Report from Dagstuhl Seminar 22372

Knowledge Graphs and their Role in the Knowledge Engineering of the 21st Century

Paul Groth[*1], **Elena Simperl**[*2], **Marieke van Erp**[*3], **and Denny Vrandečić**[*4]

1 **University of Amsterdam, NL.** p.t.groth@uva.nl
2 **King's College London, GB.** elena.simperl@kcl.ac.uk
3 **KNAW Humanities Cluster, NL.** marieke.van.erp@dh.huc.knaw.nl
4 **Wikimedia Foundation, San Francisco, US.** denny@wikimedia.org

Abstract

This report documents the programme and outcomes of Dagstuhl Seminar 22372 "Knowledge Graphs and their Role in the Knowledge Engineering of the 21st Century" held in September 2022.

The seminar aimed to gain a better understanding of the way knowledge graphs are created, maintained, and used today, and identify research challenges throughout the knowledge engineering life cycle, including tasks such as modelling, representation, reasoning, and evolution. The participants identified directions of research to answer these challenges, which will form the basis for new methodologies, methods, and tools, applicable to varied AI systems in which knowledge graphs are used, for instance, in natural language processing, or in information retrieval.

The seminar brought together a snapshot of the knowledge engineering and adjacent communities, including leading experts, academics, practitioners, and rising stars in those fields. It fulfilled its aims – the participants took inventory of existing and emerging solutions, discussed open problems and practical challenges, and identified ample opportunities for novel research, technology transfer, and inter-disciplinary collaborations. Among the topics of discussion were: designing engineering methodologies for knowledge graphs, integrating large language models and structured data into knowledge engineering pipelines, neural methods for knowledge engineering, responsible use of AI in knowledge graph construction, other forms of knowledge representations, and generating user and developer buy-in. Besides a range of joint publications, hackathons, and project proposals, the participants suggested joint activities with other scientific communities, in particular those working on large language models, generative AI, FAccT (fairness, accountability, transparency), and human-AI interaction.

The discussions were captured in visual summaries thanks to Catherine Allan – you can find more about her work at https://www.catherineallan.co.uk/. The summaries are arrayed throughout this report. Lastly, knowledge about the seminar is captured in Wikidata at https://www.wikidata.org/wiki/Q113961931

Seminar September 12–14, 2022 – http://www.dagstuhl.de/22372

2012 ACM Subject Classification Computing methodologies → Knowledge representation and reasoning; Computing methodologies → Natural language processing; Computing methodologies → Machine learning; Information systems → Information retrieval; Computing methodologies → Ontology engineering; Computing methodologies → Reasoning about belief and knowledge; Human-centered computing → Collaborative and social computing theory, concepts and paradigms

Keywords and phrases Dagstuhl Seminar

Digital Object Identifier 10.4230/DagRep.12.9.60

[*] Editor / Organizer

1 Executive Summary

Marieke van Erp (KNAW Humanities Cluster – Amsterdam, NL)
Elena Simperl (King's College London – London, GB)
Denny Vrandečić (Wikimedia Foundation – San Francisco, US)
Paul Groth (University of Amsterdam, NL)

Knowledge engineering has changed dramatically in the last twenty years. When the organisers of this seminar were starting out, it used to be about gathering highly curated knowledge from experts and encoding it into computational representations in knowledge bases. It was primarily a manual process, focusing more on how knowledge was structured and organised, for instance, as schemas or ontologies, and less on tying in existing data into that process. The results were used in expert systems and required considerable up-front investment. Today, knowledge base construction is a largely automatic process with human-in-the-loop. Owing to greater availability of data in different modalities and to advances in data management, machine learning, and crowdsourcing, knowledge bases today incorporate large amounts of knowledge. Provided access to data and (off-the-shelf) AI capabilities, an organisation can create a large knowledge base at a fraction of the costs from decades ago. It's for these reasons that we see knowledge bases, in particular in the form of knowledge graphs, routinely applied in anything from search and intelligent assistants to digital twins, supply chain management, and legal compliance. Many socio-technical challenges remain, which the seminar aimed to address with a mix of invited talks, deep-dives, and small-group workshops as following:

Landscape review: as the field has changed so much, both in research and practices, it was important to take inventory of approaches, methods, techniques, and tools by analysing real-world case studies where knowledge bases and knowledge graphs are created and used. Participants reflected on core lessons learned, knowledge gaps, and opportunities to create and maintain knowledge graphs at scale in various domains.

The knowledge graph life cycle: participants discussed extant knowledge engineering pipelines and identified gaps and connections between knowledge sources and methods and tools used in the construction and maintenance of knowledge graphs, including large language models and generative AI systems. There was consensus that we need a sustained effort to update and upgrade classical ontology engineering methodologies and develop a prototype infrastructure to make the most of the latest neurosymbolic technologies and tools. One specific challenge identified during the seminar was around taking knowledge engineering and knowledge graphs beyond structured data e.g., tables and information extraction from text to other modalities.

Using AI responsibly: as knowledge graph construction is slowly but surely embracing more and more sophisticated AI capabilities to scale, it is critical that processes and outcomes are aligned with fairness, accountability, and transparency guidance and standards. Solutions need to consider a range of end-users and stakeholders, including those that are unique to knowledge engineering settings such as domain experts, information scientists and librarians, and knowledge graph developers. Participants discussed the need for setting up task-based studies and in-depth analyses of human-centric challenges, and for developing bespoke explainability solutions and bias and fairness assessments.

Knowledge and technology transfer: knowledge graphs and knowledge engineering do not exist in isolation. From a research point of view, participants suggested activities to build capabilities to use the latest neurosymbolic technologies and tools in knowledge graph

construction, including tutorials, workshops, and hackathons, and to jointly develop frameworks and methodologies. From an application point of view, it was recognised that there is a need to promote knowledge graphs to the wider developer community and communicate their benefits, for instance, alongside neural methods.

