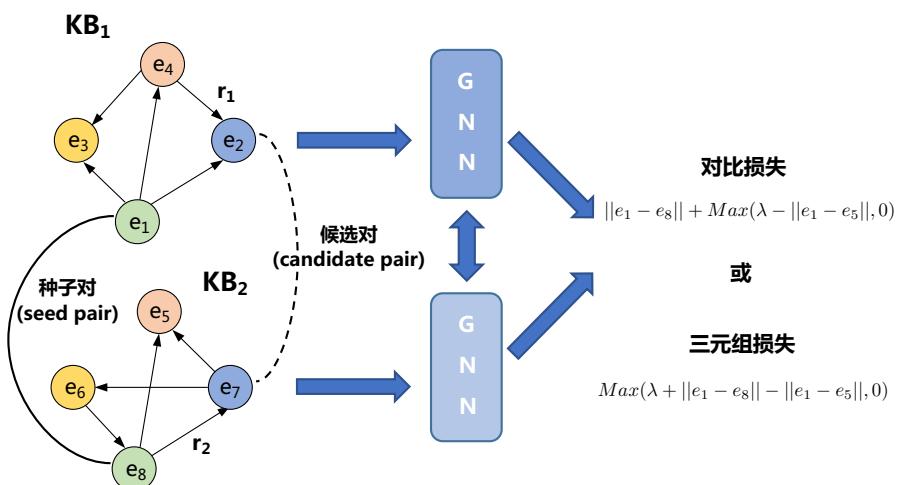


图 12.19 基于图神经网络的实体对齐方法

具体来说, GCNAlign 考虑两种知识图谱中的三元组: 关系三元组和属性三元组。关系三元组建模实体之间的关系, 例如<爱因斯坦, 毕业于, 苏黎世联邦理工学院>, 而属性三元组描述实体具备的属性, 例如<爱因斯坦, 享年, 76 岁>。因此可以将一个知识图谱表示为 $G = (E, R, A, T^R, T^A)$, 其中 E, R, A, V 表示实体、关系、属性名和属性值集合; $T_R \subset E \times R \times E$ 为关系三元组集合, $T_A \subset E \times A \times V$ 为属性三元组。给定两个知识图谱 $G_1 = (E_1, R_1, A_1, T_1^R, T_1^A)$ 和 $G_2 = (E_2, R_2, A_2, T_2^R, T_2^A)$, 用 $S = \{(e_{i_1}, e_{i_2}) \mid e_{i_1} \in E_1, e_{i_2} \in E_2\}_{i=1}^m$ 表示已知的对齐的实体对。因此跨语言实体对齐可以看作, 根据已有对齐实体对发现新的实体对的任务。GCNAlign 主要包含三部分内容: 基于图卷积网络的实体嵌入学习、对齐实体的预测和模型训练方法。

基于图卷积网络的实体嵌入学习: GCNAlign 的核心假设是: (1) 等价的实体一般拥有类似的属性; (2) 等价的实体其邻居实体一般也等价。图卷积神经网络 (GCN)^[30] 是一种直接对图数据进行操作的神经网络, 其输入是节点的特征向量和图的结构。GCN 的目标是学习输入图上特征的函数, 并产生节点级输出。因此, GCNAlign 使用 GCN 将属性信息和结构信息结合在一起, 把实体投影到低维向量空间中, 并约束在低维向量空间中等价实体之间彼此接近。

GCNAlign 的输入是一个节点特征矩阵, $\mathbf{H}^{(l)} \in \mathbb{R}^{n \times d^{(l)}}$, 其中 n 为顶点个数, $d^{(l)}$ 为特征个数。GCN 的输出是一个新的特征矩阵 $\mathbf{H}^{(l+1)}$, 图卷积的计算如下:

$$\mathbf{H}^{(l+1)} = \sigma \left(\hat{\mathbf{D}}^{-\frac{1}{2}} \hat{\mathbf{A}} \hat{\mathbf{D}}^{-\frac{1}{2}} \mathbf{H}^{(l)} \mathbf{W}^{(l)} \right) \quad (12.30)$$

其中 σ 为 ReLU 激活函数, \mathbf{A} 为 $n \times n$ 的邻接矩阵, 用来表达图的结构信息, $\hat{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, \mathbf{I} 为单位矩阵; $\hat{\mathbf{D}}$ 是 $\hat{\mathbf{A}}$ 的对角节点度矩阵; $\mathbf{W}^{(l)} \in \mathbb{R}^{d^{(l)} \times d^{(l+1)}}$ 为 GCN 的参数矩阵, $d^{(l+1)}$ 为新节点特征的维度。

为了同时利用知识图谱中的结构信息和实体的属性信息, GCNAlign 建模两种特征: 结构特征 h_s 和属性特征 h_a 。令 \mathbf{H}_s 和 \mathbf{H}_a 分别为结构特征矩阵和属性特征矩阵, 因此 GCNAlign 的卷积计算方式为:

$$\left[\mathbf{H}_s^{(l+1)}; \mathbf{H}_a^{(l+1)} \right] = \sigma \left(\hat{\mathbf{D}}^{-\frac{1}{2}} \hat{\mathbf{A}} \hat{\mathbf{D}}^{-\frac{1}{2}} \left[\mathbf{H}_s^{(l)} \mathbf{W}_s^{(l)}; \mathbf{H}_a^{(l)} \mathbf{W}_a^{(l)} \right] \right) \quad (12.31)$$

其中 $\mathbf{W}_s^{(l)}$ 和 $\mathbf{W}_a^{(l)}$ 为结构特征的权重矩阵和属性特征的权重矩阵, $[;]$ 表示矩阵的拼接操作。

GCNAlign 使用了两个两层的 GCN, 分别建模两种语言的嵌入表示。知识图谱是个有向图, 且实体之间通过代表不同关系的边相连接, 因此 GCNAlign 设计了一种特定的邻接矩阵 \mathbf{A} 的方法。另 $a_{ij} \in \mathbf{A}$ 表示从第 i 个实体传播到第 j 个实体的程度信息。考虑到两个实体通过不同的关系 (例如, hasParent 与 hasFriend) 连接到对齐的实体, 两个实体等价的概率差别很大。因此, 为每个关

系计算功能性和逆功能性两种度量：

$$\begin{aligned} \text{fun}(r) &= \frac{\# \text{关系 } r \text{ 中头实体数量}}{\# \text{关系 } r \text{ 三元组数目}} \\ \text{ifun}(r) &= \frac{\# \text{关系 } r \text{ 中尾实体数量}}{\# \text{关系 } r \text{ 三元组数目}} \end{aligned} \quad (12.32)$$

基于此第 i 个实体对第 j 个实体的影响 $a_{ij} \in \mathbf{A}$ 计算方式如下：

$$a_i = \sum_{\langle e_i, r, e_j \rangle \in G} \text{ifun}(r) + \sum_{\langle e_j, r, e_i \rangle \in G} \text{fun}(r) \quad (12.33)$$

对齐实体的预测：通过将预定义的距离函数应用于实体的 GCN 表示来预测实体对齐。对于来源于知识图谱 G_1 的实体 e_i 和来源于知识图谱 G_2 的实体 v_j ，两者的距离计算方式如下：

$$D(e_i, v_j) = \beta \frac{f(\mathbf{h}_s(e_i), \mathbf{h}_s(v_j))}{d_s} + (1 - \beta) \frac{f(\mathbf{h}_a(e_i), \mathbf{h}_a(v_j))}{d_a} \quad (12.34)$$

其中 $f(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_1$ ，而 $\mathbf{h}_s(\cdot)$ 和 $\mathbf{h}_a(\cdot)$ 表示实体的结构嵌入向量和属性嵌入向量， d_s 和 d_a 表示结构嵌入向量和属性嵌入向量的维度， β 则为平衡两者重要性的超参数。

对于对齐的实体，两者距离应该较小，对于非对齐实体，该距离预计较大。给定 G_1 中的特定实体 e_i ，计算 e_i 与 G_2 中的所有实体之间的距离，并返回排序实体列表作为候选对齐，也可以从 G_2 到 G_1 进行对齐。

模型训练：为了使 GCN 在向量空间中嵌入尽可能接近的等价实体，GCNAlign 使用一组已知的实体对齐集合 S 作为训练数据来训练 GCN 模型。通过最小化以下基于边际的排序损失函数来进行模型训练：

$$\begin{aligned} \mathcal{L}_s &= \sum_{(e, v) \in S} \sum_{(e', v') \in S'_{(e, v)}} [f(\mathbf{h}_s(e), \mathbf{h}_s(v)) + \gamma_s - f(\mathbf{h}_s(e'), \mathbf{h}_s(v'))]_+ \\ \mathcal{L}_a &= \sum_{(e, v) \in S} \sum_{(e', v') \in S'_{(e, v)}} [f(\mathbf{h}_a(e), \mathbf{h}_a(v)) + \gamma_a - f(\mathbf{h}_a(e'), \mathbf{h}_a(v'))]_+ \end{aligned} \quad (12.35)$$

其中 $[x]_+ = \max\{0, x\}$ ， $S'_{(e, v)}$ 表示由非对齐实体对的负例集合，比如将实体 e 或实体 v 替换为 G_1 或 G_2 中的一个随机实体； $\gamma_s, \gamma_a > 0$ 为分离正负例的间隔超参。 \mathcal{L}_s 和 \mathcal{L}_a 是结构嵌入和属性嵌入的损失函数，它们互相独立，可以分开优化。

12.4 知识图谱推理

推理是从一个或几个已有判断中推导出新判断的过程，是人类有别于普通物种的重要能力之一。推理能力是实现通用人工智能的最终目标中必不可少的路径，而推理过程必须依赖于先验知识和已有经验。常见的推理方法如图所示12.20可以分为三类演绎推理，归纳推理和溯因推理：

- 演绎推理（Deductive Reasoning）是一种自上而下的推理方法，即根据已知的一般或普遍性前提推理得到必然成立的结论。比如，已知“没有电力供应电脑就不会正常工作”，同时知道“今天停电了”，那么就可以推理出“今天没有办法正常使用电脑”。
- 归纳推理（Inductive Reasoning）是一种自下而上的推理方法，即根据观察所得到客观知识进行总结归纳，从而得到更一般的结论。比如，观察发现看到的电脑都安装了 Windows，于是归纳出所有的电脑使用的都是 Windows 操作系统。显然，这种归纳未必一定正确，因为还有使用 Linux 和 MacOS 的电脑。生活中这种推理方式十分有用，因为观察样本足够多时，人们可以通过这种方式推理出大概率会发生的事件。
- 漂因推理（Abductive Reasoning）是从现象出发的推理方法，即结合已有的观察和知识，推断出有可能的解释过程。假如已知知识“没有电力供应电脑就不会正常工作”，同时观察到电脑没法正常启动，我们推断出“可能是因为停电了”。

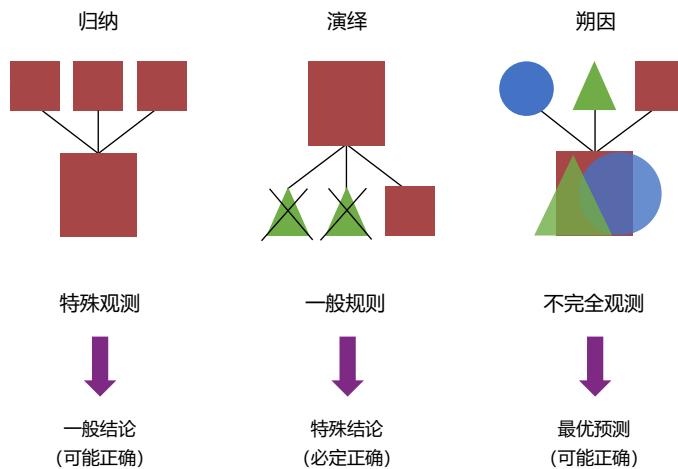


图 12.20 演绎推理、归纳推理和溯因推理示例

基于知识图谱的知识推理（Knowledge Reasoning），旨在从图谱中已存在的关联关系或事实推断出未知的关系或事实。推理得到的知识又可以反过来丰富知识图谱，为下游的应用提供更好的支持。随着知识图谱的普遍应用，基于知识图谱的推理受到了工业界和学术界的广泛关注。从推

