Transforming Text into Knowledge with Graphs: report of the GDR MADICS DOING Action

Mirian Halfeld-Ferrari[1], Anne-Lyse Minard[2], and Genoveva Vargas-Solar[3]

[1] Univ. Orléans, INSA Centre Val de Loire, LIFO UR 4022, FR-45067 Orléans, France
[2] Univ. Orléans, LLL UMR 7270, Orléans, France
[3] CNRS, Univ Lyon, INSA Lyon, UCBL, LIRIS, UMR5205, 69622, Villeurbanne, France
{mirian, anne-lyse.minard}@univ-orleans,genoveva.vargas-solar@cnrs.fr

Abstract. This paper provides an overview on graph databases for the retrieval and the integration of knowledge originating from textual data, attempting to bring together different bricks that are usually addressed separately. It explores concepts and insights that result from the scientific activities promoted by the GDR MADICS DOING Action (Intelligent Data: turning information into knowledge). The action promoted scientific discussion on the challenges, current findings, and open issues in converting textual data into information and, ultimately, knowledge. This topic has been investigated within a multidisciplinary context, involving specialists in Databases (DB), Natural Language Processing (NLP), Artificial Intelligence (AI), and professionals in various application domains.

Keywords: Text processing · graph data models· graph analytics · data science queries on graphs.

1 Overview

This paper explores concepts and insights related to graph-based modelling of textual content that result from scientific activities promoted in the coordination action Intelligent Data: turning information into knowledge (DOING) [4]. The action promoted scientific discussion on the challenges, current findings, and open issues in converting textual data into information and, ultimately, knowledge. Leveraging expertise from scientists in natural language processing (NLP), databases (DB), and artificial intelligence (AI), the study and reflection focuses on two main research directions: extracting valuable insights from textual data to enrich graph databases and developing intelligent and user-friendly techniques for effectively maintaining, querying, and conducting analytical studies

[4] DOING is a coordination action funded by the network MADICS of the French Council of Scientific Research - CNRS http://www.madics.fr/actions/doing/. Created as a regional initiative in 2019, DOING extended its scope to a national level within the GDR MADICS in 2020 before attaining official status as an action.

while ensuring quality standards. The application domains of the study encompass medical and environmental fields.

The working method adopted by the coordination action[5] has addressed diverse perspectives about the problem and associated challenges through webinars, study days, featuring discussions, panels, and roundtable sessions to synthesize viewpoints from diverse scientific communities and methods for achieving the transformation of data into knowledge. Challenges and insights have emerged in aligning communication among scientists from diverse domains, primarily from vocabulary and conceptual understanding discrepancies. Fostering fruitful interaction and facilitating collaborations between French colleagues and international partners, including those engaged in multidisciplinary research projects in Brazil, DOING has produced a multi-perspective understanding of extracting, structuring, and querying textual content using graphs for knowledge extraction.

The research contribution of this paper is to make an overview on graph databases for the retrieval and the integration of knowledge originating from textual data, attempting to bring together different bricks that are usually treated separately. In this paper, we summarise the concepts, remarks and scientific vision about adopting graph databases as the primary component for organizing information extracted from texts and developing an intelligent querying framework. This framework allows users to execute analytic queries on data within the graph, targeting diverse user requirements. Therefore, the remainder of the paper is organized as follows: Section 2 provides an overview of the fundamental concepts of graph databases. Section 3 delves into textual data structuring, while Section 4 enumerates the components and functions of an intelligent query framework. Section 5 concludes the paper by underscoring the significance of multidisciplinarity (within the computer science domain) in this study.

2 Graph Databases: the structure

A graph database management system (GDBMS) employs graphs as its fundamental data model. A graph structures data as a "network" of entities and relations, for example, in the Semantic Web, social networks, connected businesses, digital networks, knowledge networks, and Internet networks.

GDBMSs are engineered to prioritize the relationships between data that are equally significant as the data itself. These relationships enable stored data to be directly linked together, often facilitating rapid retrieval. Consequently, graph query languages are formulated to articulate paths on a graph, highlighting the interconnected nature of the stored information. A native GDBMS is purposefully designed to handle graph workloads across the entire computing stack.

[5] The impact of the DOING coordination action goes beyond national boundaries, inspiring international initiatives: (i) A regional project, APR-IA, supported by the Centre Val de Loire region (2021-2024). (ii) The international workshop DOING@ADBIS, marking its 5th edition this year, underscores the far-reaching influence of DOING.