2 Table of Contents

3 Overview of Talks

3.1 Day 1: History, Practices, Lessons Learned

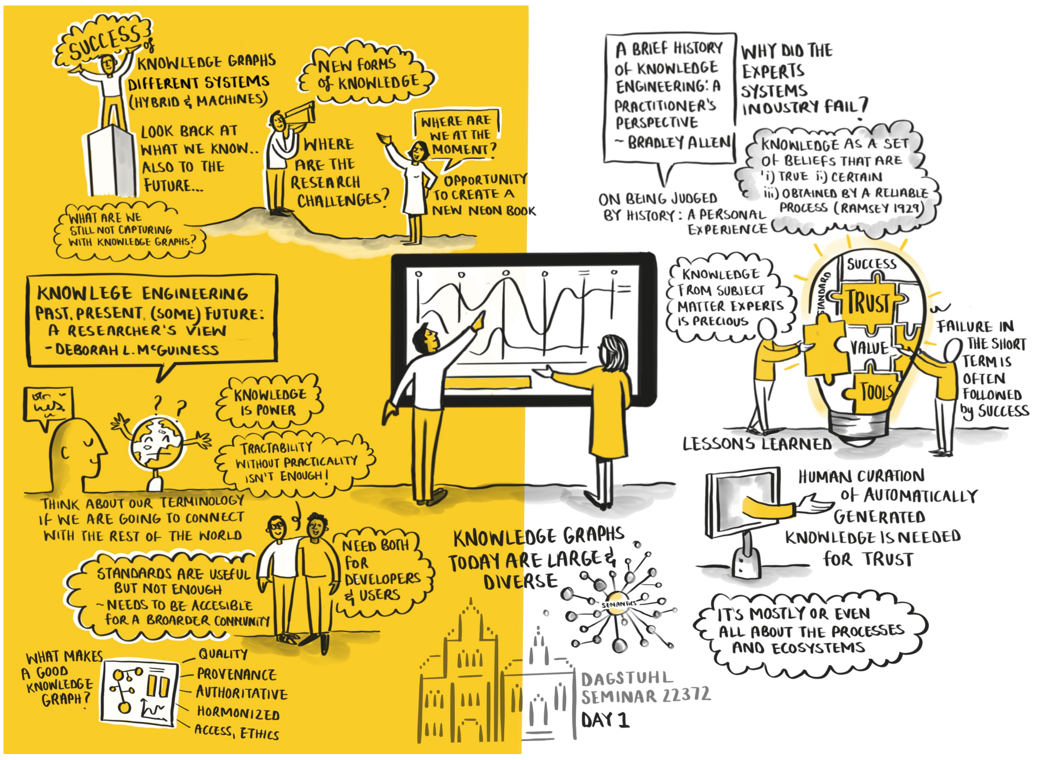
SUCCESS of KNOWLEDGE GRAPHS
DIFFERENT SYSTEMS (HYBRID & MACHINES)
LOOK BACK AT WHAT WE KNOW.. ALSO TO THE FUTURE...
NEW FORMS of KNOWLEDGE
WHERE ARE THE RESEARCH CHALLENGES?
WHERE ARE WE AT THE MOMENT?
OPPORTUNITY TO CREATE A NEW NEON BOOK
WHAT ARE WE STILL NOT CAPTURING WITH KNOWLEDGE GRAPHS?
KNOWLEGE ENGINEERING PAST, PRESENT, (SOME) FUTURE: A RESEARCHER'S VIEW - DEBORAH L. McGUINESS
KNOWLEDGE IS POWER
TRACTABILITY WITHOUT PRACTICALITY ISN'T ENOUGH!
THINK ABOUT OUR TERMINOLOGY IF WE ARE GOING TO CONNECT WITH THE REST OF THE WORLD
STANDARDS ARE USEFUL BUT NOT ENOUGH ~NEEDS TO BE ACCESIBLE FOR A BROARDER COMMUNITY
NEED BOTH FOR DEVELOPERS & USERS
WHAT MAKES A GOOD KNOWLEDGE GRAPH?
QUALITY
PROVENANCE
AUTHORITATIVE
HORMONIZED
ACCESS, ETHICS
KNOWLEDGE GRAPHS TODAY ARE LARGE & DIVERSE
SEMANTICS
DAGSTUHL SEMINAR 22372 DAY 1
A BRIEF HISTORY OF KNOWLEDGE ENGINEERING: A PRACTITIONER'S PERSPECTIVE ~ BRADLEY ALLEN
WHY DID THE EXPERTS SYSTEMS INDUSTRY FAIL?
KNOWLEDGE AS A SET OF BELIEFS THAT ARE i) TRUE ii) CERTAIN iii) OBTAINED BY A RELIABLE PROCESS (RAMSEY 1929)
ON BEING JUDGED BY HISTORY: A PERSONAL EXPERIENCE
KNOWLEDGE FROM SUBJECT MATTER EXPERTS IS PRECIOUS
SUCCESS
STANDARD
TRUST
VALUE
TOOLS
FAILURE IN THE SHORT TERM IS OFTEN FOLLOWED by SUCCESS
LESSONS LEARNED
HUMAN CURATION OF AUTOMATICALLY GENERATED KNOWLEDGE IS NEEDED FOR TRUST
IT'S MOSTLY OR EVEN ALL ABOUT THE PROCESSES AND ECOSYSTEMS

Figure 1 A History of Knowledge Engineering.

3.2 A Brief History of Knowledge Engineering: A Practitioner's Perspective

Bradley P. Allen (Merit International, Inc. – Millbrae, US, bradley.p.allen@gmail.com)

3.2.1 An Approach to the History of Knowledge Engineering

This talk is an attempt to outline the evolution of the discipline of knowledge engineering practice over time, draw some lessons from that evolution, and then raise a number of questions that this seminar is in a position to address, towards the end of defining paths forward for knowledge engineering with knowledge graphs in the 21st century.

Knowledge engineering as a discipline has changed considerably since its initial flowering during the period associated with expert systems development during the nineteen-eighties. If we define knowledge as a set of beliefs that are "(i) true, (ii) certain, [and] (iii) obtained by a reliable process" [2], we can further define knowledge engineering as the discipline of building and maintaining processes that produce knowledge. We argue that this gives us a framework to understand the history of knowledge engineering to date through the evolution of stated requirements for such knowledge production processes.

3.2.2 Seventy Years of Evolving Requirements

During the period from 1955 to today, we can identify four distinct periods, each of which began with the addition of a new set of requirements for knowledge production processes intended to address perceived shortcomings of systems developed during the preceding period (see fig. 2).