理方法的角度，知识图谱推理方法可以分为两类，一种是演绎推理，主要是基于符号逻辑的知识图谱推理，依赖于专家显示制定的知识描述和逻辑推导方法；另一种是归纳推理，代表方法为基于表示学习的知识图谱推理，更多地依赖于大规模数据和统计学习算法。本章将从这两个类别对已有的知识图谱推理方法展开介绍。

12.4.1 基于符号逻辑的知识图谱推理

包括本体推理在内的早期知识推理方法受到了广泛关注，并产生了一系列推理方法，本节将展开介绍其中的三种：基于本体的推理、基于 Datalog 的推理以及基于产生式规则的推理。

1. 基于本体的推理

一个定义完备的知识图谱包含两部分知识：(1) TBox (Theory Box)，即是关于概念术语的断言，对应 Schema 层，主要定义概念以及关系；(2) ABox (Assertion Box)，是关于个体的断言，关于事实性的描述。如图12.21所示。基于本体的推理方法，一般在 TBox 层定义本体公理 (Ontological Axioms)，然后根据演绎推理的方法推断出新的事实。

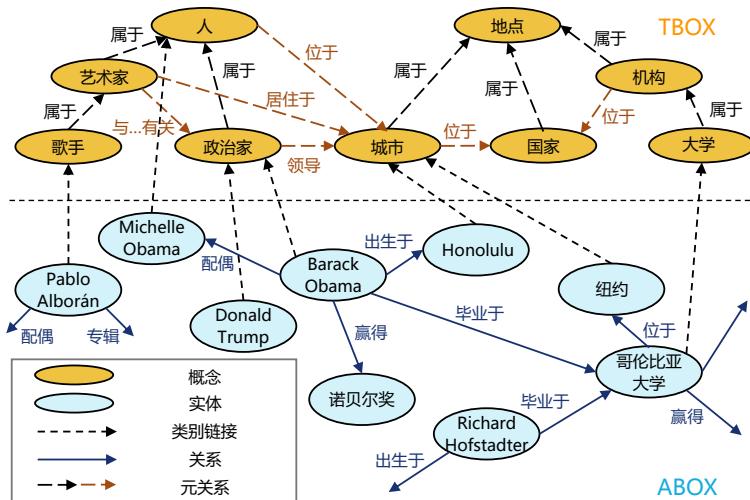


图 12.21 知识图谱中的 TBox 和 ABox 示例

本章第12.2.1节中介绍了 RDF、RDFS 和 OWL，其中 RDF 中的三元组对应客观世界的逻辑事实，基于 RDFS 描述语言的表达构件，如父子类关系，可以实现比较简单的推理。OWL 在 RDFS 进一步扩展了 Schema 的表达能力，可以完成更复杂的本体逻辑推理，如表12.7所示。

基于 OWL 的基础语法和公理基本语法，知识图谱可以实现一定的推理能力。例如：TBox 中定义“人工智能公司 subclass 高科技公司”，ABox 中存在知识“谷歌 type 人工智能公司”，那么就可以推断出“谷歌 type 高科技公司”；又比如在 OWL 本体中定义“父亲的父亲是爷爷”，已知“司马懿

表 12.7 OWL 语言基本语法

构造算子	语法	语义	例子
原子概念	A	$A^I \subseteq \Delta^I$	Human
原子关系	R	$R^I \subseteq \Delta^I \times \Delta^I$	has_child
对概念 C, D 和关系 (role) R			
合取	$C \sqcap D$	$C^I \cap D^I$	Human \sqcap Male
析取	$C \sqcup D$	$C^I \cup D^I$	Doctor \sqcup Lawyer
非	$\neg C$	$\Delta^I \setminus C^I$	\neg Male
存在量词	$\exists R.C$	$\{x \exists y. \langle x, y \rangle \in R^I \wedge y \in C^I\}$	\exists has_child. Male
全称量词	$\forall R.C$	$\{x \forall y. \langle x, y \rangle \in R^I \wedge y \Rightarrow C^I\}$	\forall has_child. Doctor

是司马昭的父亲”和“司马昭是司马炎的父亲”，可以推断出“司马懿是司马炎的爷爷”。

OWL 本体推理的核心算法为 Tableaux 算法，其基本方法是通过一系列规则构建 ABox，然后检查知识库的可满足性。Tableaux 是一颗公式树，它会根据前提和否定结论来不断创建分支，对公式进行逐级分解，当所有分支都关闭后，Tableaux 算法就会终止。简单来说，如果要证明一件事是正确的，只要将其所有反例驳斥即可达到目的。

Tableaux 算法的描述逻辑算子如表12.8所示，以第一条规则为例，如果 ABox 中声明 x 属于 C 和 D 的组合类，但是 $C(x)$ 和 $D(x)$ 并不在 ABox 中，则把 $C(x)$ 和 $D(x)$ 都加入到 ABox 中。

表 12.8 Tableaux 算法规则

- \sqcap^+ -规则：若 $C \sqcap D(x) \in \emptyset$ ，且 $C(x), D(x) \notin \emptyset$ ，则 $\emptyset := \emptyset \cup \{C(x), D(x)\}$ ；
- \sqcap^- -规则：若 $C(x), D(x) \in \emptyset$ ，且 $C(x) \sqcap D(x) \notin \emptyset$ ，则 $\emptyset := \emptyset \cup \{C(x), D(x)\}$ ；
- \exists -规则：若 $\exists R.C(x) \in \emptyset$ ，且 $R(x, y), C(y) \notin \emptyset$ ，则 $\emptyset := \emptyset \cup \{R(x, y), C(y)\}$ ，其中， y 是新加进来的个体；
- \forall -规则：若 $\exists R.C(x), R(x, y) \in \emptyset$ ，且 $C(y) \notin \emptyset$ ，则 $\emptyset := \emptyset \cup \{C(y)\}$ ；
- \sqsubseteq -规则：若 $C(x) \in \emptyset, C \sqsubseteq D$ ，且 $D(x) \notin \emptyset$ ，则 $\emptyset := \emptyset \cup \{D(x)\}$ ；
- \perp -规则：若 $\perp(x) \in \emptyset$ ，则拒绝 \emptyset ；

实现 Tableaux 算法的推理系统包括曼彻斯特大学研发的 FaCT++^[31]、美国 Franz 公司研发的 Racer^[32]、马里兰大学研发的 Pellet^[33]、牛津大学研发的 HermiT^[34] 等，具体细节可以参考相关资料。

2. 基于 Datalog 的推理

基于本体的知识推理只能基于本体概念描述推理，不支持规则型知识的推理。此外，用户无法定义自己的推理过程。Datalog 则用于解决逻辑问题，将本体推理和规则推理相结合。Datalog 是一种基于逻辑编程语言 Prolog 且适应于知识库的改进型语言，能够方便地和大型数据库进行交互，便于撰写规则实现推理。

Datalog 的基本组成是原子 $p(t_1, t_2, \dots, t_n)$ ，其中 p 是谓词， t_i 是变量或者常量。例如：has_parent(X,

Y), 其中“`has_parent`”是谓词, “ X, Y ”是常量。一条规则 $H:-B_1, B_2, \dots, B_m$ 由一个头部原子 H 和多个体部原子 B_1, B_2, \dots, B_m 构成, 表示当体部原子成立时, 可以得到头部原子成立的结论。

例如: `has_parent(X,Y):-has_mother(X,Y)`

表示“`has_mother(X,Y)`”蕴含了“`has_parent(X,Y)`”

除了规则外, Datalog 还包含了大量事实知识 $F(c_1, c_2, \dots, c_n) :-$, 即没有体部且没有变量的规则。

例如: `has_mother(X,Y):-`

Datalog 程序本质上就是规则的集合, 例如: 已知规则集合“坐落于 (X, Y):-同城 (X, Z), 坐落于 (Z, Y)”, 和事实集合“同城 (复旦大学, 华东师范大学):-; 坐落于 (华东师范大学, 上海):-”, 基于第一条规则, 第一条和第二条事实, 可以推理出新的事实“坐落于 (复旦大学, 上海)”。在实际应用中, 事实集合可能非常大, 随着规则集合越来越大, 推理的开销也会显著升高。

最常用的 Datalog 工具包括德国吉森大学研发的 DLV (Datalog with Disjunction)^[35]、波茨坦大学研发的 Clingo^[36]、牛津大学研发的 RDFox^[37]等, 具体细节可以参考相关资料。

3. 基于产生式规则的推理

产生式规则推理方法是一种前向推理系统, 从已知事实出发, 通过一定规则求得结论, 类似于一阶逻辑。其基本组成包含: Working Memory (WM) 中的事实集合, 产生式规则集合以及推理引擎。WM 中存储的事实数据包括两种, 一是用来描述对象, 例如: `type attr1:val1 attr2:val2 ... attrn:valn`, 其中 `type`, `attri`, `vali` 均为原子; 另一种用来描述关系, 例如: `basicFact relation: olderThan firstArg:John secondArg:Alice`。

产生式规则集合中存储的规则定义为 “IF conditions THAN actions” 形式, “conditions”对应条件集合, 又称为 LHS (Left Hand Side), “actions”对应动作序列, 又称为 RHS (Right Hand Side)。当 LHS 中所有条件都被满足的时候, 那么该规则触发, 执行相应的动作。

例如: IF (Gauge state: OK) AND (TEMPERATURE > 120) THEN (Cooling system state: over-heating)

表示: 当测量器状态正常且测量温度超过 120 度, 那么制冷系统的工作状态为过载

产生式规则的执行由推理引擎控制, 其核心任务是模式匹配。这个过程参考关系型数据库做知识图谱的存储和查询, 不同的条件匹配会产生大量的连接操作, 是个组合爆炸问题。此外, 还要考虑当满足多个触发规则时, 选择哪条规则执行的问题, 这被称为冲突解决。在选定规则时, 则执行规则的 actions, 对应 WM 中事实数据的增删改。

Rete 算法^[38]是产生式规则系统中常用的推理算法, 其核心思路是逐渐扩充条件集合, 同时缓存条件匹配的中间结果, 直到所有规则对应的条件集合都得到判断, 是一种空间换时间的优化方法。如图12.22所示, Rete 算法分为 α 网络和 β 网络。 α 网络中的节点对应每条 condition, β 网络的节点对应 α 网络中节点 Join 后的 condition 集合。Rete 算法会先将 WM 中的事实先和 α 网络中的节点进行匹配, 然后对满足条件的事实输入到 β 网络的节点中进行校验, 直到最后筛选出所有至少满足一条规则的条件事实组合, 完成推理。可见, 若 Rete 算法中的节点被 N 条规则共享, 即

对应推理速度加速 N 倍。

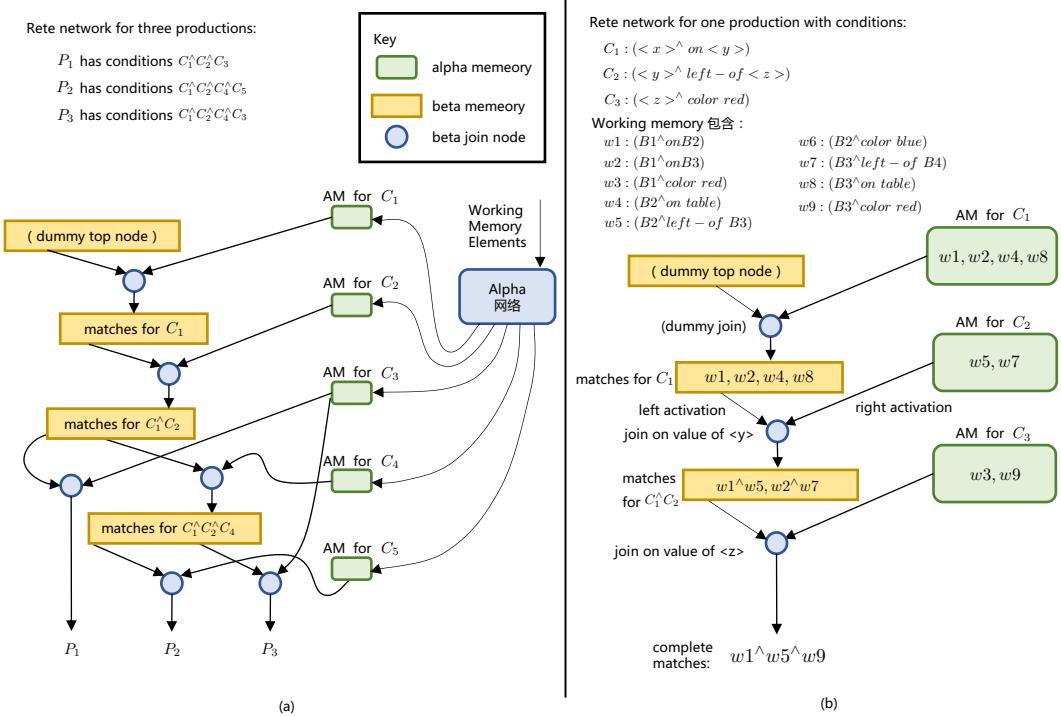


图 12.22 Rete 算法样例^[38]