2.1 Graph models and query languages

RDF and Property graphs. Graph databases adopt two models: LPG (labelled-property graphs or just property graphs) and RDF (Resource Description Framework) graphs. Property graphs prioritize analytics and querying, whereas RDF graphs underscore the importance of data integration. Both models are centered around the concept of node- and edge-labelled graphs, allowing for the encoding of intricate information in multi-graphs. Unlike RDF graphs, which consist of nodes and arcs, a property graph augments nodes or edges with attributes or properties. Each property is represented as a pair (key, value), enhancing graph data. In the study and discussion promoted by the DOING community, LPG is favored for its alignment with classical database concepts. It utilizes properties to precisely define nodes and relations, making primary relationships among entities visible. This approach prevents dense (RDF) graphs, where relations serve as links between entities and as a means to express entity properties. Neo4J stands out as the most popular property graph database system, but other noteworthy options exist, such as Memgraph and TigerGraph.

Query languages. Various query languages have emerged for querying graphs, such as SPARQL for RDF graphs, Cypher for property graphs in Neo4J, and Gremlin for property graphs in Apache TinkerPop. Despite differences in style, purpose, and implementation, these languages share a core focus on two fundamental operations: *graph pattern matching* and *graph navigation*.

Graph pattern matching involves identifying matches of a graph pattern within a graph database. In basic graph patterns, variables can represent node labels or relation types. Matches are defined as homomorphisms from the pattern to the graph instance. Various semantics exist for determining these matches. Navigational queries offer versatile querying methods that enable the exploration of the database's *topology*.

A path query specifies conditions that paths on a graph must meet. Paths can be defined using regular expressions denoted by L. Evaluation of a path query over a graph involves finding all paths whose labels satisfy L. Handling potentially infinite answers to regular path queries requires adopting specific semantics to ensure finiteness.

Designing a graph database. There is no methodology for designing a graph database for a dataset. Instead, there are general guidelines that can help model a representative graph. Unlike relational databases, the design of a graph database significantly influences query performance. Queries not optimized for the database model can result in poor performance. Graph databases excel in path traversal queries, as nodes typically store information about their neighboring nodes. This characteristic makes them attractive for exploring data relationships and employing data analytics techniques such as predicting node connections. Therefore, it is crucial to consider the types of queries the application will handle and design the graph database accordingly[6] [27].

[6] `https://cambridge-intelligence.com/graph-data-modeling-101/`

2.2 Related Work in the DOING Community

Graph databases and their query languages have been a recurring focus in the seminars organized within the DOING coordination action. George Fletcher's research group at Eindhoven University of Technology, has emphasized the expressive power of graph query languages. They are particularly interested in characterizing the ability of these languages to constrain and shape concrete graph instances based solely on their structural properties[7]. Meanwhile, the research group led by Domagoj Vrogoc at Pontificia Universidad Católica de Chile[8] focuses on the theoretical aspects of queries that facilitate traversing paths of arbitrary lengths. Additionally, the LIGM-CNRS[9] group delves into theoretical and practical aspects of query languages for property graphs, providing a foundational basis for languages like Neo4J.

3 Structuring textual data

Databases store structured information, enabling efficient and practical querying. When working with unstructured textual data for answering users' queries, structuring the data allows leveraging the full range of database functionalities, improving accessibility. This context shows the intersection of two disciplines operating at different levels of abstraction: databases (DB) and natural language processing (NLP). The database perspective relies on high-level abstraction modelling that abstractly represents the original text, emphasizing the specification of concepts and their interrelationships. The objective is to encapsulate the types of essential information to be preserved. Conversely, the NLP community pursues a lower level of abstraction that directly refers to textual data without modeling concepts that genelarize data.

The *database approach* (DB) contrasts with the *textual graph approach* that uses graphs to represent the text content directly *i.e.*, without attempting to deal with the abstractions a general schema would represent. The view promoted in the DOING action brings together the DB and NLP communities to integrate their perspectives to explore and promote the interests of their complementarity. Figure 1 illustrates the *database approach* with a graph database that can be the result of structuring, as a property graph, information obtained from the given text (an extract of PubMed).

The key idea is that the schema comprises notions that aggregate or specialize concepts detected in the text. For instance, the database schema may store *generalized* information concerning a *symptom* (*temperature of 39° C*), *i.e.*,

[7] DOING Webinar: Language-aware indexing for conjunctive path queries, George Fletcher, Eindhoven University of Technology, Netherlands, Mars,2021

[8] DOING Webinar: Evaluating navigational queries over graphs, Domagoj Vrogoc, Pontificia Universidad Católica de Chile, Chili, July 2021

[9] MADICS Symposium, DOING workshop. On July 2022: Aperçu général des langages de requêtes pour graphes à propriétés by Victor Marsault. On May 2023: A Researcher's Digest of GQL by Liat Peterfreund

A female patient in the age group 55-60 years presented to us with blurring of vision in both eyes. On slit-lamp examination, numerous circular to oval fleck-like discrete blue opacities at the level of deep corneal stroma and Descemet's membrane was observed.