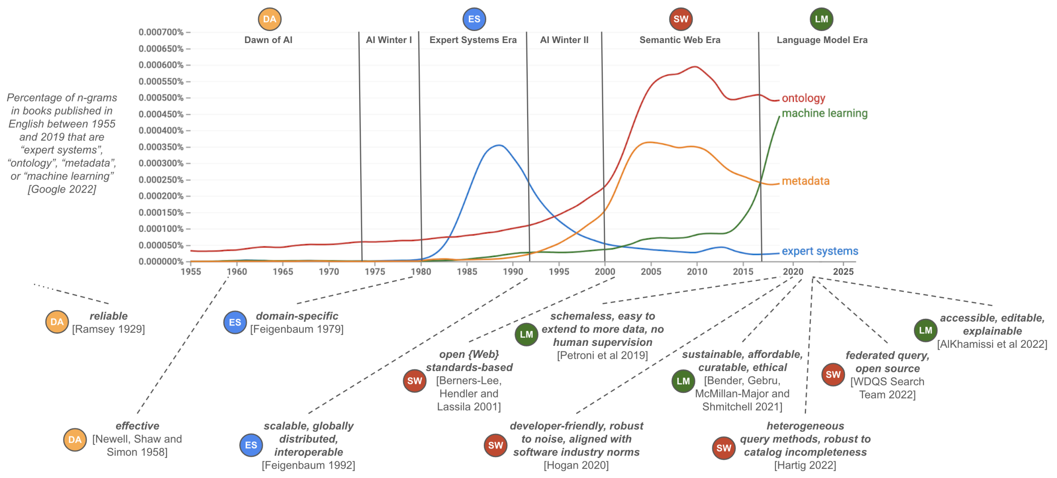

Figure 2 Seventy years of evolving requirements for knowledge production processes [1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13].

3.2.2.1 The Dawn of AI

Starting from Ramsey's simple requirement that such processes be reliable, some of the earliest work in AI identified the additional requirement that such processes also be effective, in the sense that they complete in a reasonable amount of time [3]. Newell and Simon were optimistic about the potential of goal-directed search using heuristics as a general approach to problem solving to be useful for practical applications, but by the beginning of the nineteen-seventies, it was clear that such systems were difficult to use in developing applications that were recognizably more than just toy tasks.

3.2.2.2 The Expert Systems Era

By the mid-seventies, having been deeply involved in attempting to apply Newell and Simon's model, Feigenbaum became convinced that automating knowledge production required a domain-specific focus to succeed [4]. His evangelism of knowledge engineering (a term he was instrumental in propagating the use of) engendered a period of intense activity in the construction of expert systems for the purposes of decision support in business enterprise settings. By the early nineteen-nineties, however, Feigenbaum and others acknowledged that the expert systems approach resulted in systems that were brittle and hard to maintain. Without abandoning his requirement that knowledge production be domain-specific in application focus and thus heavily dependent on subject matter expertise, he argued that future knowledge-based systems also be scalable, globally distributed, and interoperable to address these shortcomings [5]. At that point in time, however, there was no consensus about how such requirements could be addressed, but in retrospect, one can argue that in [5] Feigenbaum anticipated several aspects of what several years later would come to be known as the World Wide Web.

3.2.2.3 The Semantic Web Era

With the establishment of the Web and the emergence of Web architectural principles, Berners-Lee argued for a "Web of Data" based on linked data principles, standard ontologies, and data sharing protocols that not only provided an implementation of Feigenbaum's requirements, but with a single stroke established Web-centric open standards that anyone could adopt [6]. The subsequent twenty years witnessed the development of a globally federated open linked data "cloud", as well as the refinement of techniques for ontology engineering (i.e., the development and publishing of shared data schemas with semantics using linked data principles). Enterprises in particular found better value propositions for the use of such techniques toward the improvement of access and discovery of Web content and data, in contrast to the automation of decision making that was the primary value proposition for knowledge-based systems during the expert systems era [8]. However, while progress was made in building systems based on such principles, general adoption of specific principles advocated for by the semantic web community by the broader community of software developers and web application designers was slow, leading to semantic web researchers to identify additional requirements for broader adoption, for example that the core tools and standards used in semantic web application be more developer-friendly and more directly aligned with software industry norms, and that measures be taken to make federated open data more robust to noise [9]. Addition focus on improving the effectiveness of federated query, which proved hard to scale, and on handling the problem of data catalog incompleteness, all the while maintaining the practical benefits of open source and open standards led to new requirements towards those ends [11, 12].

3.2.2.4 The Language Model Era

The success of connectionist methods arising from the proliferation of graphical processing hardware for matrix arithmetic and concurrent innovations in neural network architectures [14] has led to a new set of possibilities for the production of knowledge graphs. This is an area that at the time of this writing is difficult to summarize due to the rapid rate of research publication, but two perspectives on the relation between language models and knowledge bases have emerged over the last several years. First, that the language model can serve directly as a knowledge base that is queryable using natural language prompts; secondly, that a language model can be a useful component in a knowledge production workflow that combines techniques based on the use of language models together with more traditional symbolic approaches [13]. Regardless of which of these perspectives is most valid, both are sure to result in work that will have an impact on the ability to produce and use knowledge graphs in knowledge engineering work moving forward.

3.2.3 Seventy Years of Lessons Learned

This decades-long evolution of knowledge engineering, bringing us to the current situation where the production of knowledge as knowledge graphs is gaining industrial acceptance at the same time as an entirely new paradigm of knowledge production through the use of large language models may be beginning to emerge, provides us with lessons learned along the way. In addition to these lessons from the history of knowledge engineering, it is also worth noting as well that this period also saw the evolution of software engineering best practices and patterns, as well as the emergence of both the software products and Internet services industries, and that many of the lessons learned in those contexts can be applied to improve the practice of knowledge engineering, particularly from a methodological perspective. Below we call out seven such lessons.

3.2.3.1 Manually Authored Knowledge from Subject Matter Experts is Precious

The digital library community has long argued that manually-created metadata is of vital importance in the creation of robust search resources, and much of the development of the World Wide Web (and continuing on to the Open Linked Data cloud) was informed by that assumption [15]. The effort of designing ontologies, taxonomies, and entity and relationship data has historically depended on expensive, labor-intensive manual effort. In many respects, the work generated by this labor is irreplaceable, and must be treated with respect. Acknowledging the essential nature of these foundational knowledge resources is not only important for an understanding of how knowledge is produced, but also to gain a clear understanding of the labor economics involved in these processes, from both a cost and an ethical perspective [16].

