诺贝尔文学奖提名知识图谱构建课程报告

作者信息

姓名	学号
宋文松	12421182
张鑫	22421264
张璇	22421035
钟好	12421099

1. 引言

在当今数字化信息爆炸的时代，知识图谱作为一种强大的知识组织和管理工具，能够以直观、高效的方式揭示数据之间的内在联系。本报告详细阐述了如何借助开放数据源 Wikidata 获取诺贝尔文学奖提名者的数据，并运用 Neo4j 构建和可视化知识图谱的全过程。通过这一项目实践，我们期望深入探索文学奖提名在历史进程中的发展趋势和潜在模式，从而加深对文学奖评选特点的理解，挖掘文学领域发展的内在规律。这不仅有助于我们从新的视角审视文学领域的发展脉络，还能为相关研究和分析提供有价值的参考依据。

2. 数据来源和预处理

数据来源

本项目的数据来源于 Wikidata，这是一个免费且开放的全球知识库，涵盖了极其广泛的知识数据，为各类研究和项目提供了丰富的数据资源。Wikidata 以其多语言支持、数据准确性和更新及时性等特点，成为众多领域研究人员获取数据的重要渠道。

数据检索

为了获取诺贝尔文学奖提名者的相关数据，我们运用了 SPARQL 查询语言从 Wikidata 中进行检索。此次检索重点关注提名者信息、提名作品以及提名年份（限定在 1965 年及以后），具体的 SPARQL 查询语句如下：

```
SELECT DISTINCT?nominee?nomineeLabel?citizenshipLabel?workLabel?
nominationYear
WHERE {
 BIND( wd:Q37922 AS?prize )
?nominee p:P1411 [ ps:P1411?prize; pq:P585?time ]
 BIND( year(?time) AS?nominationYear )
 OPTIONAL {
?nominee wdt:P27 [ rdfs:label?citizenshipLabel ]
 FILTER(LANG(?citizenshipLabel) = "en")
}

 OPTIONAL {
?nominee wdt:P800?work.
?work rdfs:label?workLabel.
 FILTER(LANG(?workLabel) = "en")
}
 FILTER NOT EXISTS {
?nominee wdt:P166?prize.
}
 FILTER(?nominationYear > 1965)
 SERVICE wikibase:label { bd:serviceParam wikibase:language "en". }
}
ORDER BY DESC(?nominationYear)?nomineeLabel
```

在这段查询语句中，首先明确了要查询的奖项（诺贝尔文学奖）的标识符，然后通过关联属性获取提名者信息、提名时间，并通过可选查询获取提名者的国籍和提名作品信息。同时，过滤掉已经获得该奖项的提名者，确保只保留未获奖的提名数据。最后，按照提名年份降序和提名者标签进行排序，以便于后续的数据处理和分析。

数据清洗

从 Wikidata 下载的数据可能存在不完整、不准确或重复的情况，因此需要进行严格的数据清洗工作。我们采用了一系列的数据清洗策略，以确保所有记录的完整性和准确性：

缺失值处理：仔细检查每一条记录，对于存在关键信息缺失（如提名者姓名、提名年份等）的记录，进行标记并分析缺失原因。对于部分可通过其他数据源补充的缺失值，尝试进行补充；对于无法补充且严重影响分析的记录，予以删除。

重复值处理：通过编写程序或使用数据分析工具，对数据集中的重复记录进行查找和删除。重复记录可能会干扰后续的分析结果，因此确保数据的唯一性至关重要。

数据格式规范化：对数据中的日期、文本等字段进行格式规范化处理，使其符合统一的标准格式，便于后续的数据处理和分析。

经过数据清洗后，我们得到了一份高质量、完整且准确的诺贝尔文学奖提名者数据集，为后续的知识图谱构建奠定了坚实的基础。

3. 知识图谱构建

数据模型设计

在构建知识图谱之前，我们需要设计合理的数据模型，明确知识图谱中的节点和关系类型。本项目设计的数据模型包含以下几类节点和关系：

节点：

Author：代表提名作者，包含作者的姓名等关键信息。

Work：代表提名作品，包含作品的标题等信息。

Country：代表作者的国籍，包含国家的名称。

Year：代表提名年份，包含具体的年份数值。

关系：

NOMINATED_FOR：表示作者与提名作品之间的关系，即作者被提名作品。

ORIGINATED_FROM：表示作者与国籍之间的关系，即作者来自某个国家。

NOMINATED_IN_YEAR：表示作品与提名年份之间的关系，即作品在某年被提名。

Cypher 导入脚本

为了将清洗后的数据导入到 Neo4j 中构建知识图谱，我们编写了以下 Cypher 导入脚本：

```
LOAD CSV WITH HEADERS FROM 'file:///nominees.csv' AS row
MERGE (author:Author {name: row.nomineeLabel})
MERGE (country:Country {name: row.citizenshipLabel})
MERGE (work:Work {title: row.workLabel})
MERGE (year:Year {value: row.nominationYear})
CREATE (author)-[:NOMINATED_FOR]->(work)
```

```
CREATE (author)-[:ORIGINATED_FROM]->(country)
CREATE (work)-[:NOMINATED_IN]->(year)
```

