通过自动化集成数据生成可靠的不良事件概况以促进健康（GRAPH-AID）：一种半自动本体构建方法

Srikar Reddy Gadusu[1,*], Larry Callahan[2], Samir Lababidi[2], Arunasri Nishtala[2], Sophia Healy[1] and Hande Küçük McGinty[1,*]

[1]Kansas State University, Manhattan KS 66502, USA
[2]U.S. Food and Drug Administration

Abstract

随着数据和知识的迅速扩展，采用系统方法生成本体变得至关重要。由于每天的数据量增加和内容频繁变化，对存储和检索用于创建知识图谱的信息的数据库的需求变得更加紧迫。先前建立的知识获取与表示方法（KNARM）概述了一种系统的解决这些挑战并创建知识图谱的方法。然而，遵循这种方法突显了无缝集成 Neo4j 数据库与 Web 本体语言（OWL）的现有挑战。之前尝试将数据从 Neo4j 整合到本体中已经被讨论过，但这些方法通常需要理解描述逻辑（DL）语法，这可能对许多用户来说并不熟悉。因此，需要一种更易于使用的方法来填补这一空白。本文提出了一种用户友好的方法，利用 Python 及其 rdflib 库支持本体开发。我们通过整合来自美国食品药品监督管理局不良事件报告系统（FAERS）数据库的数据创建的 Neo4j 数据库展示了我们的新方法。使用此数据集，我们开发了一个 Python 脚本，该脚本能够自动生成所需类及其公理，从而促进更顺畅的集成过程。这种方法为快速增长的药物不良反应数据集生成本体提供了实际解决方案，支持改进药物安全性监测和公共卫生决策。

Keywords

知识图谱, KNARM, 自动化本体生成方法, 知识工程方法论, Neo4j, 疫苗不良事件, 药物不良事件

1. 介绍

在数字时代，数据的空前增长既给知识管理带来了巨大的挑战，也为发现提供了深刻的机会。随着各种领域中的数据迅速积累，组织、搜索和有效管理这些数据的需求比以往任何时候都更加紧迫。本体论是语义网的基础，在结构化数据方面至关重要，从而增强检索能力并促进知识发现。然而，数据的增长和变化速度要求有创新的解决方案来创建本体论以跟上这一节奏，而传统上这项任务一直受到复杂技术障碍 [6], [7], [8] 的阻碍。

传统本体开发通常需要深入理解形式逻辑，如 Web 本体语言（OWL），这对于没有描述逻辑（DL）语法专门培训的人来说可能是一个重大障碍 [4], [5]. 尽管知识获取与表示方法论（KNARM）[1] 提供了一种系统的方法来构建知识图谱，但其与现代 NoSQL 数据库如 Neo4j 的集成显示出显著差距，尤其是在将这些数据库与基于 OWL 的语义框架对齐方面。Neo4j 在管理复杂数据关系方面的功能强大，但由于这些集成挑战，在语义网项目中被利用不足。

这一差距在高风险公共卫生数据的背景下尤为关键，例如美国食品药品监督管理局的药品不良事件报告系统（FAERS）[2] 和疫苗不良事件报告系统（VAERS）[3] 中的数据。这些数据集至关重要，因为 FAERS 数据集包含了药物使用过程中报告的不良反应信息，而 VAERS 数据集则包含了患者接种疫苗后报告的症状信息。这些数据集为生成假设和指导公共卫生

*Corresponding author.
srikarre@ksu.edu (S. R. Gadusu); lawrence.callahan@fda.hhs.gov (L. Callahan); samlab88@hotmail.com (S. Lababidi); arunasri.nishtala@fda.hhs.gov (A. Nishtala); hande@ksu.edu (H. K. McGinty)
https://www.koncordantlab.com/ (H. K. McGinty)
0009-0008-2606-4746 (S. R. Gadusu); 0000-0002-1406-0448 (S. Lababidi); 0000-0002-9025-5538 (H. K. McGinty)