12.4.2 基于表示学习的知识图谱推理

基于符号表示的推理本质上基于演绎逻辑推理，精准度高但是也依赖于专家制定描述规则，当数据量较大时，推理的鲁棒性和效率都会显著降低，这种情况下基于统计的归纳学习推理有着不可替代的优势。随着神经网络研究的深入，基于深度神经网络表示学习的推理方法来逐渐成为研究的重点。本节将展开介绍两种基于表示学习的知识图谱推理方法。

1. 基于神经张量网络的知识图谱推理算法

知识图谱推理可以归结为根据知识库中已有事实和关系来推断新的事实的问题，包括给定两个实体，判断它们之间是否存在某种关系；给定头实体和关系，预测尾实体；给定三元组，判断其为真或假。基于已经训练好的知识表示，也可以直接应用于简单的三元组推理任务，例如，本章第12.2.2节中已经介绍过知识图谱的向量表示方法，包括 TransE、TransR 等。

不同于 TransE 等方法给每个实体、关系单独学习一个表示向量，文献 [39] 提出使用神经张量网络（Neural Tensor Network，NTN）来分类实体对之间的关系，并使用构成实体的词向量的平均

作为该实体表示。NTN 整体流程如图12.23所示。

具体来说,假设“Sumatran tiger”是知识图谱中的实体,NTN 会给构成其的词“Sumatran”和“tiger”分别学习一个嵌入向量,然后用两者的平均作为其表示。这么做的原因是在词向量的表示空间中,概念相似的词的嵌入向量距离也较近^[40]。词向量的这种特性有助于提高具有公共子串的实体词之间共享统计强度,比如“Sumatran tiger” 和 “Bengal tiger”拥有共同的子串“tiger”,它们之间应该具有很高的关系性。在得到两个实体的表示向量(e_1, e_2)之后, NTN 使用神经张量网络来计算它们之间是否存在某种关系 R 的置信度。为此, NTN 为每个关系单独定义了一组参数用于建模实体对之间的语义关联。

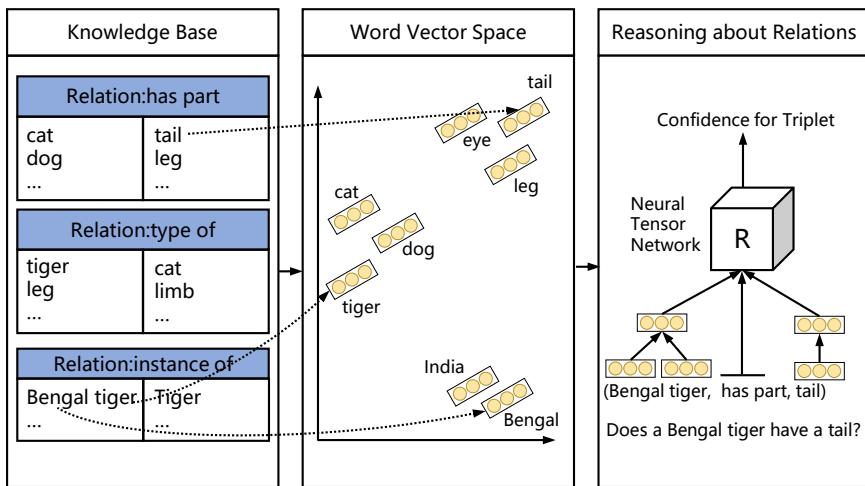


图 12.23 NTN 模型概述^[39]

NTN 算法的核心是关系分类模块,该模块用来判断实体对(e_1, e_2)是否存在某种特定的关系 R 。假设知识库中仅建模两种关系,不同于以往的方法直接将实体对的表示拼接在一起, NTN 采用双线性神经层 W_R 直接建模实体对之间的语义关系。NTN 输出的关系置信度打分,通过以下公式计算:

$$g(e_1, R, e_2) = \mathbf{u}_R^T f \left(\mathbf{e}_1^T \mathbf{W}_R^{[1:k]} \mathbf{e}_2 + \mathbf{V}_R \begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \end{bmatrix} + \mathbf{e}_R \right) \quad (12.36)$$

其中 $f = \tanh$ 为激活函数, $\mathbf{W}_R^{[1:k]} \in \mathbb{R}^{d \times d \times k}$ 对应不同关系的切片参数,例子中仅存在两种关系,所以 $k = 2$ 。双线性张量点积 $\mathbf{e}_1^T \mathbf{W}_R^{[1:k]} \mathbf{e}_2$ 会生成向量 $\mathbf{h} \in \mathbb{R}^k$,其不同维度对应了不同关系的参数计算结果,如第 i 个关系使用第 i 个参数切片计算: $\mathbf{h}_i = \mathbf{e}_1^T \mathbf{W}_R^{[i]} \mathbf{e}_2$ 。关系 R 的其他参数还包括, $\mathbf{V}_R \in \mathbb{R}^{k \times 2d}$, $\mathbf{U} \in \mathbb{R}^k$ 和 $\mathbf{b}_R \in \mathbb{R}^k$ 。

使用双线性神经层的好处是，它可以使乘法直接关联两个实体表示，而不是像标准神经网络那样隐含地关联实体向量。直观地说，张量 \mathbf{W}_R 的每个切片负责一种类型的实体对或关系的实例化。例如，模型可以从词向量空间中学习到组成关系，比如 $(\text{dog}, \text{haspart}, \text{lag})$ 和 $(\text{car}, \text{haspart}, \text{engine})$ 。

模型的训练目标与 TransE 类似，即真实存在的三元组 $T^{(i)} = \langle e_1^{(i)}, R^{(i)}, e_2^{(i)} \rangle$ 相比随机实体替换得到的三元组 $T_c^{(i)} = \langle e_1^{(i)}, R^{(i)}, e_c \rangle$ 能够得到更高的分数。令 NTN 的所有参数表示为 $\Omega = \{\mathbf{u}, \mathbf{W}, \mathbf{V}, \mathbf{b}, \mathbf{E}\}$ ，NTN 的损失函数为：

$$\mathcal{L}(\Omega) = \sum_{i=1}^N \sum_{c=1}^C \max \left(0, 1 - g(T^{(i)}) + g(T_c^{(i)}) \right) + \lambda \|\Omega\|_2^2 \quad (12.37)$$

其中 N 为训练集中三元组的个数，正确的三元组打分与替换三元组打分之间的差值最多为 1，为每个三元组生成 C 个替换后的三元组。

NTN 为构成实体的每个词学习一个单独的表示，而非整个实体。比如实体词“Sumatran tiger”，其表示为 $\mathbf{v}_{\text{Sumatran tiger}} = 0.5\mathbf{v}_{\text{Sumatran}} + 0.5\mathbf{v}_{\text{tiger}}$ 。虽然可以从随机初始化的词向量为每个词学习一个最终表示，但是 NTN 还证明了使用无监督预训练的词向量初始化可以进一步提升模型性能，这是因为大量的无监督语料有助于词向量更准确地捕获句法和语义信息。

2. 基于复空间关系旋转的知识图谱推理

已有的知识图谱表示学习方法，一般是根据观察到的知识事实对知识图谱中缺失的关系进行建模和判断。在目标关系里存在一些特殊的关系，如对称关系（如婚姻关系）、反对称关系（如父子关系）、反转关系（如上位词和下位词关系），还有多个关系可以组合为一个新的关系（如母亲的丈夫是父亲）等。上三种形式的关系具体形式定义如下：

对称/反对称关系：一个关系 r 是对称的（反对称的），如果对任意的实体对 x, y ，存在：

$$r(x, y) \Rightarrow r(y, x) \quad (12.38)$$

反转关系：关系 r_1 和关系 r_2 是反转的，如果对任意的实体对 x, y ，存在：

$$r_2(x, y) \Rightarrow r_1(y, x) \quad (12.39)$$

组合关系：关系 r_1 是 r_2 和 r_3 的组合关系，如果对任意的实体 x, y, z ，存在：

$$r_2(x, y) \wedge r_3(y, z) \Rightarrow r_1(x, z) \quad (12.40)$$

此前的方法只是显式或是隐式地建模以上部分关系，无法做到对上述三种类型关系的准确推断。以 TransE 为例，给定三元组 $\langle \text{科比}, \text{婚姻关系}, \text{瓦内萨} \rangle$ ，由 $\mathbf{h}_{\text{科比}} + \mathbf{h}_{\text{婚姻关系}} = \mathbf{h}_{\text{Vanessa}}$ 以及 $\mathbf{h}_{\text{瓦内萨}} + \mathbf{h}_{\text{婚姻关系}} = \mathbf{h}_{\text{科比}}$ ，推断出 $\mathbf{h}_{\text{科比}} = \mathbf{h}_{\text{瓦内萨}}$ ，这明显与事实不符。

文献 [41] 提出 RotatE 知识表示学习的方法以实现对上述所有关系的准确推理。RotatE 是受到欧拉公式的启发，即 $e^{i\theta} = \cos \theta + i \sin \theta$ ，一个复数可以看作是在复平面上的一个旋转。相应地，RotatE 将实体和关系映射到复向量空间中，然后将关系看作是从头实体向量往尾实体向量的旋转。除了可以建模以上所有关系之外，RotatE 的扩展性也非常好，当知识图谱规模扩大时，RotatE 的时间复杂度和存储消耗都是线性的。

具体来说，给定三元组 $\langle h, r, t \rangle$ ，其中 $h, r, t \in \mathbb{C}^k$ 是表示向量，RotatE 希望表示向量之间满足 $t = h \circ r$ ，其中 \circ 表示按元素相乘。所以，对在复空间中表示向量的每一维，希望满足：

$$t_i = h_i r_i, |r_i| = 1 \quad (12.41)$$

其中 $h_i, r_i, t_i \in \mathbb{C}$, $|r_i| = 1$ 用于约束关系表示向量的模长。因此， r_i 的复数形式为 $e^{i\theta_{r,i}}$ ，它对应于围绕复平面的原点逆时针旋转 $\theta_{r,i}$ 弧度，仅影响复向量空间中实体表示向量的相位。

理论上，这样简单的复空间乘法操作可以有效建模上述所有关系。例如，一个关系 r 是对称的，当且仅当它的表示向量的每个元素，即 r_i 满足 $r_i = e^{0/i} = 1$ ；两个关系 r_1 和 r_2 是反转的，当且仅当它们的嵌入是共轭的： $r_2 = \bar{r}_1$ ；关系 $r_3 = e^{i\theta_3}$ 是其他两个关系 $r_1 = e^{i\theta_1}$ 和 $r_2 = e^{i\theta_2}$ 的组合当且仅当 $r_3 = r_1 \circ r_2$ （即 $\theta_3 = \theta_1 + \theta_2$ ）。

以 1 维的表示向量为例，对比 TransE 和 RotatE 方法，如图12.24所示。TransE 将关系向量建模为实线上的向量平移，可以建模反转关系、反对称关系和组合关系，但是无法建模对称关系。但是 RotatE 将关系向量表示为复平面上的旋转，除了 TransE 可以建模的关系之外，还可以另关系向量满足 $r_i = -1$ ，来使得其满足对称关系的性质。