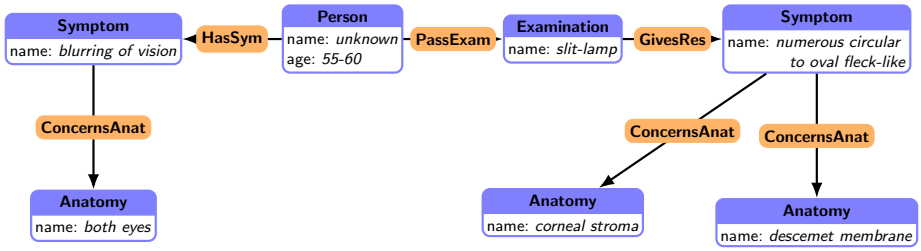

Fig. 1. Example of a Neo4J graph database instance obtained from the given text (edge properties are not shown).

aggregating the variable *temperature* and its value $39° C$ in a unique entity. The schema may store *specialized* information concerning a *symptom* (e.g., *discrete blue opacities*) and the *anatomy* (e.g., *the cornea*). The database approach focuses on how to instantiate the classes composing the database schema: on how to populate the graph specifying instances of nodes and edges types defined by a graph database schema. A structured instance built according to the schema is the database on top of which queries and analytics are performed.

An essential challenge of NLP is grouping initial NLP abstractions into more generalized ones. For instance, in the phrase *500 mg of paracetamol*, entities value (500) and unit (*mg*) can be grouped into the concept of dosage, which, along with the entity drug, can be further grouped into the concept of treatment. For instance, this latter concept can be represented as a node and attributes in an LPG model.

The study and discussion promoted in DOING are guided by the hypothesis that data from texts are instances of a database whose schema is (or is not) predefined. Accordingly, structuring textual data implies using NLP tools and techniques and relying on a higher level of abstraction by grouping and occasionally factorizing the extracted information. This hypothesis assumes that information may only sometimes be uniformly represented, leading to the acceptance of missing data (often seen as anomalies that can harm the database consistency and query answer completeness).

3.1 Information Extraction

NLP techniques can accurately extract structured semantic information from unstructured texts and insights to categorise and organise them. These techniques can help uncover insights contained within text in a scalable and efficient manner by identifying and analysing explicit concepts and relationships. Information extraction includes two tasks: entity recognition and relation extraction.

Entity recognition (NER) Consists in detecting and classifying entities in the text. The most current detected entities are the so-called named entities, which traditionally include person names, location names, organisation names and numerical expressions. In the medical domain, the entities of interest can be signs or symptoms, pathology, drugs, clinical tests, etc. [24] In the example text of Figure1, an entity recognition system can be used to detect the anatomy information, the symptoms, the examination and the patient's age. For the most common entity classes in English or French, some existing annotated corpora can be used to train supervised machine learning algorithms. For example, for the medical domain, we can cite the CAS corpus [15] (French) or the E3C corpus [20] (five languages, including English and French). In other specialised domains (e.g. geological domain), it is sometimes necessary to use techniques other than supervised machine learning, such as dictionary-based extraction or rules, because no annotated corpora are available. A BERT-based language model can be used to apply many machine-learning algorithms, such as CRF, Bi-LSTM CRF, or transformers. Once entities have been detected and classified, an entity-linking task can be performed. It consists of a disambiguation task by linking entities to a unique identifier. These identifiers can come from a knowledge base (e.g., DBpedia), a thesaurus (e.g., UMLS for the medical domain), or other knowledge representations. It is an important step to reduce variation and ambiguity while structuring textual information into a graph database.

Relation extraction (RE) Consists in detecting and classifying the relationships between entities. Different types of relations can be of interest, mainly temporal, causal, and other semantic relations between specific entities. In the example text of Figure 1, a RE system can be used to extract the temporal relation between the symptom "blurring of vision" and the examination or the semantic relation "ConcernsAnat" that links a symptom and a specific anatomy element. Various machine learning algorithms, such as SVM, CNN, and RNN, are used for the RE task. Bi-LSTM is used for long-distance relations. Unfortunately, few annotated corpora exist for relation extraction, especially for French, which makes the task more difficult to resolve. Syntactic structures, such as constituency trees [23] or dependency trees [28], are often exploited to detect the relations in supervised or rule-based systems.