3.2.3.2 Automatic Generation of Knowledge is Needed for Scale

Automatic generation of knowledge graphs are needed to scale the extraction and production of knowledge. With the emergence of statistical natural language processing capable of dealing with training corpora on the order of half a trillion tokens, text available in massive curated corpora or the Web at large are now a effective source of manually authored knowledge. The sheer amount of human-authored content across the Web and in hand-crafted ontologies for linked open data require the automation of the knowledge graph creation process. Automation can also reduce time-to-market and enable larger and more up-to-date knowledge graphs to be generated, making knowledge graphs more accessible and useful.

3.2.3.3 Human Curation of Automatically Generated Knowledge is Needed for Trust

While automated systems can produce large knowledge graphs, they are limited in their ability to interpret and contextualize this output (though with the advent of language models this may be changing). Human curation is needed to verify that the knowledge graph production process is accurate. This process of verification is a necessary condition in many applications for users to be able to trust the knowledge and use it effectively. Additionally, human curation can provide insights into the data that automated systems may miss, such as potential ethical implications, biases, and areas for improvement.

3.2.3.4 User Buy-in to the Value Proposition is Essential

The failure of expert systems in delivering value to commercial enterprises can be viewed as an example of the failure of software product developers to understand users' needs and to effectively communicate value propositions to their users [17]. In striving to replace human decision makers, knowledge engineering in the expert systems era was attempting to solve a problem that ultimately turned out to be not of great importance to many enterprises. The Semantic Web era saw a realignment of knowledge engineering with user values by developing knowledge graphs that supported the needs of organizations to develop ways of guiding their users to the right sources of knowledge and information.

3.2.3.5 Developer Buy-in is Critical for Adoption of Standards and Tools

Software developer buy-in is critical for the successful adoption of standards and tools in any given field. Without their buy-in, the standards and tools will not be leveraged correctly by developers, or at all. We see this in the controversies around the adoption of Semantic Web standards and tools. In part, some developers are hesitant to use these standards and tools due to limited support by commercial vendors, and the lack of resources to help them understand the technology and how to incorporate it into their projects. Without the buy-in of software developers, knowledge graph standards and tools will continue to lack widespread adoption. In instances where commercially-useful enterprise knowledge graphs have been produced, such as Google Knowledge Graph [18], Amazon's Product Graph [19], and Wikidata [20], this has led to a reliance on custom architectures and approaches, which does not address the requirements of interoperability and federation of knowledge resources identified by Feigenbaum and Berners-Lee [5, 6].

3.2.3.6 Open Source/Access/Standards are a Huge Accelerant for Adoption

Open source/access/standards promote adoption because they make it easier to share, collaborate, and replicate research. For example, the pace of research and development in the area of large language models has been greatly accelerated by open source initiatives such as GPT-3 [21], TensorFlow [22], and PyTorch [23]. Initiatives such as these have provided researches in both academic and industrial contexts quick and easy access to cutting-edge tools and datasets, which in turn allows researchers to share, replicate, and collaborate on research quickly and easily through open access publication platforms such as Arxiv [24]. As a result, researchers are able to develop more sophisticated models and applications faster than ever before; this is in contrast with the experience of knowledge engineering in the expert systems era, which was heavily dependent on proprietary, closed source tools and technologies, and hence compromised with respect to the speed of innovation and technology transfer.

3.2.3.7 Failure in the Short Term is Often Followed by Success in the Long Term

It is easy to be disillusioned by the inability to deliver clear benefits out of the early adoption of technologies that initially seemed to carry significant promise. But often that perception of failure is due to insufficient time yet invested in working through the challenges of deployment and adoption. The history of speech recognition is a wonderful example of this. The initial approaches taken by participants in the ARPA Speech Understanding Research Project of the mid-nineteen-seventies laid the groundwork for much of what has come to be the statistical and neural language processing technologies approaching universal adoption today, at levels of accuracy barely dreamed of by the researchers of the time. At the conclusion of that effort, however, the evaluation of the project's result was decidedly mixed, with some expert evaluators arguing that the effort had in fact been a step backwards for the research area [25]. This example argues for patience in the effort to demonstrate the benefits of the use and application of knowledge graphs in knowledge engineering.

3.2.4 The Road Ahead: Questions for the Seminar

This seminar provides us with an opportunity to reflect on the past and come up with a set of goals for future progress towards the continuing evolution of knowledge engineering. Below are five questions that we believe need to be addressed to arrive at a robust set of goals.

In what ways does knowledge engineering deliver value today?
Knowledge graphs have demonstrated their ability to improve knowledge access, knowledge discovery, and heterogeneous data integration. But in many respects these are incremental improvements over what has been accomplished with software engineering in general. Can we identify economically and societally important problems either cannot be solved

without knowledge engineering, or are best solved with it? These problems will give us the basis for reinforcing the case for the benefits of knowledge engineering, that can be used to drive further adoption.

What should be the requirements for knowledge production processes?
Best practice in software product development requires us to clearly establish that our technology choices are properly motivated by our users' needs. What are we to carry forward from the cumulative sum of requirements articulated in the body of worked referenced in fig. 2 above? The requirements for knowledge graph production processes should include capabilities for data integration from multiple structured sources, data quality checks, entity resolution, merges and links, query optimization, and natural language processing. Moreover, the production processes should be automated to enable efficient updates and maintenance of the knowledge graph. Finally, the production process should incorporate mechanisms for security and privacy, as well as access control mechanisms to ensure that the data stays secure and only authorized users have access. It is worth observing that many of these issues have been explored to date in the more generic context of data engineering and data science architectures and platforms. To what extent does knowledge engineering add value to those architectures and platforms, and how current knowledge engineering and knowledge graphs tools and standards can be best integrated with them?

Why do we believe that knowledge graphs are a key enabling technology?
A fundamental premise of this seminar is there is a consensus that knowledge graphs are the preferred representation for knowledge for knowledge engineering. What evidence do we have for this assertion? Anecdotally, there is a better developer experience associated with the use of graphs as opposed to, e.g., rules, but what evidence has been gathered to support this view?