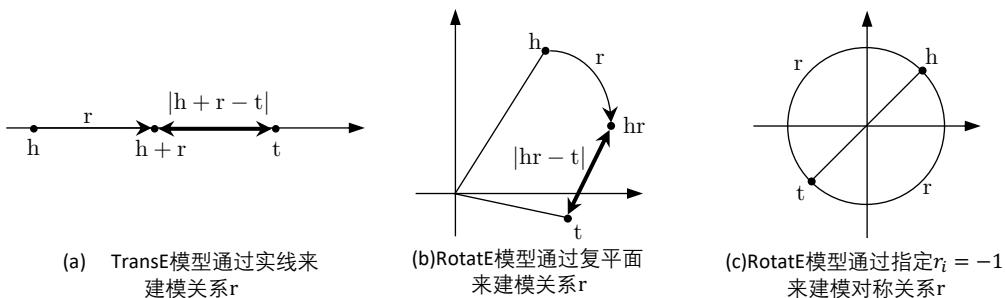


图 12.24 1 维表示向量上 TransE 与 RotatE 对比^[41]

类似于 TransE，RotatE 将三元组 $\langle h, r, t \rangle$ 的距离函数定义为：

$$d_r(h, t) = \|h \circ r - t\| \quad (12.42)$$

受 Word2Vec 中的负采样技术启发, RotatE 定义损失函数为:

$$\mathcal{L} = -\log \sigma(\gamma - d_r(\mathbf{h}, \mathbf{t})) - \sum_{i=1}^n \frac{1}{k} \log \sigma(d_r(\mathbf{h}'_i, \mathbf{t}'_i) - \gamma) \quad (12.43)$$

其中 γ 为固定边距, σ 是 sigmoid 函数, $\langle \mathbf{h}'_i, \mathbf{r}, \mathbf{t}'_i \rangle$ 为第 i 个负样本三元组, k 是表示向量的维度。

基于均匀负采样的三元组存在优化效率低下的问题, 因为随着训练的进行, 许多样本三元组明显是假的, 不能提供任何有意义的信息。因此, RotatE 提出了一种称为自对抗的负采样方法, 即根据现有的嵌入模型对负样本进行采样。具体来说, 负样本三元组被采样的概率被定义为:

$$p(h'_j, r, t'_j | \{ \langle h_i, r_i, t_i \rangle \}) = \frac{\exp \alpha f_r(\mathbf{h}'_j, \mathbf{t}'_j)}{\sum_i \exp \alpha f_r(\mathbf{h}'_i, \mathbf{t}'_i)} \quad (12.44)$$

其中 α 为温度系数。此外, RotatE 还将负样本的采样概率作为其权重加速训练。因此, 最终的自对抗训练采用如下目标作为最终损失函数:

$$\mathcal{L} = -\log \sigma(\gamma - d_r(\mathbf{h}, \mathbf{t})) - \sum_{i=1}^n p(h'_i, r, t'_i) \log \sigma(d_r(\mathbf{h}'_i, \mathbf{t}'_i) - \gamma) \quad (12.45)$$

12.5 知识图谱问答

在本书第??章中我们介绍了智能问答的历史渊源和各种方法, 本节将围绕知识图谱问答(Knowledge Base Question Answering, KBQA) 相关方法展开介绍, 知识图谱问答也称 Knowledge Graph Question Answering (KGQA)。知识图谱问答旨在以知识库为知识源来回答自然语言问题。与文本问答相比, 知识图谱的结构化知识能够提供更加精准的答案, 并且方便扩展。知识图谱问答在许多智能应用中发挥着重要作用, 引起了研究人员的广泛关注。例如, Amazon Alexa^[42]、Apple Siri^[43] 和 Microsoft Cortana^[44] 都集成了回答用户事实性问题的功能, 微软小冰和 Zo^[44] 等聊天机器人也展示了高度的对话能力, 可以回答很多事实性问题。

知识图谱问答方法大概可以分为两类: (1) 基于语义解析的方法, 即将自然语言组织的问句转化为知识库可以识别的结构化查询语句, 然后在知识库中查询得到答案。而这个问句到结构化查询的映射, 可以通过规则定义的模板或者训练过的自然语言解析器来完成; (2) 基于信息检索的方法, 通过对问句中的实体和关系识别锁定问题的主题实体, 然后根据主题实体得到知识图谱中的候选实体, 最后对候选进行排序得到最终答案。随着深度学习技术的成熟, 深度学习技术也开始在知识图谱问答中广泛应用, 用来提升语义解析和检索排序方法的效果。本节会对这几种方法以及改进, 依次进行介绍。

12.5.1 基于语义解析的知识图谱问答

基于语义解析的方法是通过对自然语言进行语义分析，将其转化为计算机可以处理的语义表示，进而利用知识库对问句进行推理查询，得到最后的答案。在第4章中我们介绍了多种语义表示方法，逻辑表达式就是其中一种常用于知识图谱问答的语义表示方法，如图12.25所示。自然语言问句“who is Justin Bieber’s brother?”转化为 Lamda 算子表示，谓词 sibling_of 表示兄弟姐妹关系，谓词 gender 表示性别。

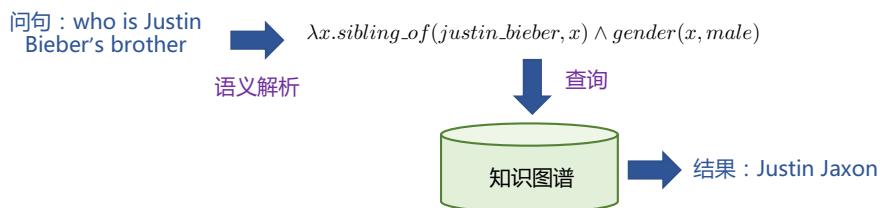


图 12.25 问句一步语义解析样例

直接将问句转化为语义表示的方法存在两个问题：一是非结构化的自然语言中许多的实体或是关系名称并不能直接匹配，例如图12.25示例文本中的“brother”和知识图谱中的关系描述“sibling_of”不能直接匹配；此外，自然语言是多样化和口语化的，模板和知识库无法覆盖所有描述，难以将所有自然语言问句进行转化。

另外一种方式采用两步解析，首先将自然语言问句转化成一种中间表示，再进一步将中间表示翻译成逻辑语言，得到查询语句或查询语句的逻辑形式，在知识库上执行所得查询语句获得正确答案集。这里的查询语句常用有 SPARQL 查询语言，表达逻辑形式的逻辑语言常用有 $\lambda - DCS$ 。这样可以使得解析更加精细，便于逻辑语言和知识库的对齐。图12.26给出了一个两步解析方法的示例。首先将问句转换为了与自然语言更加接近的中间表示。同样是针对问句“who is Justin Bieber’s brother？”，中间表示 brother_of 与句子中的单词直接可以匹配。

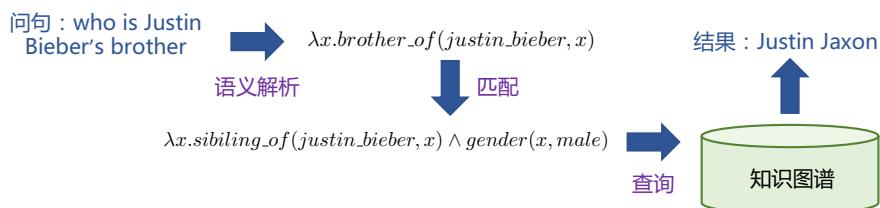


图 12.26 问句二步语义解析样例

1. 逻辑表达语言

有很多种逻辑表达语言，例如 Lamda 算子经常被用来对问句进行形式化描述。Lamda 演算子^[45]由一个单一转换规则和一个单一函数定义系统构成。包括函数定义、标识符引用和函数应用三个部分。对于问题“What states border Texax?”，定义 x 是一个变量， λx 表示获取一个参数 x 并且等待返回值的 lambda 函数，接下来通过两个函数 state 和 border 代表类型以及谓词语义，最后将函数 state 和 border 应用于参数 x 。此外，可以添加其他修饰符，如存在量词、最大最小等，可以进一步增强语义刻画能力。据此上述问句可以转换为 $\lambda x.state(x) \wedge borders(x, texas)$ 。

依存组合语义（Dependency-based Compositional Semantics, DCS）^[46]是一种特殊的操作桥接（Bridging）操作，把两个独立的语义表示片段连接起来，如图12.27所示。在语义分析过程，句子中汇总了不同片段对应的语义表示，可能无法直接进行合并，但是很多情况下过于分散的语义片段无法匹配到知识库中内容。引入桥接操作可以将两个不相邻的语义相连接，从而达到解析复杂语义的目的，尽可能连接这些离散的语义片段，在一定程度上保证语义分析能够完整。

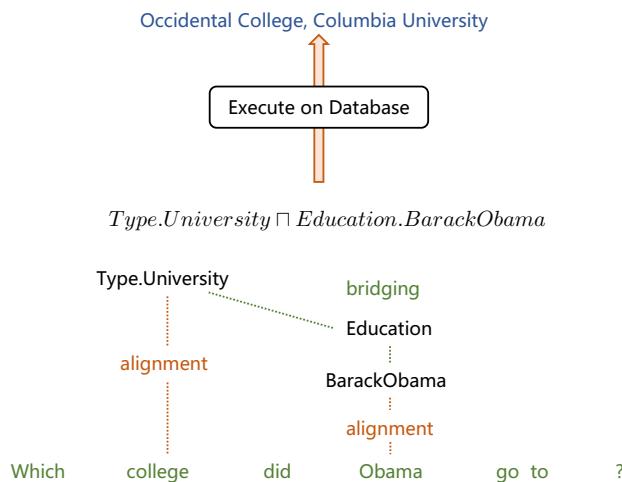


图 12.27 依存组合语义 DCS

2. 语义解析

语义解析的基本步骤可以分为短语检索、资源映射、语义组合和逻辑表达式生成这四个步骤。短语检索用于识别出问句中的实体、关系等各种短语。资源映射的目标是建立问句和知识图谱之间的映射，这包括实体链接、概念匹配和关系分类。将问句中实体、关系和知识图谱中概念相匹配后，还需要做句法分析、组合模型训练等工作。最终组合生成一个可执行的逻辑表达式，在知识图谱中获取最终答案。

我们以“复旦大学所在城市的人口有多少？”为例来介绍语义解析的具体过程，如图12.28所示。第一步将问句中关键信息提取出来，包括实体、关系谓词等。第二步与知识图谱建立各种关系，核心技术是：实体链接、概念匹配、关系抽取与分类。当给定问题后首先执行短语检测操作，主要识别句子中的变量（Variable）V，类别（Category）C，实体（Entity）E 和关系谓词（Relation）R。这个过程可以使用序列标注模型来完成。在短语检测之后，需要完成依赖关系检测，例如实体“复旦大学”与关系“所在地”之间的依赖关系。最后完成短语向知识图谱的映射，这个过程可以采用实体链接、关系分类的技术，但很多时候也要依赖词典库辅助建立链接。

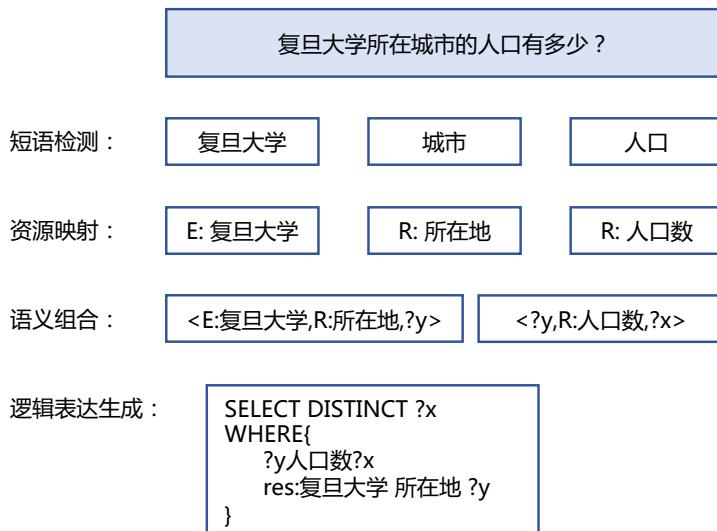


图 12.28 语义解析自然语言问句样例

在语义组合阶段，可以使用机器学习或者深度学习方法，生成有效的逻辑表达式。如果我们把自然语言及对应的逻辑语言看作是两种不同语言，语义分析任务也可以视作一种机器翻译任务。