决策提供了丰富的资源。高效地将这些数据整合到语义框架中可能会改变研究人员和政策制定者访问医疗保健数据的方式以及推断新见解的能力，从而增强应对公共卫生挑战的反应。

为了解决这些问题，本文提出了一种使用 Python 和 rdflib 库来简化 Neo4j 数据库与 OWL 本体之间连接的用户友好方法。我们的方法不仅自动化了从 Neo4j 数据结构生成类和公理的过程，还使本体数据对更广泛的用户群体更加易于访问。通过将此方法应用于 FAERS 数据集作为用例，我们展示了该方法如何支持动态本体重构，这对于适应快速变化的数据环境至关重要。我们方法的增强可访问性和实用性允许从复杂数据集中生成可行假设，从而扩展公共卫生研究和政策制定的能力。

2. 方法论

为了进一步自动化知识获取与表示方法（KNARM）[1]，实现了三项任务。首先，对美国食品药品监督管理局的药品不良事件报告系统（FAERS）和疫苗不良事件报告系统（VAERS）中的不良事件数据进行了建模。其次，开发了一个图数据库，以高效存储并查询从 FAERS 数据库中提取的嵌套、互连的数据。第三，构建了一个 OWL 本体来表示这些数据集中嵌入的语义关系。该数据集提出了特定的挑战，将在以下子章节中进行讨论。

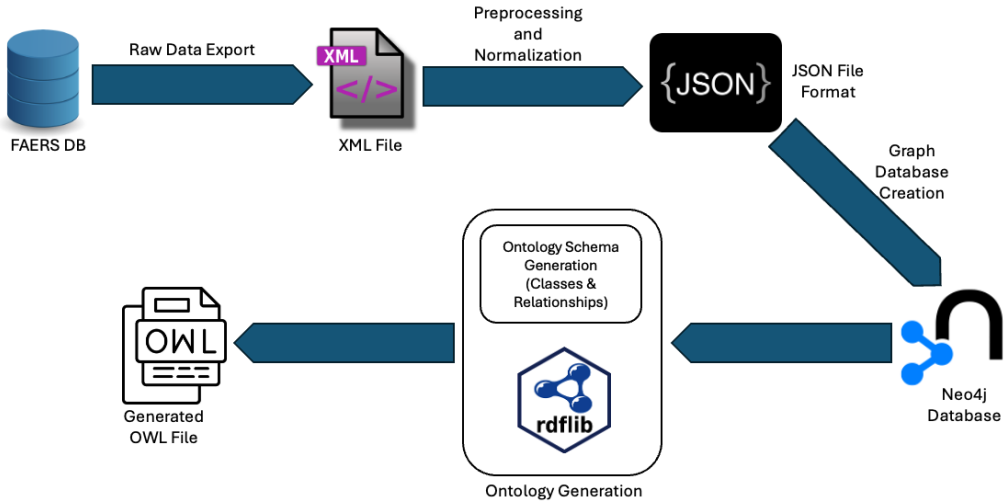

Figure 1: 半自动化本体构建工作流概述。原始不良事件数据从美国食品药品监督管理局不良事件报告系统（FAERS）以可扩展标记语言（XML）格式导出，然后进行预处理并转换为 JavaScript 对象表示法（JSON）。结构化的 JSON 数据使用 Cypher 查询加载到 Neo4j 图数据库中。本体模式生成（类和关系）使用 RDFLib（Python 的资源描述框架库）完成，并将生成的本体序列化为 Web 本体语言（OWL）格式。

除了处理 FAERS 数据外，VAERS 数据集 [3] 也被纳入 Neo4j 数据库中以用于可视化目的。VAERS 数据每年以 CSV 格式分发，每个年度包含三个文件：VAERS 数据文件（包含患者 ID）、症状文件（列出报告的症状）和疫苗文件（记录接种的疫苗）。每个 VAERS ID 对应一个唯一的患者记录。实施了 Cypher 查询来摄取这些 CSV 文件，并将它们转换为 Neo4j 数据库中的节点和关系。生成的图结构如图 2 所示。这种建模方法与后续章节中描述的 FAERS 数据集成非常相似。

2.1. 建模数据和创建数据库

KNARM 方法及其各个步骤在先前的工作中已详细描述 [1]。本研究重点关注通过提出的流程进一步增强的 KNARM 步骤，具体为：*元数据创建与建模*、*数据库形成*和*半自动本体构建*。本节描述了对*元数据创建与建模*和*数据库形成*步骤所做的修改，如整体工作流图（图 **??**）所示。