What other enabling technologies are there, and how do they interact with knowledge graphs?
Large language models show early promise as a enabling technology that can significant improve and complement knowledge graphs. Can they be harnessed to this end, or do they instead they present an alternative approach to knowledge engineering? In addition, graph databases are necessary for the storage and querying of knowledge graphs, but there is a bifurcation within the community between the use of RDF graphs and labeled property graphs to represent knowledge graphs. How can we reconcile these two approaches (for example, as described in [26])?

How can we convince people that knowledge graph engineering is mainstream software engineering?
Finally, and perhaps most importantly, the majority of software engineering efforts today do not involve the use of knowledge engineering techniques, even in use cases where knowledge engineers can see clear benefits to be gained in their use. Knowledge engineering is still a niche skill set that is unfamiliar to most practicing software engineers. However, the architectures and methodologies emerging from the commercial applications of machine learning, data science, and data engineering [27] in many ways borrow heavily from those developed to support knowledge engineering. How can we better relate knowledge engineering concepts, tools and methodologies to the industry consensus and ecosystem that has been established for data engineering and data science platforms, and drive mainstream adoption in the future?

References

1 Google Books Ngram Viewer. "percentage of n-grams in books published in english between 1955 and 2019 that are "expert systems", "ontology", "metadata", or "machine learning"". `https://books.google.com/ngrams/graph?content=expert+systems%2Contology%2Cmachine+learning%2Cmetadata&year_start=1955&year_end=2019&corpus=26&smoothing=1`, 2022. Accessed: 2022-08-10.

2 F.P. Ramsey. Knowledge. In *F.P. Ramsey: Philosophical Papers*, pages 110–111. Cambridge University Press, 1929.

3 Allen Newell, John Calman Shaw, and Herbert A Simon. Elements of a theory of human problem solving. *Psychological review*, 65(3):151, 1958.

4 Edward A Feigenbaum. The art of artificial intelligence: Themes and case studies of knowledge engineering. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, volume 2. Boston, 1977.

5 Edward A Feigenbaum. *A personal view of expert systems: Looking back and looking ahead.* Knowledge Systems Laboratory, Department of Computer Science, Stanford ..., 1992.

6 Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific american*, 284(5):34–43, 2001.

7 Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

8 James Hendler, Fabien Gandon, and Dean Allemang. *Semantic web for the working ontologist: Effective modeling for linked data, RDFS, and OWL.* Morgan & Claypool, 2020.

9 Aidan Hogan. The semantic web: Two decades on. *Semantic Web*, 11(1):169–185, 2020.

10 Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big?. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623, 2021.

11 Olaf Hartig. "reflections on linked data querying and other related topics". `https://olafhartig.de/slides/Slides-DKG-SWSA-Talk.pdf`, 2022. Accessed: 2022-03-17.

12 WDQS Search Team. "wdqs backend alternatives: The process, details and results". `https://www.wikidata.org/wiki/File:WDQS_Backend_Alternatives_working_paper.pdf`, 2022. Accessed: 2022-08-15.

13 Badr AlKhamissi, Millicent Li, Asli Celikyilmaz, Mona Diab, and Marjan Ghazvininejad. A review on language models as knowledge bases. *arXiv preprint arXiv:2204.06031*, 2022.

14 Haohan Wang and Bhiksha Raj. On the origin of deep learning. *arXiv preprint arXiv:1702.07800*, 2017.

15 Jane Greenberg. Big metadata, smart metadata, and metadata capital: toward greater synergy between data science and metadata. *Journal of Data and Information Science*, 2(3):19–36, 2017.

16 Amandalynne Paullada, Inioluwa Deborah Raji, Emily M Bender, Emily Denton, and Alex Hanna. Data and its (dis) contents: A survey of dataset development and use in machine learning research. *Patterns*, 2(11):100336, 2021.

17 Adrian Payne, Pennie Frow, and Andreas Eggert. The customer value proposition: evolution, development, and application in marketing. *Journal of the Academy of Marketing Science*, 45(4):467–489, 2017.

18 Natasha Noy, Yuqing Gao, Anshu Jain, Anant Narayanan, Alan Patterson, and Jamie Taylor. Industry-scale knowledge graphs: Lessons and challenges: Five diverse technology companies show how it's done. *Queue*, 17(2):48–75, 2019.

19 Xin Luna Dong. Challenges and innovations in building a product knowledge graph. In *Proceedings of the 24th ACM SIGKDD International conference on knowledge discovery & data mining*, pages 2869–2869, 2018.

20 Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

21 Robert Dale. Gpt-3: What's it good for? *Natural Language Engineering*, 27(1):113–118, 2021.

22 Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. {TensorFlow}: a system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

23 Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

24 Paul Ginsparg. Lessons from arxiv's 30 years of information sharing. *Nature Reviews Physics*, 3(9):602–603, 2021.

25 Wayne A Lea and June E Shoup. Review of the arpa sur project and survey of current technology in speech understanding. Technical report, Office of Naval Research, January 1979. Final Report.

26 Renzo Angles, Harsh Thakkar, and Dominik Tomaszuk. Mapping rdf databases to property graph databases. *IEEE Access*, 8:86091–86110, 2020.

27 Matt Bornstein, Jennifer Li, and Casado. Martin. Emerging architectures for modern data infrastructure. `https://future.com/emerging-architectures-modern-data-infrastructure/`, 2020. Accessed: 2022-12-02.

3.3 Knowledge Engineering Past, Present, (some) Future: A Researcher's View

Deborah L. McGuinnness (Rensselaer Polytechnic Institute – Troy, US, dlm@cs.rpi.edu)

I provide a brief historical perspective on significant research contributions and highlight some key lessons, some of which may be particularly worthy for reflection as we move forward. I begin with expert systems as a foundational motivating area but I also highlight the evolution and contributions from the structured object and ontology communities. I also reflect on the early notion of knowledge engineering as the applied side of artificial intelligence (from Feigenbaum) and present that notion in the grounding of the 21st century environment. I also present a range of characteristics as considerations for evaluating if a knowledge engineered system is "good".