最后，根据得到语义组合构建逻辑表达式，并在此基础上，结合知识图谱的存储结构，生成检索语句。利用检索语句从知识库中检索得到的结果，就可以用于回答问题。

12.5.2 基于信息检索的知识图谱问答

结合知识库来回答自然语言问题的另一种思路是，首先定位出自然问句中的主题实体，然后从知识库中的该实体相邻节点中选出候选答案，再对候选实体进行排序得到最后的答案。这种基于检索的知识图谱问答有两个核心问题：一是主题实体的识别，如果主题实体定位错误，之后的候选实体很大可能也是错误的。当知识图谱较复杂时，这需要通过实体链接将文本中的实体指代与其知识图谱中真正对应的实体完成映射；另一个是排序模型的效果，候选实体的得分也直接决

定最终结果，排序模型可以采用规则或者深度学习的方法实现。

文献 [47] 提出基于图谱的问答系统，利用 Freebase^[48] 知识图谱回答问题。该方法基于的假设来自于日常人们如何寻找答案的过程：一个自然语言问题可能包括一个或几个主题，那么我们就可以在知识库中找相关的主题节点，然后从主题节点距离几跳的相关节点根据节点间的关系发现答案。例如询问“What is the name of Justin Bieber brother?”, 并允许人们访问 Freebase 等知识库，人们通常会首先确定这个问题是关于 Justin Bieber，进一步根据 Justin Bieber 的 Freebase 页面所提供的关联节点，寻找他兄弟的名字。

该方法包括如下几个关键步骤：(1) 解析自然语言问句，围绕主题实体建立关于问句潜在特征的问题图；(2) 利用主题实体在 Freebase 中的邻近节点建立主题图；(3) 对齐问题图的特征和主题图的特征，找到问题与答案之间的关联；(4) 关系映射，即通过概率计算得到最可能为问题答案的关系。接下来将分别介绍上述步骤的具体方法。

1. 问题图构建

在查询答案时，通常存在多个逻辑约束。例如，对于问题“What is the name of Justin Bieber brother?”, 可以根据以下内容寻找一个人的名字：依存关系 nsubj(what, name) 表示问题是寻求一个名字的信息；依存关系 prep_of(name, brother) 表示这个名字是关于一个兄弟的；依存关系 nn(brother, Bieber) 表明 Bieber 是一个人名。通过以上的逻辑推理，可以在知识库中查询到所需要的答案。

查询答案的逻辑约束决定了基于依存特征的问题图设计。图12.29左侧给出了一个示例。左图使用虚线框中带注释的问题进行依存分析，转换后的特征图（右图）仅保留有关原始问题的相关和一般信息。许多语言信息对于答案提取提供了很多信息：疑问词 (qword) 反映了答案的类型，例如 What/who/how；问题焦点 (qfocus) 给出了预期答案类型的提示，例如姓名/金钱/时间；从问题的主要动词中提取的疑问动词 (qverb)，对答案类型也有很好的提示作用，例如动词“演奏”之后可能会跟着乐器、电影或运动队；问题主题 (qtopic) 则有助于找到相关的 Freebase 页面。

将问句的依存树转换为更通用的问题图，主要包含如下步骤：(1) 如果一个节点被标记了一个问题特征，那么用它的问题特征替换这个节点，例如，what→ qword=what；(2) 如果 qtopic 节点被标记为命名实体，则将该节点替换为其命名实体形式，例如 bieber → qtopic=person；(3) 删除任何作为限定词、介词或标点符号的叶节点。转换后的结果如图12.29右侧所示，称为问题特征图，图中每个节点和关系都是这个问题的潜在特征。

2. 主题图构建

给定一个主题，通过选择与主题节点关系在若干跳内的节点形成主题图。除了传入和传出关系之外，节点还应具有属性：描述节点属性的字符串，例如节点类型、性别或身高。需要注意的是，关系和属性之间的一个主要区别是关系的两个参数都是节点，而属性只有一个参数是节点，另一个参数是字符串。关系描述节点之间相互关联的特性，例如，伦敦可以是贾斯汀比伯的出生地，也

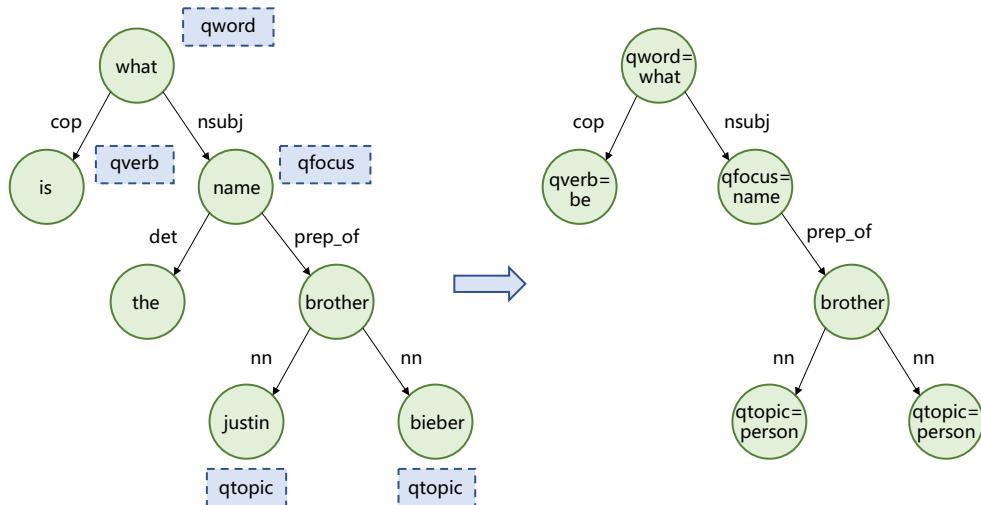


图 12.29 基于依存特征的问题特征图设计示例

可以是英国的首都。属性的参数是仅“附加”到某些节点并且没有传出边的属性。

图12.30给了一个Freebase主题图示例。以 Justin Bieber 为主题节点，通过节点间关系扩展后得到的主题图。图中实线框为节点，虚线框内为节点属性。阴影节点 Jaxon Bieber 为目标答案。

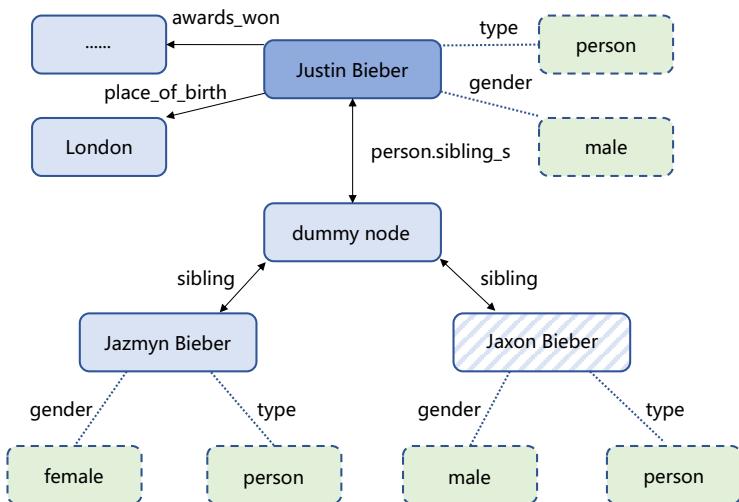


图 12.30 关于 Justin Bieber 主题的 Freebase 视图

3. 特征生成

在问题图和主题图构建完成后，需要构建人工特征用于表示问题和答案之间的关系。文献[47]所提出的方法主要构建了两种特征：问题特征和候选答案特征。**问题特征**：在问题特征图中的每一条边 $e(s,t)$ ，提取 $s, t, s|t, s|e|t$ 作为特征。其中 s 是源节点， t 是目标节点， e 表示关系。例如，对于图12.29右侧，通过边 $\text{prep_of}(\text{qfocus}=\text{name}, \text{brother})$ 可以提取以下特征： $\text{qfocus}=\text{name}$, brother , $\text{qfocus}=\text{name}| \text{brother}$, $\text{qfocus}=\text{name}| \text{prep_of} | \text{brother}$ 。**候选答案特征**：一个节点的关系和属性对区分答案非常重要，对于一个问题对应的主题图，提取每个节点的所有关系和属性作为特征。然后将问题的一个特征和候选答案的一个特征组合在一起，这样可以捕捉问题模式和答案节点之间的关系。例如，问题-答案组合特征： $\text{qfocus}=\text{money} | \text{node_type}=\text{currency}$ 。

4. 关系映射

关系映射的目标是构建知识库中的关系与自然语言单词之间的映射，发现问题所对应的关系。例如，对于“谁是乔治六世国王的父亲？”这个问题，目标是寻找 `people.person.parents` 关系。关系映射可以形式化的定义为：给定一个包含多个单词 w 的问题 Q ，希望找出使概率 $P(R|Q)$ 最大的关系 R 。为了计算的简单性，假设单词之间的条件独立并应用朴素贝叶斯方法可以得到：

$$\tilde{P}(R | Q) \propto \tilde{P}(Q | R) \tilde{P}(R) \quad (12.46)$$

$$\approx \tilde{P}(w | R) \tilde{P}(R) \quad (12.47)$$

$$\approx \prod_w \tilde{P}(w | R) \tilde{P}(R) \quad (12.48)$$

其中 $\tilde{P}(R)$ 是关系 R 的先验概率， $\tilde{P}(w | R)$ 是给定 R 的单词 w 的条件概率。

如果没有关系 R 可以使得条件概率足够大，可以采用“子关系”（sub-relation）：关系 R 是一系列子关系 $R = r = r_1.r_2.r_3$ 的串联。例如，`people.person.parents` 的子关系是 `people`、`person` 和 `parents`。同样，假设子关系之间的条件独立并应用朴素贝叶斯方法：

$$\tilde{P}_{\text{backoff}}(R | Q) \approx \tilde{P}(r | Q) \quad (12.49)$$

$$\approx \prod_r \tilde{P}(r | Q) \quad (12.50)$$

$$\propto \prod_r \tilde{P}(Q | r) \tilde{P}(r) \quad (12.51)$$

$$\approx \prod_r \prod_w \tilde{P}(w | r) \tilde{P}(r) \quad (12.52)$$

为了获得上述先验概率和条件概率，需要大量的数据进行统计。文献[47]中使用 ClueWeb09 语料库^[49] 以及基于该语料库的实体标注语料库 FACC1^[50]。基于上述两个语料集合，针对 Freebase

中的关系，使用包含关系的两个实体的句子进行关系学习 $\tilde{P}(w | R)$ 和 $\tilde{P}(w | r)$ 。整个学习过程可以分为以下几个步骤：

- (1) 将每个文档按句子拆分，并根据 Freebase 提取包含两个以上实体，并且实体间构成某种关系的句子；
- (2) 构造两个平行的语料库，一个是“关系-句子”对（用于估计 $\tilde{P}(w | r)$ 和 $\tilde{P}(R)$ ），另一个是“子关系-句子”对（对于 $\tilde{P}(w|r)$ 和 $\tilde{P}(r)$ ）。使用 ClueWeb09 和 FACC1 总计分别构建了 12 亿对平行语料；
- (3) 计算对齐关系和单词之间的共现矩阵。总共有 10484 个关系和子关系。
- (4) 根据共现矩阵，计算 $\tilde{P}(w | R), \tilde{P}(R), \tilde{P}(w | r)$ 和 $\tilde{P}(r)$ 。

12.5.3 基于深度神经网络的知识图谱问答

通过上一节的介绍，我们可以看到依靠人工构造特征和规则来进行问题理解和答案抽取，集成了大量繁琐的步骤，系统回答复杂问题能力也较难提升。近年来，随着深度学习技术的发展，基于深度学习的知识图谱问答已经成为一个重要的研究领域。