2.1.1. 来自 FAERS 的数据收集

美国食品和药物管理局（FDA）维护着一个药品不良事件报告系统（FAERS），这是一个用于收集和分享与商业可用药物使用相关的不良事件的数据库。该数据库包含从 2004 年开始的季度报告。每份报告包括患者报告的不良事件，以及所用药物的信息。该数据集提供了大量支持假设生成的信息，特别是在识别与不良事件相关的药物组合方面，这些通常由于个别临床测试和分析的成本和复杂性而研究不足。本研究获得了 FAERS 数据库中的季度数据 [2]。获取的数据以 XML 格式提供，包含关于不良事件的详尽细节，包括含有药物标识、患者人口统计信息以及报告反应的安全报告。

在数据处理之前，对数据库结构进行了全面审查，以记录所有可用字段并熟悉数据集模式。多个字段使用了映射到标准化词汇术语的数字代码；例如，患者年龄组用 1 至 6 的值编码，代表：1=新生儿，2=婴儿，3=儿童，4=青少年，5=成人，和 6=老年人。

所有包含此类映射的相关字段均已识别并记录，以确保在后续的数据转换和分析阶段能够准确解读。基于此次评审，设计了一个初步数据模型来定义 Neo4j 图结构，具体说明了节点和关系如何在数据导入图数据库过程中被创建。

2.1.2. 数据转换

鉴于 FAERS 数据库中的 XML 数据具有结构化但嵌套的特性。这些 XML 数据展现了一种高度嵌套的层次结构。每个*安全报告*元素包含多个子元素，描述了患者的人口统计信息、药物使用情况和相关的不良事件。此外，子元素通常还包含额外的嵌套子元素。例如，在每个*安全报告*中，一个*患者*元素包括进一步的子元素如*药物*和*反应*，每个都包含了它们各自的属性和嵌套数据。由于这种复杂性，直接转换成扁平表格格式（例如 CSV）带来了显著挑战。将数据展平为行的风险在于引入了数据重复，并且损失了关系上下文，这会损害后续图数据库中的数据完整性。

为了保留嵌套关系并最小化数据丢失，XML 数据首先被转换为 JSON 格式。此方法之所以被选中，是因为 JSON 具有直接表示嵌套层次结构的内在能力，从而简化了后续处理步骤，并避免了使用 CSV 等平面格式时出现的复杂问题。

在数据检查过程中，观察到各个安全性报告之间存在不一致。某些报告包含了完整的信息，包括患者详情、给药情况和报告的不良事件，而其他报告则缺少这些关键要素中的部分信息。特别是药物名称或不良事件术语的缺失限制了此类报告对下游分析的实用性，尤其是在识别可能与药物组合相关的不良事件模式时。因此，缺乏药物或不良事件信息的安全性报告被排除在进一步处理之外。

一个 Python 脚本被开发用于自动化数据清理和过滤。该脚本系统地移除了不必要的元素，并排除了缺少药物或不良事件标识的安全报告。经过过滤的数据集仍然以 XML 格式存在，然后使用基于 Python 的解析器将其转换为 JSON，以便支持后续的数据库摄入和本体开发阶段。

来自 VAERS 数据集的结果图形结构示例如图 2 所示。

Figure 2: 网络图展示了两个患者节点（由唯一 ID 标识）在 VAERS 数据集中共享共同症状。

2.1.3. Neo4j 数据库生成

JSON 格式的数据集被导入到 Neo4j 图数据库中。在数据加载之前，安装了 Awesome Procedures On Cypher (APOC) 库以扩展数据库的功能，包括处理 JSON 文件的函数。导入过程使用了 `apoc.load.json` 函数，该函数直接从指定的项目目录读取 JSON 文件。