I then present some perspectives on the our current landscape that may be significantly different from the past. These include: much greater need for knowledge graph interoperability (as well, of course, as the needs for compatibility and interoperability with a wide range of ontologies); The very large linked open data world ; the significantly more diverse architectures for hybrid AI systems, with large language models as an increasing component; the increasingly diverse community of co-designers and co-authors of large "smart" data portals. I conclude with a set of driving research questions along with a take home message that process and methodology is becoming even more critical for our field to increase impact and buy-in.

3.4 Automated Knowledge Graph Construction

Lise Stork (Vrije Universiteit of Amsterda, NL, l.stork@vu.nl)

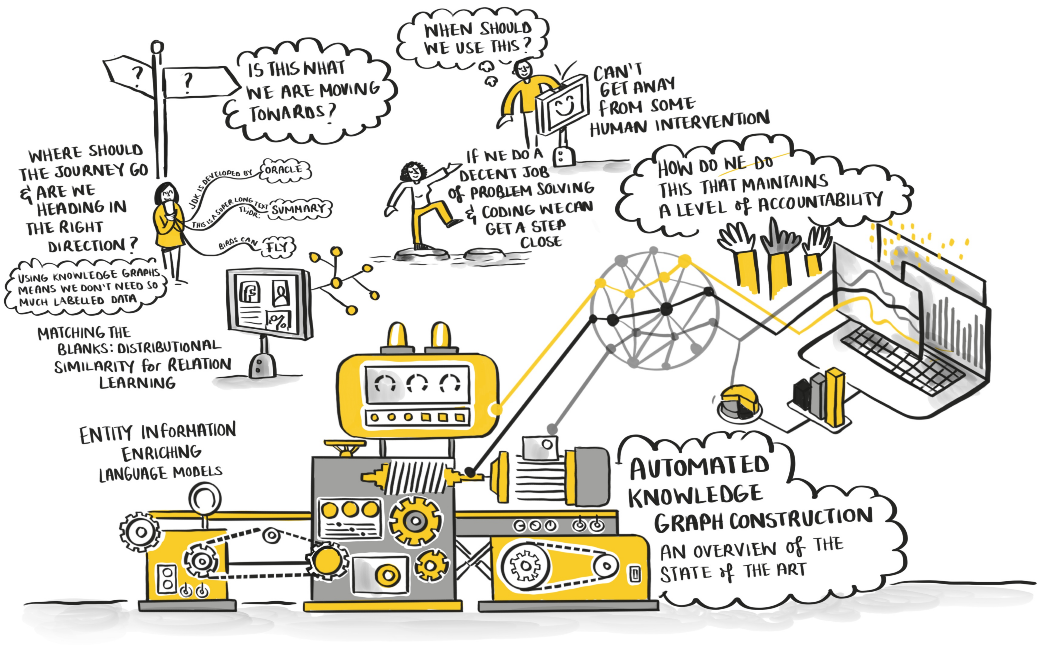
IS THIS WHAT WE ARE MOVING TOWARDS?
WHEN SHOULD WE USE THIS?
CAN'T GET AWAY FROM SOME HUMAN INTERVENTION
WHERE SHOULD THE JOURNEY GO & ARE WE HEADING IN THE RIGHT DIRECTION?
JDK IS DEVELOPED BY ORACLE
THIS IS A SUPER LONG TEXT TL;DR: SUMMARY
BIRDS CAN FLY
IF WE DO A DECENT JOB of PROBLEM SOLVING & CODING WE CAN GET A STEP CLOSE
HOW DO WE DO THIS THAT MAINTAINS A LEVEL of ACCOUNTABILITY
USING KNOWLEDGE GRAPHS MEANS WE DON'T NEED SO MUCH LABELLED DATA
MATCHING THE BLANKS: DISTRIBUTIONAL SIMILARITY for RELATION LEARNING
ENTITY INFORMATION ENRICHING LANGUAGE MODELS
AUTOMATED KNOWLEDGE GRAPH CONSTRUCTION
AN OVERVIEW of THE STATE of THE ART

Figure 3 Automated Knowledge Graph Construction.

The talk gave an overview of Knowledge Graph Construction (KGC) methods, four big focus shifts in the development of these methods [1], and sketched some open challenges and future directions for KGC.

Over the past decade, many methods have been proposed for KGC: human-based collaborative or curated approaches in which experts work together to create and curate knowledge graphs, but also automated approaches, classified broadly into approaches that use a predefined schema for extraction, versus open information extraction (IE) [2, 3]. Tasks become increasingly harder (i) with less data available for training, (ii) when relationships are increasingly complicated to extract (binary vs n-ary relations) and (iii) the openness of the task: schema-driven vs open IE.

Methods proposed for KGC have shifted focus from the engineering of features, to the engineering of model architectures, the engineering of tasks or objectives, to the engineering of prompts [1]. Before 2013, domain experts used their expertise about a domain to define salient textual features to be used in an NLP task. After the rise of Neural Nets, the focus shifted towards Architecture Engineering: convolutional neural networks as well as recurrent neural networks such as LSTMs and BiLSTMs were applied in a fully supervised manner, where features were learned jointly with the supervised classification task.

Around 2017, the power of language models increased, mostly due to the discovery that proximity in the input is less important than expected, and context is much better represented when sentences are processed fully, using attention mechanisms [4], instead of sequentially. Such a method at the same time proved easier to train. Since these models, when trained on large corpora, appeared powerful enough to be used in a variety of down-stream tasks, the focus then began to shift towards the fine-tuning of pre-trained LLMs specific tasks [5, 6, 7, 8, 9].

Lastly, since 2019, it was found that these LLMs are interesting to probe, given that they have learned a lot of interesting facts. It was hypothesized that LLMs could serve as knowledge graphs themselves; new ways had to be discovered to query them. Therefore, the focus shifted towards creating, either manually or algorithmically, prompts in order to get out the interesting facts these LLMs models had learned, and 'prompt engineering' became an active field of research [1, 10, 11].

Open challenges that were proposed:

1. how to automatically construct "higher-order or higher-ary knowledge", such as scopes, context, degrees of belief, confidence, and how to evaluate these;
2. how to deal with n to M relations;
3. how do we integrate LMs in the knowledge engineering pipeline;
4. how to deal with bias, trust and control in LMs as KGs; how to add provenance to statements in LMs;
5. how to deal with explainability of answers from prompts;
6. how to update facts in LLMs as KGs;
7. what types of knowledge representations do we extract.

References

1 Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *CoRR*, abs/2107.13586, 2021.