1. 基于多列卷积神经网络的知识图谱问答

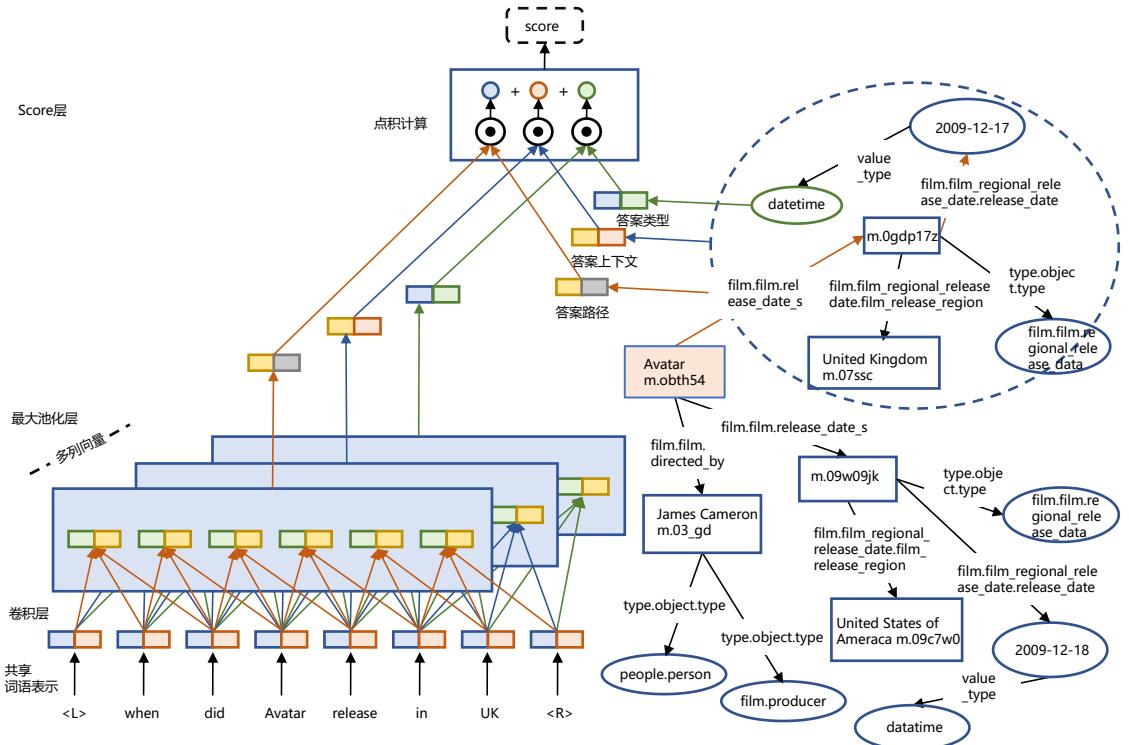
文献 [51] 中提出多列卷积神经网络（Multi-Column Convolutional Neural Networks, MCCNN）来进行知识图谱问答。具体来说，MCCNN 使用不同的列网络从输入问题中提取答案类型、关系和上下文信息，同时将知识库中的实体和关系也编码为低维向量。在此基础上，使用评分层根据问题和候选答案的表示对候选答案进行排名。MCCNN 算法框架如图12.31所示。例如，对于“What did Avatar release in UK?”问题，首先从 Freebase 知识库中查询实体 Avatar 的相关节点，将这些相关节点视为候选答案集合 C_q 。然后对于每个候选答案 a ，模型预测一个分数 $S(q, a)$ 用来确定它是否为正确答案。

MCCNN 用多列卷积神经网络来学习问题的表示。对于每个问题 q ，这些不同的列卷积学习的向量表示为 $f_1(q)$ 、 $f_2(q)$ 、 $f_3(q)$ 。Freebase 中出现的候选答案的同样也需要进行嵌入表示，对于每个候选答案 a ，利用列卷积计算其向量表示并将它们表示为 $g_1(a)$ 、 $g_2(a)$ 、 $g_3(a)$ 。这三个向量对应于问题理解中使用的三个方面。使用为问题和答案定义的这些向量表示，可以计算问答对 (q, a) 的分数。具体而言，评分函数 $S(q, a)$ 定义为：

$$S(q, a) = \underbrace{\mathbf{f}_1(q)^\top \mathbf{g}_1(a)}_{\text{候选答案路径}} + \underbrace{\mathbf{f}_2(q)^\top \mathbf{g}_2(a)}_{\text{候选答案上下文}} + \underbrace{\mathbf{f}_3(q)^\top \mathbf{g}_3(a)}_{\text{候选答案类型}} \quad (12.53)$$

其中 $f_i(q)$ 和 $g_i(a)$ 具有相同的维度。如图12.31所示，通过分层计算得分并将它们相加。下面将具体描述如何去获取候选答案、抽取答案特征和问题特征，并进行计算的过程。

候选实体生成：根据问题首先需要从 Freebase 中检索问题的候选答案。如果问题中包含一个已识别的实体，可以通过该实体可以链接到知识库。这里使用了 Freebase Search API^[48] 来查询问

图 12.31 MCCNN 模型结构图^[51]

题中的命名实体。如果问题中没有任何命名实体，则查询名词短语。然后，将链接实体的所有 2 跳节点视为候选答案。将问题 q 的候选集表示为 C_q 。

问题理解：MCCNN 使用多个卷积神经网络从共享输入词嵌入中学习问题的不同方面。如图12.31的左侧所示。对于问题 $q = w_1, \dots, w_n$, 查找层将每个单词转换成一个向量 $w_j = \mathbf{W}_v u(w_j)$, 其中 $\mathbf{W}_v \in R^{d_v \times |\mathbb{V}|}$ 是词嵌入矩阵, $u(w_j) \in \{0, 1\}^{|\mathbb{V}|}$ 是 w_j 的 one-hot 表示, \mathbb{V} 表示单词表, $|\mathbb{V}|$ 是词表中单词数量。

卷积层计算滑动窗口中单词的表示。对于 MCCNN 的第 i 列, 卷积层计算问题 q 的 n 个向量。其中第 j 个词向量为:

$$\mathbf{x}_j^{(i)} = h \left(\mathbf{W}^{(i)} [\mathbf{w}_{j-s}^\top \dots \mathbf{w}_j^\top \dots \mathbf{w}_{j+s}^\top]^\top + \mathbf{b}^{(i)} \right) \quad (12.54)$$

其中 $(2s + 1)$ 是窗口大小, $\mathbf{W}^{(i)} \in R^{d_q \times (2s+1)d_v}$ 是卷积层的权重矩阵, $\mathbf{b}(i) \in R^{d_q \times 1}$ 是偏置向量, $h(\cdot)$ 是非线性函数 (例如 softsign、tanh、sigmoid 等)。

最后, 使用一个最大池化层来获得问题的固定大小的向量表示。第 i 列中的最大池化层通过

以下方式计算问题 q 的表示:

$$f_i(q) = \max\{x_j^{(i)}\} \quad (12.55)$$

其中 $\max\{\cdot\}$ 是向量上的元素运算符。

编码候选答案: 候选答案 a 学习三种不同的向量表示 $\mathbf{g}_1(a), \mathbf{g}_2(a), \mathbf{g}_3(a)$ 的学习方法分别如下:

候选答案路径 $\mathbf{g}_1(a)$ 表示候选答案节点和被询问的实体之间的关系。如图12.31所示, 实体 Avatar 与正确答案之间的 2 跳路径为 (film.film.release-date-s, film.film-regional-release-date.release-date)。候选答案路径表示为 $\mathbf{g}_1(a) = \frac{1}{\|\mathbf{u}_p(a)\|_1} \mathbf{W}_p \mathbf{u}_p(a)$, 其中 $\|\cdot\|_1$ 是 1-范数, $\mathbf{u}_p(a) \in \mathbb{R}^{|R| \times 1}$ 表示答案路径中每个关系是否存在的 0/1 向量, $\mathbf{W}_p \in \mathbb{R}^{d_q \times |R|}$ 是参数矩阵, $|R|$ 是关系的数量。

候选答案上下文 $\mathbf{g}_2(a)$ 表示连接到候选答案路径的 1 跳实体和关系被视为候选答案上下文。它用于处理问题中的约束。如图12.31所示, 询问了《阿凡达》在英国的发布日期, 因此仅考虑答案路径上的三元组是不够的。在上下文信息的帮助下, 英国的发布日期得分高于美国。上下文表示为 $\mathbf{g}_2(a) = \frac{1}{\|\mathbf{u}_c(a)\|_1} \mathbf{W}_c \mathbf{u}_c(a)$, 其中 $\mathbf{W}_c \in \mathbb{R}^{d_q \times |C|}$ 是参数矩阵, $\mathbf{u}_c(a) \in \mathbb{R}^{|C| \times 1}$ 是表示上下文节点的存在与否, $|C|$ 是出现在答案上下文中的实体和关系的数量。

候选答案类型 $\mathbf{g}_3(a)$: Freebase 中的类型信息是对候选答案评分的重要线索。如图12.31所示, 2009-12-17 的类型是 datetime, 而 James Cameron 的类型是 people.person 和 film.producer。对于示例问题 Avatar 何时在英国发布, 应为类型为 datetime 的候选答案分配比其他答案更高的分数。向量表示定义为 $\mathbf{g}_3(a) = \frac{1}{\|\mathbf{u}_t(a)\|_1} \mathbf{W}_t \mathbf{u}_t(a)$, 其中 $\mathbf{W}_t \in \mathbb{R}^{d_q \times |T|}$ 是类型嵌入的矩阵, $\mathbf{u}_t(a) \in \mathbb{R}^{|T| \times 1}$ 是表示答案类型存在或不存在的二进制向量, $|T|$ 是类型的数量。

模型训练: 对于问题 q 的每一个正确答案 $a \in A_q$, 从候选答案 C_q 集合中随机抽取 k 个错误答案 a' , 并将它们作为否定实例来估计参数。针对 (q, a) 和 (q, a') 的损失函数为:

$$\mathcal{L}(q, a, a') = \max(0, (m - S(q, a) + S(q, a'))) \quad (12.56)$$

其中 $S(\cdot, \cdot)$ 是等式中定义的评分函数, m 是用于规范两个分数之间的差距的边际参数。

2. 基于知识图谱多跳问答算法 EmbedKGQA

简单的知识图谱问答只需要从 1 跳关系中获取答案, 而知识图谱中关系之间跳转可能存在更多的潜在候选答案。多跳知识图谱问答正是需要对知识图谱中的多条边进行推理以得出正确答案。但是知识图谱通常是不完整的, 有许多缺失的关系, 这给多跳知识图谱问答带来了额外的挑战。

文献 [52] 提出了用于知识图谱的多跳问答任务的 EmbedKGQA 方法, 使用知识图谱嵌入来完成该任务, 在稀疏知识图谱检索时特别有效。知识图谱中所有实体和关系的集合分别用 E 和 R 表示, $K \subseteq E \times R \times E$ 表示知识图谱中所有可用事实的集合。知识图谱问答中的问题涉及给定一个自然语言问题 q 和问题中存在的主题实体 $h \in E$, 任务是提取一个正确回答问题 q 的实体 $t \in E$ 。

EmbedKGQA 使用知识图谱嵌入来回答多跳自然语言问题, 并利用缺失链接预测可以用来降

低知识图谱的稀疏性，提升知识图谱在多跳问答的表现。EmbedKGQA 的结构如图12.32所示，主要包含知识图谱编码、问题编码以及答案选择三个主要部分。EmbedKGQA 算法，首先在嵌入空间中学习知识图谱的表示。然后对给定问题，模型学习问题嵌入表示。这里将嵌入空间称为 C_d 。最后，模型将结合知识图谱嵌入表示和问题嵌入表示来预测答案。