数据导入通过几个结构化的步骤进行：

- **展开数据：** 使用展开函数迭代从 JSON 文件中提取的安全报告数组中的每个元素。
- **节点创建和匹配：** 利用合并函数基于唯一标识符创建新节点或匹配现有节点。对于每个安全报告，建立了一个对应的安全报告节点，并使用集合命令填充属性。
- **连接节点：** 节点之间的关系已建立于安全报告和患者之间。每个患者通过纳入关系链接到相关药物，并通过有经验的关系链接到报告的不良事件。
- **属性设置和关系构建：** 额外的属性被分配给节点，并使用对每个循环迭代构建关系，以处理每个安全报告中包含的药物和不良事件列表。

2.2. 创建一个 OWL 文件

作为半自动本体构建步骤的一部分，构建了一个本体来正式表示在 FAERS 数据集中记录的不良事件，重点关注可能与报告结果相关的药物组合。由于完整数据集规模庞大，选择了一个具有代表性的子集用于创建本体，以确保高效处理并允许仔细建模相关实体和关系。Neo4j 数据库模式经过精心设计，以符合预期的本体结构，在填充本体阶段可以无缝提取实体和关系。

本体生成是使用基于 rdf 库库 [9] 的定制 Python 应用程序进行的。该过程通过 Python 官方 Neo4j 驱动程序建立与 Neo4j 图数据库的连接来启动，从而能够执行有针对性的 Cypher 查询。这些查询旨在提取四个主要概念类别的基本实体和关系：安全报告、患者、药物和不良事件。提取的数据包括安全报告标识符、患者信息、管理的药物、报告的副作用以及每种药物中包含的相应活性物质。返回的数据被构建为便于高效转换为资源描述框架 (RDF) 格式 [8]，适用于本体构建。

本体初始化涉及使用 rdflib 中的图形 () 函数实例化 RDF 图对象。定义了一个专用命名空间，以系统地组织统一资源标识符 (URI)，确保本体组件的一致性、唯一性和可检索性。该命名空间作为组织所有本体元素（包括类、属性和实例）的基础框架。

本体论模式由四个核心实体的正式类组成：安全报告、患者、药物和不良事件。每个类都通过形式为（主体，RDF.type，RDFS.Class）的三元组在 RDF 图中声明，其中每个主题都被分配了一个来自命名空间的唯一 URI。这些 RDF 声明正式定义了本体论的类层次结构，使得能够对所表示的数据进行语义推理。

为了捕获实体之间的关系，定义了包括有病人、采取了、已报告和是_部分引起的对象属性。这些属性被声明为 OWL. 对象属性实例，而含有活性物质则被建模为 OWL. 数据类型属性以表示诸如药物中的活性物质等字面数据值。所有属性都在命名空间内分配了唯一的 URI，并通过 g.add() 操作被纳入 RDF 图。

为了确保本体内的逻辑一致性，明确指定了每个属性的域和范围限制。例如，属性有患者被限定其域为安全报告，范围为患者，通过形式如（属性，RDFS 域，类）和（属性，RDFS. 范围，类）的三元组进行形式化。这些约束允许定义明确的语义关系，适用于推理和推断任务。

遵循模式定义，RDF 图通过从 Neo4j 数据库中检索的个体实体实例进行了填充。对于每个唯一的安全报告、患者、药物和不良事件出现，创建了一个实例并通过 RDF 类型语句将其链接到其类。每个实例通过分配的 URI 唯一标识。在适用的情况下引入了子类关系，使用 RDFS.subClassOf 语句来捕捉数据中存在的层级结构或专业化分组。

为了进一步强化语义完整性和复杂领域约束，应用了 OWL 约束。这些使用空白节点作为匿名类表达式进行建模。每个空白节点都被声明为与特定属性通过 OWL 上的属性相关联的 OWL. 限制。然后使用 OWL.allValuesFrom 或 OWL.someValuesFrom 应用值约束，以指定每个属性的可接受目标类。这些限制允许对更为细微的领域语义进行建模，特别是对于捕获药物与不良事件之间因果关系的是_部分导致等属性。

模式构建、实例填充和限制建模完成后，使用 g.serialize() 函数将 RDF 图序列化为 OWL 文件。生成的本体文件以./owl 格式保存，可以使用 Protégé 本体编辑器 [10] 进行检查、验证和可视化。这种半自动方法大大简化了本体开发，并实现了对复杂医疗数据的形式语义表示。