The RE task often focuses on binary relations, but extracting n-ary relations or grouping entities is also needed. For example, in the text shown in Figure 1, all information about one patient must be gathered: his gender, his age, his origin, etc. In medical documents, it is also often the case for the medication for which we have various information: substance, dosage, frequency, method for administration, etc. However, this task is rarely treated in the literature, and few annotated corpora are available.

The described information extraction tasks and techniques can structure textual information into graph databases. The main difficulty is the need for large quantities of annotated data to train supervised learning systems, particularly for relation extraction. The following lines present different perspectives for textual structuring depending on how an annotation schema is implied.

3.2 Exploring Approaches for Textual Structuring

Text structuring can be done through top-down and bottom-up methodologies. In the top-down approach, a schema is predetermined, tackling the problem as a query of the text to extract or identify pertinent "relevant" entities. Conversely, the bottom-up approach involves extracting terms, entities, and relationships from the text, followed by their classification and grouping, often based on a similarity distance metric. Hybrid methodologies, which combine these two perspectives, have demonstrated potential for enhanced efficiency.

Top-down methods can rely on information extractors and ontologies. In the theoretical database community, extractors are viewed as functions [25], referred to as *spanners*, designed to extract a relation of spans from a document d. In this context, d is a string over a finite alphabet, and a span x is an interval of positions within d that represents a substring d_x. Regular spanners are the most studied. However, they are not capable of handling various complex scenarios found in natural language, such as nested structures. Significant formal advancements have been made in this field, offering insights into the expressive power (such as [11,26]) and complexity of these extractors (such as [1,14]).

Bottom-up methods implement an Open Information Extraction task (OpenIE) to extract triples from raw texts without using pre-defined relations or an annotation schema [3]. OpenIE systems rely on linguistic features obtained with PoS tagging, chunking or dependency parsing. Other bottom-up methods are applied for ontology learning, defining a sequence of tasks [10,29]. (1) First, identify the natural language terms associated with the target domain. (2) Then, identifying synonyms to avoid redundant concepts, since two or more natural language terms can represent the same *concept*. The notion of "concept" is controversial but is usually associated with a knowledge abstraction that can be found with some regularity in a text from a given domain. Finding concepts in a text means finding regularities that the same abstraction can generalize. Most research in this area uses unsupervised machine learning techniques like clustering [6]. However, some works use supervised learning methods to improve results [13]. When dealing with ontologies, hierarchical relationships (the *is_a* relationship) between pairs of concepts are established after determining the concepts. This step leads to a taxonomy of concepts. (3) The following task concerns finding non-hierarchical relations. Usually, it combines statistical analysis and linguistic analysis as in [5]. (4) In ontology learning, the last step focuses on discovering inference rules from text, such as *If X is author of Y*, then *X wrote Y*. Constructing a database schema from text follows the same steps except for finding taxonomies.

Modelling a database under a bottom-up approach by structuring textual data implies following learning steps (i.e., similar to ontologies). For instance, in [16], syntax trees enriched with entities and relationships found in a preprocessing step are analysed, and similar sub-trees are grouped until reaching a tree format that respects specified rules. This strategy can be considered a hybrid method but connected to the ontology learning ideas.

While there is a notion that a database schema and an ontology may be equivalent, it is essential to recognize their distinct purposes. A database schema describes a set of instances tailored for efficient storage and querying, for example, structuring data in graph databases profoundly influencing query efficiency. Conversely, while constraints in an ontology primarily convey machine-readable semantics to facilitate automated reasoning, those in databases mainly serve to ensure data integrity. In contrast, an ontology, which may serve as a schema, serves broader objectives such as interoperability, reasoning, and knowledge representation.

3.3 Related Work in the DOING Community

The study of the DOING action discussed domain-specific, domain adaptation and domain-independent systems. The research conducted at the LISN by Aurélie Névéol [10] and by Perceval Wajsbürt at HP-HP [11] show concrete applications of domain specific systems for information extraction in the medical domain.

Another relevant topic DOING addresses is modelling extracted information and large-scale textual structuring. Davide Buscaldi at the LIPADE [12] has developed research on knowledge graph generation from scientific literature [9]. The SCICERO system described in [9] extracts entities and relations, then merges entities and relations, leading to a set of triples used to build the knowledge graph.