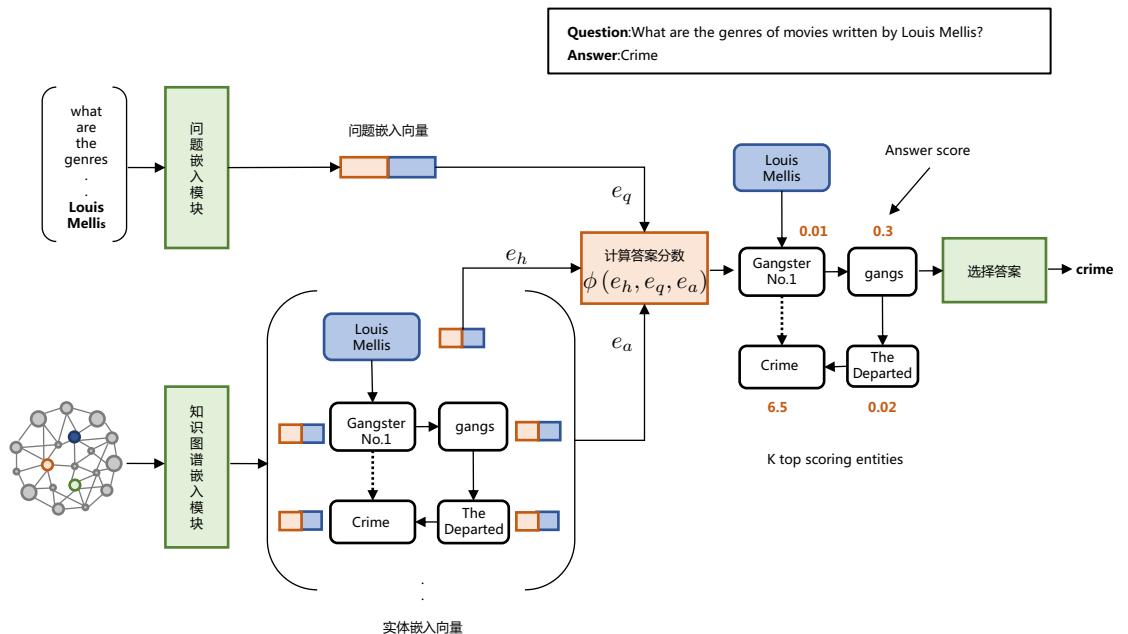


图 12.32 EmbedKGQA 算法神经网络结构^[52]

知识图谱编码模型：为知识图谱中的所有实体 $h, t \in E$ 和所有关系 $r \in R$ 训练嵌入向量表示 e_h, e_r, e_t ，使得 $e_h, e_r, e_t \in C_d$ 。这里的知识图谱嵌入向量表示用于计算问题和答案实体之间的评分函数。

问题编码模型：将自然语言问题 q 编码到同一维度的向量空间中 $e_q \in C_d$ 中。这是使用前馈神经网络完成的，该网络首先使用 RoBERTa^[53] 将问题 q 编码到 768 维向量中。然后通过具有 ReLU 激活的 4 个完全连接的线性层，最后投影到高维空间 C_d 上。

给定一个问题 q ，主题实体 $h \in E$ 和一组候选答案实体 $A \subseteq E$ ，使用如下方式学习问题嵌入表示：

$$\begin{aligned} \phi(e_h, e_q, e_a) &> 0 \quad \forall a \in A \\ \phi(e_h, e_q, e_a) &< 0 \quad \forall a \notin A \end{aligned} \tag{12.57}$$

对于每个问题，使用所有候选答案实体 $a' \in E$ 计算分数 $\phi(\cdot)$ 。通过最小化 Sigmoid 得分和目

标标签之间的二元交叉熵损失来学习模型。

$$\mathcal{L} = BCE(\text{sigmoid}(\phi(e_h, e_q, e_a)), \text{Label}) \quad (12.58)$$

答案选择模型：最终在推理时，模型根据所有可能的答案 $a' \in E$ 对 (head, question) 对进行评分。对于相对较小的知识图谱，如 MetaQA 等，只需选择得分最高的实体：

$$e_{ans} = \arg \max_{a' \in \epsilon} \phi(e_h, e_q, e_{a'}) \quad (12.59)$$

为了从大规模识图谱中选择候选答案，EmbedKGQA 还引入一个评分函数 $S(r, q)$ ，用于对给定问题 q 的每个关系 $r \in R$ 进行排序。 \mathbf{h}_r 是关系 r 的嵌入向量， $q = (< s >, w_1, \dots, w_{|q|}, < /s >)$ 是问题 q 中输入到 RoBERTa 的单词序列。评分函数定义为 RoBERTa(\mathbf{h}_q) 的最后一个隐藏层的最终输出与关系 $r(\mathbf{h}_r)$ 的点积，之后使用 Sigmoid 函数归一化。

$$\mathbf{h}_q = \text{RoBERTa}(q) \quad (12.60)$$

$$S(r, q) = \text{sigmoid}(\mathbf{h}_q^T \mathbf{h}_R) \quad (12.61)$$

在所有关系中，选择那些得分大于 0.5 的关系，记为集合 R_a 。对于目前获得的每个候选实体 a' ，找到头部实体 h 和 a' 之间最短路径中的关系，并将这组关系记为 $R_{a'}$ 。每个候选答案实体的关系分数定义为它们的交集的大小：

$$RelScore_{a'} = |R_a \cap R_{a'}| \quad (12.62)$$

最后，使用关系分数和答案得分的线性组合来找到最终的答案实体。

$$e_{ans} = \arg \max_{a' \in N_h} \phi(e_h, e_q, e_{a'} + \gamma * RelScore_{a'}) \quad (12.63)$$

其中 γ 是一个可调节超参数。

12.5.4 知识图谱问答语料库

知识图谱问答的评价指标和分类问题类似，通常用准确率 (Accuracy)、精确率 (Precision)、召回率 (Recall) 和 F1 值等，这些指标在前文多有介绍，这里不再展开描述。目前常用的知识图谱问答语料库如表12.9所示。

1. QALD 知识图谱问答语料库

QALD^[54, 55] 自 2011 年提出，到 2022 为止最新版本为 QALD-9 Plus。该语料库针对 DBpedia 知识库，构建了多语言问答、基于链接数据的跨数据集问答、融合文本数据的混合问答等任务。包

表 12.9 知识图谱问答数据集

数据集	知识图谱	语言	数据集规模
QALD	DBpedia	多语言	558
WebQuestions	Freebase	英语	5810
SimpleQuestions	Freebase	英语	10.8 万
MetaQA	Wikipedia	英语	40 万

含 558 个问题，并针对每个问题提供可以在 DBpedia 上检索得到正确答案的 SPARQL 语句、答案在 DBpedia 上的 URI 标识以及答案的类型。

2. WebQuestions 知识图谱问答语料库

WebQuestions^[56] 是 2013 年由斯坦福大学研究人员构建的通用领域的知识图谱问答评测集合，包含 5810 个问题答案对，基于 Freebase 知识库构建。数据集构建过程中通过使用 Google Suggest API 获取了超过一千万问题，并从中选取了 10 万条，采用众包的方式人工利用 Freebase 知识库进行回答。从所有回答中选取了具有多个一致答案的问题和答案组成了 WebQuestions 语料库。

3. SimpleQuestions 知识图谱问答语料库

SimpleQuestions^[57] 是 2015 年由 Facebook AI 研究人员构建的针对简单问题的大规模知识图谱问答语料。该语料库采用人工标注的方法，以 Freebase 作为答案来源，根据知识库中的事实人工构造问句，总计包含 108442 个问题答案对。

4. MetaQA 知识图谱问答语料库

MetaQA^[58] 是 MoviE Text Audio QA 的缩写，主要包含三个部分：普通文本数据（Vanilla text data）、神经网络翻译数据（NTM text data）以及音频数据（Audio data）。通常知识图谱问答中使用普通文本数据，其中包含 40 万个问题，其中单跳（1-hop）问答来源于 Facebook MovieQA（也成为 WikiMovies）数据集中的“wiki_entities”分支，同时移除了问题中的歧义实体，并数十个增加了多跳问答类型。

12.6 延伸阅读

人类的知识是真正理解语言的基石，随着知识表示学习、知识获取技术和各种知识感知应用的出现，表示实体之间结构关系的知识图谱已成为认知智能的主流研究方向。本章对以知识图谱表示与存储、知识图谱获取与构建、知识图谱推理以及知识图谱问答等四个方面的基础概念和经典方法展开了介绍。虽然学术界和工业界在知识图谱的上述方面都开展大量系统的研究，但是在分布式的知识表示学习方法、知识图谱动态扩展以及知识图谱推理等方面仍有很多开放问题等待解决。

分布式的知识表示学习方法能够高效地建模知识图谱的拓扑特征，且具有一定的扩展性。过

去十年中开发了许多成功的知识表示学习方法，除了本章中已经介绍的 TransE^[14]、TransR^[16] 和 RotatE^[41]，还包括 TransH^[59]、TransD^[60]、ComplEx^[61] 和 PairRE^[62] 等等方法。当前知识表示学习的一个挑战是需要有高质量的训练数据。训练数据通常需要人工标注，这是一项耗时且费力的工作，同时，由于知识的复杂性，人类标注的数据也可能存在不一致性或错误。这些都会影响知识表示学习方法的准确性和可靠性。另一个挑战是它们难以解释，无法对低度实体建模，存在词表外实体无法表示的问题。这使得在当前知识表示学习方法在标注数据稀疏场景下的应用大大受限。

大多数知识图谱从静态数据中捕获信息，但实体和关系可能随时间而变化，因此想要自主地维护时态知识图谱的完整性和正确性，需要对时态元素进行适当的建模。文献 [63] 首先提出使用递归神经网络构建时间知识图谱，文献 [64] 通过为静态模型配备一个历时实体嵌入函数来学习任何时间点的实体特征。时间感知知识图嵌入方法 TeLM^[65] 则使用线性时间正则化器和多向量嵌入来执行时间知识图的四阶张量分解。解决这个问题的另一个思路是，利用事件信息辅助建模时态要素。事件是人类社会的核心特征之一，人们的社会活动往往是事件驱动的。知识图谱研究聚焦于实体和实体之间的关系，缺乏对事理逻辑知识的挖掘。针对上述问题，哈尔滨工业大学刘挺教授团队提出了事理图谱 (Eventic Graph, EG) 的概念^[66]，旨在将文本中对事件及其关系的描述抽取并抽象出来，构建描述事件之间演化规律和模式的事理逻辑知识库。

基于神经网络的方法提供了高效的推理能力，但是知识图谱中的信息通常是不完整的，也就是说，它只涵盖了部分专业领域的信息，推理系统在进行推理时可能会遇到缺乏信息的情况，这会影响推理的准确性。因此零样本推理^{[67][68][69]} 受到研究界的广泛关注，即在知识图谱中进行推理时，没有相关的样本数据可供使用。这意味着算法必须依靠图谱本身的结构和已知的实体和关系来推断新的信息。同样，可解释性人工智能的研究可以用来设计新的可解释的链接预测模型^{[70][71][72]}，即利用知识图谱中的实体和关系来推断出新的信息的过程中，会根据已知的信息推断出新的结论，并能够对这些推断过程进行解释，以便人们更好地理解推理的基础和原因。

12.7 习题

- (1) 基于属性图的知识图谱表示有什么缺点？
- (2) TransR 算法相比 TransE 算法做了什么改进，可以解决什么问题？
- (3) 使用关系型数据库作为知识图谱的存储介质，想要兼顾存储空间的利用率和查询效率，应当使用哪种存储方案？
- (4) 实体对齐技术主要解决哪些问题？存在哪些技术难点？
- (5) 基于表示学习的知识图谱推理相比于基于符号的知识图谱推理，有何优势？
- (6) 深度学习技术可以应用在知识图谱问答的基于语义解析和基于信息检索的范式的哪些阶段？

参考文献

- [1] Feigenbaum E A. The art of artificial intelligence: Themes and case studies of knowledge engineering [C]//Proceedings of the Fifth International Joint Conference on Artificial Intelligence: volume 2. Boston, 1977.
- [2] Powers D M. Robot intelligence[J]. Electronics Today International (Australia), 1983:15-18.
- [3] Sowa J F. Semantic networks[J]. 1987.
- [4] Gruber T R. Toward principles for the design of ontologies used for knowledge sharing?[J]. International journal of human-computer studies, 1995, 43(5-6):907-928.
- [5] Berners-Lee T, Hendler J, Lassila O. The semantic web[J]. Scientific american, 2001, 284(5):34-43.
- [6] Sullivan D. A reintroduction to our knowledge graph and knowledge panels[J]. The Keyword, Google, May, 2020, 20.
- [7] Zhang Z, Han X, Liu Z, et al. Ernie: Enhanced language representation with informative entities[C]// Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019: 1441-1451.
- [8] Sun Y, Wang S, Li Y, et al. Ernie: Enhanced representation through knowledge integration[J]. arXiv preprint arXiv:1904.09223, 2019.
- [9] Yamada I, Asai A, Shindo H, et al. Luke: Deep contextualized entity representations with entity-aware self-attention[C]//Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2020: 6442-6454.
- [10] Angles R. The property graph database model.[C]//AMW. 2018.
- [11] Miller J J. Graph database applications and concepts with neo4j[C]//Proceedings of the southern association for information systems conference, Atlanta, GA, USA: volume 2324. 2013.
- [12] Consortium W W W, et al. Rdf 1.1 concepts and abstract syntax[J]. 2014.