2 Gerhard Weikum, Xin Luna Dong, Simon Razniewski, and Fabian Suchanek. Machine knowledge: Creation and curation of comprehensive knowledge bases. *Foundations and Trends in Databases*, 10(2-4):108–490, 2021.

3 Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. Knowledge graphs. *ACM Computing Surveys (CSUR)*, 54(4):1–37, 2021.

4 Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

5 Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

6 Shanchan Wu and Yifan He. Enriching pre-trained language model with entity information for relation classification. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 2361–2364, 2019.

7 Christoph Alt, Marc Hübner, and Leonhard Hennig. Fine-tuning pre-trained transformer language models to distantly supervised relation extraction. *arXiv preprint arXiv:1906.08646*, 2019.

8 Xu Han, Tianyu Gao, Yankai Lin, Hao Peng, Yaoliang Yang, Chaojun Xiao, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. More data, more relations, more context and more openness: A review and outlook for relation extraction. *arXiv preprint arXiv:2004.03186*, 2020.

9 Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. Matching the blanks: Distributional similarity for relation learning. *arXiv preprint arXiv:1906.03158*, 2019.

10 Dimitrios Alivanistos, Selene Báez Santamaría, Michael Cochez, Jan-Christoph Kalo, Emile van Krieken, and Thiviyan Thanapalasingam. Prompting as probing: Using language models for knowledge base construction. *arXiv preprint arXiv:2208.11057*, 2022.

11 Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

3.5 Day 2 Challenges and Future Directions

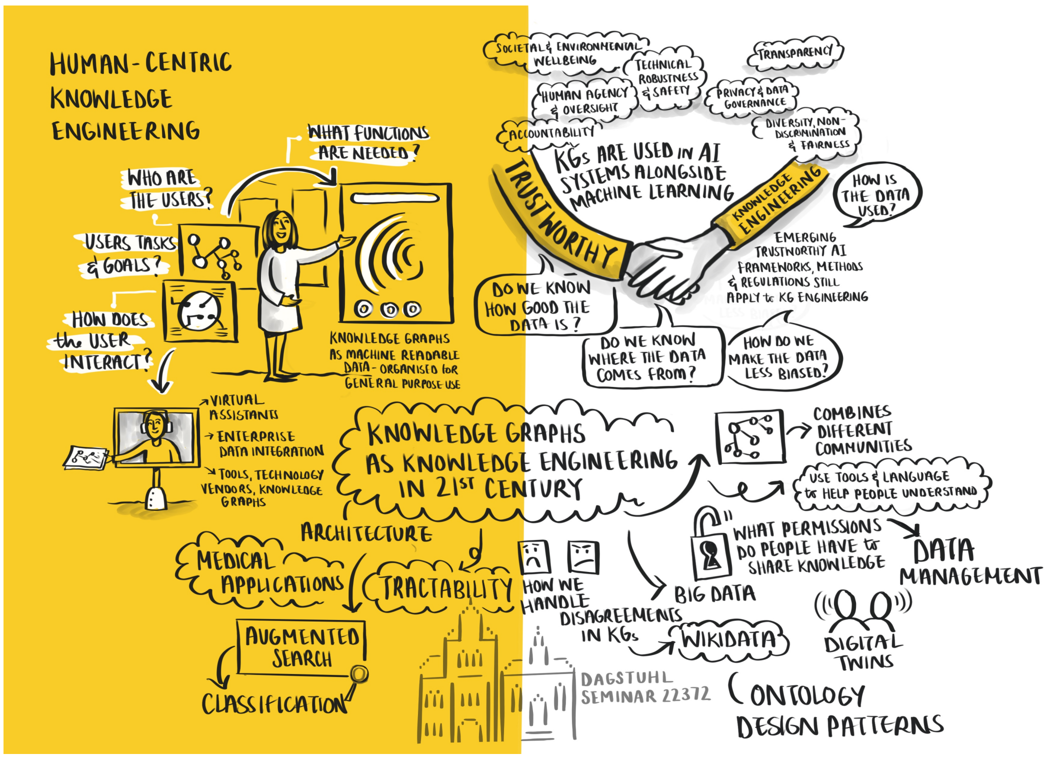
HUMAN-CENTRIC KNOWLEDGE ENGINEERING
WHAT FUNCTIONS ARE NEEDED?
WHO ARE THE USERS?
USERS TASKS & GOALS?
HOW DOES the USER INTERACT?
KNOWLEDGE GRAPHS AS MACHINE READABLE DATA - ORGANISED for GENERAL PURPOSE USE
VIRTUAL ASSISTANTS
ENTERPRISE DATA INTEGRATION
TOOLS, TECHNOLOGY VENDORS, KNOWLEDGE GRAPHS
KNOWLEDGE GRAPHS AS KNOWLEDGE ENGINEERING IN 21ST CENTURY
ARCHITECTURE
MEDICAL APPLICATIONS
TRACTABILITY
AUGMENTED SEARCH
CLASSIFICATION
SOCIETAL & ENVIRONMENTAL WELLBEING
TECHNICAL ROBUSTNESS & SAFETY
TRANSPARENCY
HUMAN AGENCY & OVERSIGHT
PRIVACY & DATA GOVERNANCE
ACCOUNTABILITY
DIVERSITY, NON-DISCRIMINATION & FAIRNESS
KGs ARE USED IN AI SYSTEMS ALONGSIDE MACHINE LEARNING
TRUSTWORTHY
KNOWLEDGE ENGINEERING
HOW IS THE DATA USED?
EMERGING TRUSTWORTHY AI FRAMEWORKS, METHODS & REGULATIONS STILL APPLY to KG ENGINEERING
DO WE KNOW HOW GOOD THE DATA IS?
DO WE KNOW WHERE THE DATA COMES FROM?
HOW DO WE MAKE THE DATA LESS BIASED?
COMBINES DIFFERENT COMMUNITIES
USE TOOLS & LANGUAGE to HELP PEOPLE UNDERSTAND
WHAT PERMISSIONS DO PEOPLE HAVE to SHARE KNOWLEDGE
DATA MANAGEMENT
HOW WE HANDLE DISAGREEMENTS IN KGs
BIG DATA
WIKIDATA
DIGITAL TWINS
DAGSTUHL SEMINAR 22372
ONTOLOGY DESIGN PATTERNS

Figure 4 Future Directions: Human-Centric Knowledge Graph Construction.