4 Towards intelligent querying on graph databases

Graph querying techniques can be grouped into two families. The first encompasses traditional querying approaches commonly adopted by database and information retrieval engines [30,12]. These techniques allow the expression of graph traversal operations. In this scenario, results are characterized by their completeness concerning the graph database being queried [4]. The second family, in contrast, concerns the analysis of graphs with modelling, prediction objectives and recommendations. These data (i.e., graph) samples are processed with artificial intelligence algorithms (i.e., models), suggesting that outcomes are based on partial or representative data sets rather than exhaustive analyses. This distinction highlights the varying approaches to data querying, emphasizing thoroughness and accepting and accounting for uncertainty in its analysis.

[10] DOING Webinar: Natural language processing for epidemiology & public health, Aurélie Névéol, CNRS, LISN, France, 5th July, 2021

[11] MADICS Symposium, DOING workshop: Tools for processing clinical reports in health data warehouses, Perceval Wajsbürt, Assistance de Paris – Hôpitaux de Paris (AP-HP), France, 25th June, 2023

[12] DOING Webinar: Building Scientific Knowledge Graphs from Scholarly Data, Davide Buscaldi, LIPN, Université Sorbonne Paris Nord, France, 17th May, 2021

Graph analytics, a technique for analyzing data in a graph format, has attracted substantial attention in contemporary decision-making processes. Analyzing data in this way makes it possible to discover links and relationships that would otherwise escape traditional methods. Graph stores with different models and properties, querying facilities, and analytics libraries with built-in graph analytics algorithms provide tools for exploring graphs. The question is under which conditions the various solutions are better adapted to address different analytics (i.e., intelligent) queries.

4.1 Data Science Pipelines on Graphs

The data science pipeline involves a series of steps to gather raw data from multiple sources, analyse it, and present the results in an understandable format. This process can be applied to constructing, exploring, and analysing graphs. General templates for these tasks can be adapted based on the algorithms and strategies used.

Data analytics queries on graph stores like Neo4j, Amazon Neptune, and TigerGraph offer built-in graph operations that implement graph analytics operations such as community detection, centrality (e.g., PageRank and betweenness algorithms), similarity and pathfinding, and search operations. While these systems enable declarative querying of stored graphs, users are responsible for memory management and routing graph components to the execution space when applying graph analysis functions. An example of analyzing a graph built from medical text using PageRank with Neo4j involves constructing a graph representation of the medical text data within the Neo4j database.

```
    // Step 1: Create graph nodes and relationships
    from medical text data
LOAD CSV WITH HEADERS FROM 'file:///medical_data.csv' AS row
MERGE (m:MedicalTerm {name: row.term})
MERGE (d:Disease {name: row.disease})
MERGE (m)-[:MENTIONS]->(d);
```

This graph likely includes nodes representing medical terms, diseases, symptoms, treatments, and other relevant entities, with their relationships based on semantic connections extracted from the text. The PageRank algorithm is applied to calculate the importance of each medical term node in the graph based on the relationships between nodes.

```
     // Step 2: Run the PageRank algorithm to
     calculate node importance
CALL algo.pageRank(
  'MATCH (m) RETURN id(m) as id',
  'MATCH (m)-[:MENTIONS]->() RETURN id(m) as source, id() as target',
  {write: true, writeProperty: 'pagerank'}
);
```

The results are retrieved, and the top 10 medical terms with the highest PageRank scores are returned, providing insights into the most significant terms within the medical text data.

```
    // Step 3: Retrieve results and inspect the
    nodes with highest PageRank scores
MATCH (m:MedicalTerm)
RETURN m.name, m.pagerank
ORDER BY m.pagerank DESC
LIMIT 10;
```

The pipeline is thus a sequence of "queries" that operate step by step on the "prepared" graph to produce results. Various factors come into play when considering graph store solutions for applying machine learning algorithms on stored graphs. These factors include performance, scalability, flexibility, built-in algorithms, ease of use, community support, and cost.

Imperative approaches for implementing graph analytics pipelines. Data mining and machine learning techniques can be integrated into imperative programs to analyze graphs, where analytics pipelines' control and data flow are implicit. Data analytics environments often utilize libraries with proprietary data structures and built-in functions for graph creation and analysis [31]. Examples include Python Networkx, Spark Graphix, and deep learning models (e.g., graph neural networks) for tasks like node classification, link prediction, and graph clustering, as well as graph processing systems like Pregel and Giraph. A typical pipeline constructs an ad hoc graph representing the data corresponding to data preparation and engineering tasks. Subsequently, the analytics code sequence applies functions with calibrated parameters to analyze the prepared graph. The iterative trial-and-error process for parameter tuning is not explicitly coded but can be automated using other machine-learning techniques.