- [13] Hitzler P, Krötzsch M, Parsia B, et al. Owl 2 web ontology language primer[J]. W3C recommendation, 2009, 27(1):123.
- [14] Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality[C]//Advances in neural information processing systems. 2013: 3111-3119.
- [15] Bordes A, Usunier N, Garcia-Duran A, et al. Translating embeddings for modeling multi-relational data[J]. Advances in neural information processing systems, 2013, 26.
- [16] Lin Y, Liu Z, Sun M, et al. Learning entity and relation embeddings for knowledge graph completion [C]//Twenty-ninth AAAI conference on artificial intelligence. 2015.
- [17] Wylot M, Hauswirth M, Cudré-Mauroux P, et al. Rdf data storage and query processing schemes: A survey[J]. ACM Computing Surveys (CSUR), 2018, 51(4):1-36.
- [18] Francis N, Green A, Guagliardo P, et al. Cypher: An evolving query language for property graphs [C]//Proceedings of the 2018 International Conference on Management of Data. 2018: 1433-1445.
- [19] Sirin E, Parsia B. Sparql-dl: Sparql query for owl-dl.[C]//OWLED: volume 258. Citeseer, 2007.
- [20] Rodriguez M A. The gremlin graph traversal machine and language (invited talk)[C]//Proceedings of the 15th Symposium on Database Programming Languages. 2015: 1-10.
- [21] Xu H, Wang W, Mao X, et al. Scaling up open tagging from tens to thousands: Comprehension empowered attribute value extraction from product title[C]//Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019: 5214-5223.
- [22] Hoffart J, Suchanek F M, Berberich K, et al. Yago2: exploring and querying world knowledge in time, space, context, and many languages[C]//Proceedings of the 20th international conference companion on World wide web. 2011: 229-232.
- [23] Lehmann J, Isele R, Jakob M, et al. Dbpedia—a large-scale, multilingual knowledge base extracted from wikipedia[J]. Semantic web, 2015, 6(2):167-195.
- [24] Bollacker K, Tufts P, Pierce T, et al. A platform for scalable, collaborative, structured information integration[C]//Intl. Workshop on Information Integration on the Web (IIWeb' 07). 2007: 22-27.
- [25] Wu W, Li H, Wang H, et al. Probase: A probabilistic taxonomy for text understanding[C]// Proceedings of the 2012 ACM SIGMOD international conference on management of data. 2012: 481-492.

- [26] Kolitsas N, Ganea O E, Hofmann T. End-to-end neural entity linking[J]. CoNLL 2018, 2018:519.
- [27] Ganea O E, Hofmann T. Deep joint entity disambiguation with local neural attention[C]//Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. 2017: 2619-2629.
- [28] Chen M, Tian Y, Yang M, et al. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment[C]//Proceedings of the 26th International Joint Conference on Artificial Intelligence. 2017: 1511-1517.
- [29] Wang Z, Lv Q, Lan X, et al. Cross-lingual knowledge graph alignment via graph convolutional networks[C]//Proceedings of the 2018 conference on empirical methods in natural language processing. 2018: 349-357.
- [30] Bruna J, Zaremba W, Szlam A, et al. Spectral networks and locally connected networks on graphs [J]. arXiv preprint arXiv:1312.6203, 2013.
- [31] Tsarkov D, Horrocks I. Fact++ description logic reasoner: System description[C]//International joint conference on automated reasoning. Springer, 2006: 292-297.
- [32] Haarslev V, Möller R. Racer system description[C]//International Joint Conference on Automated Reasoning. Springer, 2001: 701-705.
- [33] Sirin E, Parsia B, Grau B C, et al. Pellet: A practical owl-dl reasoner[J]. Journal of Web Semantics, 2007, 5(2):51-53.
- [34] Glimm B, Horrocks I, Motik B, et al. Hermit: an owl 2 reasoner[J]. Journal of Automated Reasoning, 2014, 53(3):245-269.
- [35] Eiter T, Gottlob G, Mannila H. Disjunctive datalog[J]. ACM Transactions on Database Systems (TODS), 1997, 22(3):364-418.
- [36] Gebser M, Kaminski R, Kaufmann B, et al. Clingo= asp+ control: Preliminary report[J]. arXiv preprint arXiv:1405.3694, 2014.
- [37] Nenov Y, Piro R, Motik B, et al. Rdfox: A highly-scalable rdf store[C]//International Semantic Web Conference. Springer, 2015: 3-20.
- [38] Forgy C L. Rete: A fast algorithm for the many pattern/many object pattern match problem[M]// Readings in Artificial Intelligence and Databases. Elsevier, 1989: 547-559.

- [39] Socher R, Chen D, Manning C D, et al. Reasoning with neural tensor networks for knowledge base completion[J]. Advances in neural information processing systems, 2013, 26.
- [40] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space [C]//ICLR. 2013.
- [41] Sun Z, Deng Z H, Nie J Y, et al. Rotate: Knowledge graph embedding by relational rotation in complex space[J]. arXiv preprint arXiv:1902.10197, 2019.
- [42] Purington A, Taft J G, Sannon S, et al. " alexa is my new bff" social roles, user satisfaction, and personification of the amazon echo[C]//Proceedings of the 2017 CHI conference extended abstracts on human factors in computing systems. 2017: 2853-2859.
- [43] Kaplan A, Haenlein M. Siri, siri, in my hand: Who' s the fairest in the land? on the interpretations, illustrations, and implications of artificial intelligence[J]. Business Horizons, 2019, 62(1):15-25.
- [44] Hoy M B. Alexa, siri, cortana, and more: an introduction to voice assistants[J]. Medical reference services quarterly, 2018, 37(1):81-88.
- [45] Wong Y W, Mooney R. Learning synchronous grammars for semantic parsing with lambda calculus [C]//Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics. 2007: 960-967.
- [46] Liang P. Lambda dependency-based compositional semantics[J]. arXiv preprint arXiv:1309.4408, 2013.
- [47] Yao X, Durme B V. Information extraction over structured data: Question answering with freebase [J]. meeting of the association for computational linguistics, 2014.
- [48] Bollacker K, Evans C, Paritosh P, et al. Freebase: a collaboratively created graph database for structuring human knowledge[C]//Proceedings of the 2008 ACM SIGMOD international conference on Management of data. 2008: 1247-1250.
- [49] Callan J, Hoy M, Yoo C, et al. Clueweb09 data set[Z]. 2009.
- [50] Gabrilovich E, Ringgaard M, Subramanya A. Facc1: Freebase annotation of clueweb corpora[M]. Version, 2013.
- [51] Dong L, Wei F, Zhou M, et al. Question answering over freebase with multi-column convolutional neural networks[C]//Proceedings of the 53rd Annual Meeting of the Association for Computational

- Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers): volume 1. 2015: 260-269.
- [52] Saxena A, Tripathi A, Talukdar P P. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings[J]. meeting of the association for computational linguistics, 2020.
- [53] Liu Y, Ott M, Goyal N, et al. Roberta: A robustly optimized bert pretraining approach[J]. arXiv preprint arXiv:1907.11692, 2019.
- [54] Unger C, Forascu C, Lopez V, et al. Question answering over linked data (qald-4)[C]//Working Notes for CLEF 2014 Conference. 2014.
- [55] Usbeck R, Gusmita R H, Ngomo A N, et al. 9th challenge on question answering over linked data (QALD-9)[C/OL]//Joint proceedings of the 4th Workshop on Semantic Deep Learning (SemDeep-4) and NLIWoD4: Natural Language Interfaces for the Web of Data (NLIWOD-4) and 9th Question Answering over Linked Data challenge (QALD-9) co-located with 17th International Semantic Web Conference (ISWC 2018), Monterey, California, United States of America, October 8th - 9th, 2018. 2018: 58-64. <https://svn.aksw.org/papers/2018/QALD9/public.pdf>.
- [56] Berant J, Chou A, Frostig R, et al. Semantic parsing on freebase from question-answer pairs[C]// Proceedings of the 2013 conference on empirical methods in natural language processing. 2013: 1533-1544.
- [57] Bordes A, Usunier N, Chopra S, et al. Large-scale simple question answering with memory networks [J]. arXiv preprint arXiv:1506.02075, 2015.
- [58] Zhang Y, Dai H, Kozareva Z, et al. Variational reasoning for question answering with knowledge graph[C]//Thirty-second AAAI conference on artificial intelligence. 2018.
- [59] Wang Z, Zhang J, Feng J, et al. Knowledge graph embedding by translating on hyperplanes[C]// Proceedings of the AAAI conference on artificial intelligence: volume 28. 2014.
- [60] Ji G, He S, Xu L, et al. Knowledge graph embedding via dynamic mapping matrix[C]//Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers). 2015: 687-696.
- [61] Trouillon T, Welbl J, Riedel S, et al. Complex embeddings for simple link prediction[C]//International conference on machine learning. PMLR, 2016: 2071-2080.

58 自然语言处理导论 -- 张奇、桂韬、黄萱菁

- [62] Chao L, He J, Wang T, et al. Pairre: Knowledge graph embeddings via paired relation vectors[J]. arXiv preprint arXiv:2011.03798, 2020.
- [63] Garcia-Durán A, Dumanović S, Niepert M. Learning sequence encoders for temporal knowledge graph completion[J]. arXiv preprint arXiv:1809.03202, 2018.
- [64] Goel R, Kazemi S M, Brubaker M, et al. Diachronic embedding for temporal knowledge graph completion[C]//Proceedings of the AAAI Conference on Artificial Intelligence: volume 34. 2020: 3988-3995.
- [65] Xu C, Chen Y Y, Nayyeri M, et al. Temporal knowledge graph completion using a linear temporal regularizer and multivector embeddings[C]//Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2021: 2569-2578.
- [66] Li Z, Ding X, Liu T. Constructing narrative event evolutionary graph for script event prediction[C]// Proceedings of the 27th International Joint Conference on Artificial Intelligence. 2018: 4201-4207.
- [67] Wang X, Ye Y, Gupta A. Zero-shot recognition via semantic embeddings and knowledge graphs[C]// Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 6857-6866.
- [68] Kampffmeyer M, Chen Y, Liang X, et al. Rethinking knowledge graph propagation for zero-shot learning[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 11487-11496.
- [69] Liu L, Zhou T, Long G, et al. Attribute propagation network for graph zero-shot learning[C]// Proceedings of the AAAI Conference on Artificial Intelligence: volume 34. 2020: 4868-4875.
- [70] Guo S, Wang Q, Wang L, et al. Knowledge graph embedding with iterative guidance from soft rules [C]//Proceedings of the AAAI Conference on Artificial Intelligence: volume 32. 2018.
- [71] Wang P, Dou D, Wu F, et al. Logic rules powered knowledge graph embedding[J]. arXiv preprint arXiv:1903.03772, 2019.
- [72] Cheng K, Yang Z, Zhang M, et al. Uniker: A unified framework for combining embedding and definite horn rule reasoning for knowledge graph inference[C]//Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. 2021: 9753-9771.

索引

- Distance Model, 9
Entity Alignment, 26
Entity Linking, 22
Knowledge Base Question Answering, KBQA, 39
Knowledge Graph, KG, 1
Knowledge reasoning, 31
Knowledge Representation Learning, 9
Ontology, 3
Property Graph, 6
RDFS, 8
Resource Description Framework, RDF, 7
Semantic Network, 3
Semantic Web, 3
Translational Model, 9
Web Ontology Language, OWL, 8
实体对齐, 26
实体链接, 22
属性图, 6
平移模型, 9
归纳推理, 31
本体, 3
溯因推理, 31
演绎推理, 31
知识图谱, 1
知识图谱问答, 39
知识推理, 31
知识表示学习, 9
网络本体语言, 8
语义网, 3
语义网络, 3
资源描述框架, 7
资源描述框架模式, 8
距离模型, 9