3. 结论与未来工作

本研究建立了一个半自动的本体生成框架，整合了来自食品药品监督管理局不良事件报告系统（FAERS）和疫苗不良事件报告系统（VAERS）的数据，使用 Neo4j 进行基于图的数据存储，并结合 Web 本体语言（OWL）进行语义建模，通过 Pythonrdflib 库实现。

所提出的方法解决了将大规模动态演变的公共卫生数据集纳入正式语义框架的关键挑战。通过自动化数据采集、转换和本体生成，该系统提高了知识图谱构建的效率和可重复性，同时保持了语义丰富性。这使得下游应用如假设开发、风险信号检测以及药物安全性数据分析的综合分析成为可能。

通过简化本体论开发，所提出的系统促进了公共健康研究人员和监管机构对复杂生物医学数据集的更广泛访问。例如，该本体论可以支持关于由多药物相互作用引起的新兴不良

药物事件的假设生成，或协助识别特定不良反应风险更高的患者亚组。这些能力有可能加速药物警戒工作，改善公共卫生响应，并最终支持患者安全。

未来的工作将集中在进一步自动化本体生成管道上，以增强可扩展性并减少人工干预。计划改进包括从 Neo4j 数据直接生成本体类，并实现 OWL 本体与 Neo4j 数据库之间的双向转换。这些功能将扩大该方法的应用范围，使其适用于需要快速、准确语义建模的知识工程和决策支持的各种领域。

References

[1] McGinty, H. KNowledge Acquisition and Representation Methodology (KNARM) and Its Applications. (University of Miami,2018)

[2] U.S. Food and Drug Administration, FAERS Public Dashboard, URL: https://-fis.fda.gov/extensions/FPD-QDE-FAERS/FPD-QDE-FAERS.html

[3] U.S. Department of Health and Human Services Vaccine Adverse Event Reporting System (VAERS) Public Dashboard. (Available online), https://vaers.hhs.gov/data/datasets.html, Accessed on [05/2024]

[4] Hitzler, P. & Shimizu, C. Modular Ontologies as a Bridge Between Human Conceptualization and Data. *Graph-Based Representation And Reasoning - 23rd International Conference On Conceptual Structures, ICCS 2018, Edinburgh, UK, June 20-22, 2018, Proceedings*. **10872** pp. 3-6 (2018), https://doi.org/10.1007/978-3-319-91379-7

[5] Kowalczuk, E. & Lawrynowicz, A. The Reporting Event Ontology Design Pattern and Its Extension to Report News Events. *Advances In Ontology Design And Patterns [revised And Extended Versions Of The Papers Presented At The 7th Edition Of The Workshop On Ontology And Semantic Web Patterns, WOP@ISWC 2016, Kobe, Japan, 18th October 2016]*. **32** pp. 105-117 (2016), https://doi.org/10.3233/978-1-61499-826-6-105

[6] Shimizu, C. & Cheatham, M. An Ontology Design Pattern for Microblog Entries. *Proceedings Of The 8th Workshop On Ontology Design And Patterns (WOP 2017) Co-located With The 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 21, 2017*. **2043** (2017), http://ceur-ws.org/Vol-2043/paper-06.pdf

[7] Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter F. Patel-Schneider, and Sebastian Rudolph, *OWL 2 Web Ontology Language: Primer (Second Edition)*, W3C Recommendation 11 December 2012, Available from http://www.w3.org/TR/owl2-primer/, 2012.

[8] Richard Cyganiak, David Wood, and Markus Lanthaler, *RDF 1.1 Concepts and Abstract Syntax*, W3C Recommendation 25 February 2014, Available from http://www.w3.org/TR/rdf11-concepts/, 2014.

[9] RDFLib Development Team. *RDFLib Documentation*. Available at: https://rdflib.readthedocs.io/en/stable/.

[10] M. A. Musen. The Protégé project: A look back and a look forward. *AI Matters*, 1(4):4–12, 2015. Available at: https://protege.stanford.edu/

A. 在线资源

源代码可在以下 github 链接中获取

- https://github.com/koncordantlab/Neo4jToOWL.git