The pipelines may encompass various tasks to convert graphs into vector representations suitable for training, testing, and evaluating Graph Neural Network (GNN) models. The preprocessing pipeline's complexity can vary depending on the graph type (e.g., heterogeneous or homogeneous, directed or undirected, properties in nodes and/or edges). For example, it may involve extracting nodes and edges separately, along with their properties, and transforming them into vectors using techniques like text2vec. Alternatively, for heterogeneous graphs, one may select graph views where nodes and edges share the same type and compute a global vector to represent the entire graph.

4.2 Related Work in the DOING Community

The study done by the DOING community focuses analytical queries on graphs, seeking to integrate database and machine learning techniques to enable the answering of more sophisticated queries. Imperative approaches rely on libraries and execution environments with no built-in options for managing graph views,

resource allocation and graph persistence. In contrast, declarative approaches relying on underlying graph management systems profit from the manager's strategies for managing the graphs on disk and main memory. A collaborative experimental comparison [31] done by the labs LIFO, University of Orléans and LIRIS, CNRS lab in Lyon lead to a tutorial and talk in the contexto of DOING.

We discussed the specification and design of data science pipelines that could process graphs representing textual content. We worked with experts from different disciplines, including databases, artificial intelligence and natural language processing, to address these questions.The work done in Paris at the Leonardo da Vinci School of Engineering (ESILV[13]) by the group of Nicolas Travers [17] and at the LIPADE and LIPN [18,22][14] have developed research with the bottom up approach where graphs are designed from input datasets considering the type of analysis to perform. The work in U. Manchester by Andre Freitas[30][15] relies on graph representations of textual content to perform exploration queries for retrieving factual data and analytics operations for discovering knowledge within the graph. A set of data science pipelines adopting different analytics techniques lead to the construction of knowledge graphs at the LIRIS in Lyon [8]. Experts from the DB and AI communities from the University of Tours in France provided insight into graphs analytics with neural network models [19,7] [16].

5 Multidisciplinary Context and Concluding Remarks

This paper summarises insights about the problem of structuring textual content and transforming it into knowledge from a multidisciplinary perspective (within the computer science domain). The discussion conclusions show how the NLP, DB and AI fields reason about textual data. In NLP and DB, the guiding challenge is proposing different forms of abstraction. While the NLP domain focuses on numerous details and variants of unstructured data, the DB domain typically necessitates abstractions that consolidate information into unified concepts or relation types. For both the DB and AI domains, the emphasis lies in reasoning—exploring, correlating facts, and uncovering implicit knowledge —from data extracted from textual content. As AI methods are employed for querying, there is a shift in perspective within the DB domain, moving from certainties to possibilities and from general deduction to a reliance on samples and instantiations to yield results. Throughout the transformation of textual data into knowledge,

[13] `https://www.esilv.fr`

[14] DOING Webinars: Managing data quality in the age of big data, Salima Benbernou, Université Paris Descartes, LIPADE, France, 5th June 2020; Building Scientific Knowledge Graphs from Scholarly Data, Davide Buscaldi, LIPN, Université Sorbonne Paris Nord, France, 17th May, 2021

[15] DOING Webinar: From Deep Learning to Deep Semantics, Andre Freitas, University of Manchester, UK, 8th July 2020.

[16] DOING Panel, *Representing content and extracting knowledge from texts: automatic language learning, graph management systems, machine learning for graph analysis and semantic web approaches*, Symposium MADICS, Lyon, 2022

database management systems play a mediating role. (G)DBMS facilitate the structuring of textual content in NLP practices and applying AI techniques on structured textual content for analysis and knowledge extraction.

The challenge lies in blurring the boundaries among the three disciplines and integrating database concepts into both areas while simultaneously incorporating their respective principles. For example, leveraging database techniques for efficient storage, retrieval, and manipulation of textual data can significantly improve the performance and scalability of NLP and AI systems. Ultimately, this integration of databases, NLP, and AI principles drives innovation and paves the way for more robust and practical solutions in data-driven endeavors.

The study conducted by the DOING coordination action has identified a primary requirement: the development of generic tools to facilitate communication between different system components in collaboration with other scientific domains, particularly in the medical and environmental fields. Such tools involve reasoning on a metamodel, which serves as a framework onto which information can be initially mapped and reconfigured into a more specific model. An example of this principle is the approach proposed in [16], which utilizes a general abstraction to guide textual data into a tree structure corresponding to a formal grammar parse tree, adaptable to different database models. Despite the natural inclination towards graph models as a mode of representation, institutions like BRGM [17] are seeking metamodels to postpone adhering to specific models for modeling content. This trend is evident in other recent works such as [2,21].