这段脚本首先从本地文件系统中加载包含提名者数据的 CSV 文件，然后使用 MERGE 语句创建或匹配节点，如果节点已经存在则直接使用，不存在则创建新节点。最后，通过 CREATE 语句创建节点之间的关系，从而构建出完整的知识图谱。

4. 可视化和分析

可视化展示

在 Neo4j 浏览器中，我们可以直观地可视化构建好的知识图谱。通过简单的 Cypher 查询语句，即可将知识图谱以图形化的方式展示出来，便于我们观察和分析数据之间的关系。本项目使用以下代码生成可视化知识图谱：

```
MATCH (n)-[r]->(m)
RETURN n, r, m
```

这条命令匹配所有的节点 (n) 和 (m) 以及它们之间的关系 [r]，然后将这些节点和关系返回并在 Neo4j 浏览器中以图形化的形式展示出来，使我们能够清晰地看到作者、作品、国籍和提名年份之间的关联。

数据分析

除了可视化展示，我们还执行了各种 Cypher 查询来深入分析数据，挖掘其中隐藏的信息和规律：

最频繁提名的作者：通过编写以下 Cypher 查询语句，找出被提名次数最多的作者：

```
MATCH (author:Author)-[:NOMINATED_FOR]->()
RETURN author.name, COUNT(*) AS nomination_count
ORDER BY nomination_count DESC
LIMIT 10
```

通过分析这些最频繁提名的作者，我们可以了解到在文学领域中哪些作者受到了长期的关注和认可，以及他们的创作风格和影响力。

最频繁提名的作品：使用类似的查询语句，找出被提名次数最多的作品：

```

```

```
MATCH (work:Work)<-[:NOMINATED_FOR]-()
RETURN work.title, COUNT(*) AS nomination_count
ORDER BY nomination_count DESC
LIMIT 10
```

分析这些作品的特点和主题，可以帮助我们了解诺贝尔文学奖的评选倾向和审美标准。

5. 导出知识图谱

图形文件导出

为了便于在报告和展示中使用知识图谱，我们可以利用 Neo4j 浏览器的导出功能将图形视图导出为 PNG 或 SVG 文件。在 Neo4j 浏览器中，只需点击相应的导出按钮，选择所需的文件格式，即可将可视化的知识图谱保存为图片文件。

数据文件导出

除了图形文件，我们还可以导出知识图谱的数据文件，以便进行进一步的分析和处理。本项目提供了两种数据文件导出方式：

使用 apoc.export.csv.query 导出特定查询结果：通过该工具，可以将特定的 Cypher 查询结果导出为 CSV 文件，方便在其他数据分析工具中进行分析。例如，如果我们想导出所有提名者的信息，可以使用以下命令：

```
CALL apoc.export.csv.query('MATCH (author:Author) RETURN author.name, author.
citizenship', 'file:///authors.csv')
```

使用 neo4j-admin dump 导出整个数据库：如果需要导出整个知识图谱数据库，可以使用 `neo4j-admin dump` 命令。该命令将数据库以二进制格式进行备份，方便在需要时进行恢复和迁移。

6. 结论

通过本项目的实践，我们成功地利用 Wikidata 数据源和 Neo4j 工具构建并可视化了诺贝尔文学奖提名知识图谱。这一过程不仅使我们能够有效地可视化和分析诺贝尔文学奖的提名数据，揭示了文学领域随时间变化的趋势，还让我们深入了解了知识图谱构建的各个环节和关键技术。通过 Neo4j 框架的使用，我们掌握了如何快速构建一个知识图谱，从数据获取、清洗、模型设计到图谱构建、可视化和分析，每一个步骤都积累了宝贵的经验。这些经验将为我们今后在知识图谱领域的进一步研究和应用奠定坚实的基础。同时，我们也认识到在项目实施过程中存在的一些不足之处，例如数据的完整性和准

确性仍有待进一步提高，数据分析的深度和广度还可以拓展。在未来的研究中，我们将继续优化数据处理流程，探索更多的分析方法，以挖掘出更多有价值的信息。

7. 文件架构说明

本项目的文件架构如下：

nobel.csv：包含诺贝尔文学奖提名者数据的数据集，是整个项目的数据基础。

records.json：以 JSON 格式存储的部分知识图谱数据，可用于特定的数据交换和处理场景。

export.csv：通过特定查询导出的 CSV 文件，包含了经过筛选和处理的知识图谱数据。

graph.png：可视化的知识图谱图片文件，直观展示了知识图谱的结构和关系。