The DOING coordination action has enabled a collaborative and multidisciplinary exploration of transforming information into knowledge, making data intelligent. DOING highlights the significance of fostering synergy between diverse scientific disciplines by identifying potential interactions among methods and strategies in NLP, DB, and AI. Such an approach underscores the importance of technological advancements in effectively addressing real-world challenges in medicine, environment, geography and beyond.

Acknowledgements

This work was partially supported by the DOING project, a regional project funded by the council of the Centre Val de Loire Region (APR-IA).

References

1. Amarilli, A., Bourhis, P., Mengel, S., Niewerth, M.: Constant-delay enumeration for nondeterministic document spanners. In: ICDT. LIPIcs, vol. 127, pp. 22:1–22:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2019)
2. Balalau, O., Ebel, S., Galizzi, T., Manolescu, I., Massonnat, Q., Deiana, A., Gautreau, E., Krempf, A., Pontillon, T., Roux, G., Yakin, J.: Statistical claim checking: Statcheck in action. In: Hasan, M.A., Xiong, L. (eds.) Proceedings of the

[17] French geological survey `https://www.brgm.fr/en`

31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, October 17-21, 2022. pp. 4798–4802. ACM (2022)
3. Banko, M., Cafarella, M.J., Soderland, S., Broadhead, M., Etzioni, O.: Open information extraction from the web. In: Proceedings of the 20th International Joint Conference on Artifical Intelligence. pp. 2670–2676. IJCAI'07, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2007)
4. Bonifati, A., Fletcher, G., Voigt, H., Yakovets, N., Jagadish, H.: Querying graphs, vol. 10. Springer (2018)
5. Buitelaar, P., Olejnik, D., Sintek, M.: A protégé plug-in for ontology extraction from text based on linguistic analysis. In: ESWS. Lecture Notes in Computer Science, vol. 3053, pp. 31–44. Springer (2004)
6. Cimiano, P., Hotho, A., Staab, S.: Learning concept hierarchies from text corpora using formal concept analysis. J. Artif. Intell. Res. **24**, 305–339 (2005)
7. Conte, D.: Graphs in pattern recognition: successes, shortcomings, and perspectives. Journal of Electronic Imaging **32**(2), 020701–020701 (2023)
8. Coste, L., Helmers, F., Kheddouci, H., Le Nestour, L., Niazi, M., Vargas-Solar, G.: Strategies for creating knowledge graphs to depict a multi-perspective queer communities representation. In: Workshops of the EDBT/ICDT 2023 Joint Conference. vol. 3379 (2023)
9. Dessí, D., Osborne, F., Reforgiato Recupero, D., Buscaldi, D., Motta, E.: Scicero: A deep learning and nlp approach for generating scientific knowledge graphs in the computer science domain. Knowledge-Based Systems **258**, 109945 (2022)
10. Drummond, L., Girard, R.: A survey of ontology learning procedures. In: Proceedings of the 3rd Workshop on Ontologies and their Applications (2008)
11. Fagin, R., Kimelfeld, B., Reiss, F., Vansummeren, S.: Document spanners: A formal approach to information extraction. J. ACM **62**(2), 12:1–12:51 (2015)
12. Farokhnejad, M., Pranesh, R.R., Vargas-Solar, G., Mehr, D.A.: S_covid: An engine to explore covid-19 scientific literature. In: Proceedings of the 24th International Conference on Extending Database Technology (EDBT), Nicosia, Cyprus. pp. 23–26 (2021)
13. Faure, D., Nedellec, C.: Knowledge acquisition of predicate argument structures from technical texts using machine learning: The system ASIUM. In: EKAW. Lecture Notes in Computer Science, vol. 1621, pp. 329–334. Springer (1999)
14. Florenzano, F., Riveros, C., Ugarte, M., Vansummeren, S., Vrgoc, D.: Constant delay algorithms for regular document spanners. CoRR **abs/1803.05277** (2018)
15. Grabar, N., Claveau, V., Dalloux, C.: CAS: French corpus with clinical cases. In: Proceedings of the Ninth International Workshop on Health Text Mining and Information Analysis. pp. 122–128. Association for Computational Linguistics, Brussels, Belgium (Oct 2018)
16. Hiot, N.: Phd. thesis (in preparation)
17. Lefebvre, P., Moal, S.L., Azough, A., Travers, N.: Neosgg: A scene graph generation framework for video-surveillance tasks. In: Proceedings 27th International Conference on Extending Database Technology, EDBT 2024, Paestum, Italy, March 25 - March 28. pp. 838–841. OpenProceedings.org (2024)
18. Lovera, F., Cardinale, Y., Buscaldi, D., Charnois, T.: A knowledge graph-based method for the geolocation of tweets. In: Workshop Proceedings of the 19th International Conference on Intelligent Environments (IE2023). pp. 53–62. IOS Press (2023)
19. Maekawa, S., Sasaki, Y., Fletcher, G., Onizuka, M.: Benchmarking gnns with gencat workbench. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 607–611. Springer (2022)

20. Magnini, B., Altuna, B., Lavelli, A., Speranza, M., Zanoli, R.: The E3C project: Collection and annotation of a multilingual corpus of clinical cases. In: Proceedings of the Seventh Italian Conference on Computational Linguistics, CLiC-it 2020, Bologna, Italy, March 1-3, 2021. CEUR Workshop Proceedings, vol. 2769 (2020)
21. Mali, J., Ahvar, S., Atigui, F., Azough, A., Travers, N.: A global model-driven denormalization approach for schema migration. In: Research Challenges in Information Science - 16th International Conference, RCIS 2022, Barcelona, Spain, May 17-20, 2022, Proceedings. Lecture Notes in Business Information Processing, vol. 446, pp. 529–545. Springer (2022)
22. Mammar Kouadri, W., Benbernou, S., Ouziri, M., Ben Amor, I.: Wssa: Weakly supervised semantic-based approach for sentiment analysis. In: Proceedings of the 34th International Conference on Scientific and Statistical Database Management. pp. 1–4 (2022)
23. Minard, A.L., Ligozat, A.L., Grau, B.: Apport de la syntaxe pour l'extraction de relations en domaine médical. In: TALN 2011. p. 383. Montpellier, France (Jun 2011)
24. Minard, A., Roques, A., Hiot, N., Halfeld Ferrari, M., Savary, A.: DOING@DEFT : cascade de CRF pour l'annotation d'entités cliniques imbriquées (DOING@DEFT: cascade of CRF for the annotation of nested clinical entities). In: Actes de la 6e conférence conjointe Journées d'Études sur la Parole (JEP, 33e édition), Traitement Automatique des Langues Naturelles (TALN, 27e édition), Rencontre des Étudiants Chercheurs en Informatique pour le Traitement Automatique des Langues (RÉCITAL, 22e édition). Atelier DÉfi Fouille de Textes, Nancy, France, June 8-19, 2020. pp. 66–78. ATALA et AFCP (2020)
25. Peterfreund, L.: Grammars for document spanners. In: ICDT. LIPIcs, vol. 186, pp. 7:1–7:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021)
26. Peterfreund, L., ten Cate, B., Fagin, R., Kimelfeld, B.: Recursive programs for document spanners. In: ICDT. LIPIcs, vol. 127, pp. 13:1–13:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2019)
27. Prevoteau, H., Djebali, S., Laiping, Z., Travers, N.: Propagation measure on circulation graphs for tourism behavior analysis. In: Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing. pp. 556–563 (2022)
28. Savary, A., Silvanovich, A., Minard, A., Hiot, N., Halfeld Ferrari, M.: Relation extraction from clinical cases for a knowledge graph. In: New Trends in Database and Information Systems - ADBIS 2022 Short Papers, Doctoral Consortium and Workshops: DOING, K-GALS, MADEISD, MegaData, SWODCH, Turin, Italy, September 5-8, 2022, Proceedings. Communications in Computer and Information Science, vol. 1652, pp. 353–365. Springer (2022)
29. Toledo-Alvarado, J.I., Guzman-Arenas, A., Luna, G.L.M.: Automatic building of an ontology from a corpus of text documents using data mining tools. Journal of Applied Research and Technology **10**, 398–404 (2012)
30. Valentino, M., Ferreira, D., Thayaparan, M., Freitas, A., Ustalov, D.: Textgraphs 2022 shared task on natural language premise selection. In: Proceedings of TextGraphs-16: Graph-based Methods for Natural Language Processing. pp. 105–113 (2022)
31. Vargas-Solar, G., Marrec, P., Halfeld Ferrari Alves, M.: Comparing graph data science libraries for querying and analysing datasets: towards data science queries on graphs. In: International Conference on Service-Oriented Computing. pp. 205–216. Springer (2021)