3.6 Human-Centric Knowledge Engineering: Making Knowledge Engineering Trustworthy

Elena Simperl (King's College London, GB, elena.simperl@kcl.ac.uk)

Knowledge engineering has changed dramatically since I was doing my PhD. It was always meant to be and has remained a challenging, inter-disciplinary subject – the question of how to encode a domain in a computational representation will always be non-trivial and will require some form of human-in-the-loop. The field has advanced considerably: the knowledge bases we build today are much larger and more complex than twenty years ago, there are a range of technologies and end-user tools to support standard tasks, as well as several notable open-source projects delivering large knowledge graphs that can bootstrap applications without massive up-front investment. And yet, our understanding of user-centric aspects of knowledge engineering remains limited. The reasons for this are as often sociotechnical, but the result is clear: we are not (yet) in a position to fully answer questions like these:

- Who are the users?
- What are the users' tasks and goals?
- How does a user interact with the knowledge graph?
- What are the users' experience levels with it, or with similar environments?
- What functionalities does the user need?
- What additional information might the users need, and in what form do they need it?
- How does the user think knowledge engineering tools should work?
- Is the user multitasking? Are they working on a mobile phone, desktop computer etc?
- Does the interface utilise different input modes, such as touch, speech, gestures or orientation?
- How can we support multi-disciplinary teams?
- How can we support remote work, decision making, conflict resolution?

Answering such questions will require studies of specific knowledge engineering projects or tool environments, but would deliver invaluable insights to improve both user experience and knowledge graph outcomes. In time, it would lead to a culture of user-centric design and to research advances that are applicable beyond knowledge engineering contexts. with the recent changes, it is also worth revisiting the surveys and handbooks written a decade or so ago to deliver up-to-date comparative surveys and tool evaluations, relevant to how knowledge graphs are built today in terms of scale, complexity, and degree of automation.

Using automation, particularly the latest AI capabilities, raises interesting human-centric challenges, which other communities such as natural language processing and computer vision are starting to explore. These are grouped under the banner of trustworthy AI, which is concerned with questions of transparency, accountability, fairness, human agency and oversight, and sustainability when AI is used by (or impacts) different groups of people. There is a large body of work happening right now to define frameworks, guidance, regulation, and standards for trustworthy AI[1] – for instance, the European Commission has proposed seven dimensions for designing AI systems, shown in Figure 5 and there are many standardisation activities at national and international levels (e.g. ISO).[2]

[1] For an overview, see e.g., OECD AI Policy Observatory, `https://oecd.ai/`, visited in September 2022
[2] See a list of AI-related ISO standards at `https://www.iso.org/committee/6794475/x/catalogue/`, visited in December 2022

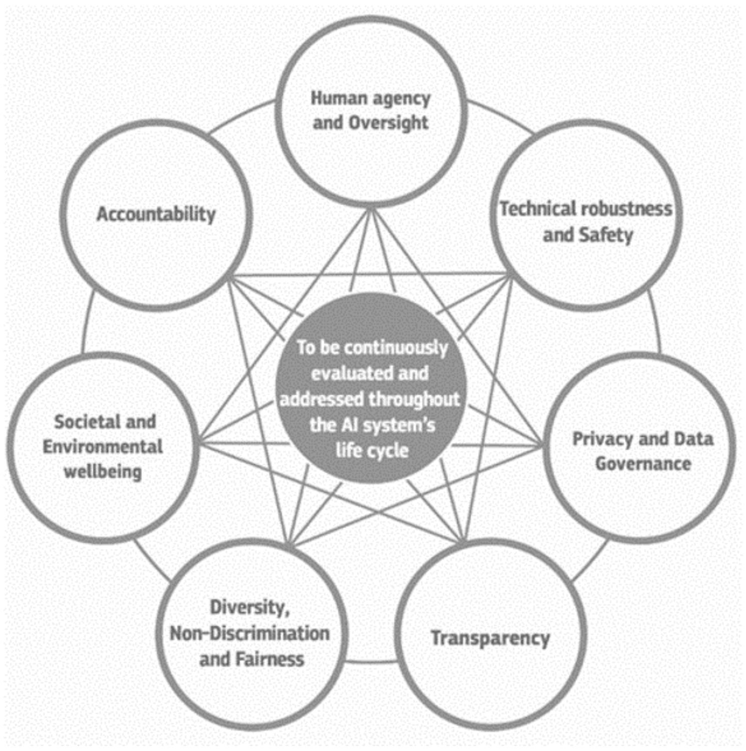
Human agency and Oversight
Technical robustness and Safety
Privacy and Data Governance
Transparency
Diversity, Non-Discrimination and Fairness
Societal and Environmental wellbeing
Accountability
To be continuously evaluated and addressed throughout the AI system's life cycle

Figure 5 Dimensions of Trustworthy AI according to the European Commission.

Ongoing research in trustworthy AI proposed a core set of methods and best practices to meet the regulatory requirements of trustworthy AI systems [1]. These include factsheets [2], model cards [3], canvases,[3] explainability methods [4] and fairness and debiasing methods [5]. Knowledge graphs and knowledge graph construction systems need to build on these works to ensure the processes we follow and their outcomes can be trusted by end-users and stakeholders. This includes: the domain expert or business analyst involved in knowledge acquisition, the knowledge engineer building the knowledge graph construction pipeline, the crowd worker labelling training data, the developer of downstream applications using the knowledge graph, for instance in the form of embeddings [6] and the users of those applications.

In my team we undertook research into knowledge communities such as DBpedia and Wikidata to understand how different components of trust emerge and propose socio-technical methods to improve the quality of the knowledge graph and make it more complete, up-to-date and less biased. The research pursued questions such as:

Do we know how good the data in the knowledge graph is? In [7] we surveyed 28 quality approaches and methods for Wikidata and proposed a joint framework.

Do we know where the data comes from In [8] and [9] we proposed an AI architecture with human-in-the-loop to assess quality of triple provenance across five languages.

Do we know how to audit our data to make it less biased? In [10] we proposed a method to detect content gaps in open knowledge graphs and applied it to three main types of biases: gender, recency, and socioeconomic biases.

[3] For example, the data ethics canvas of the Open Data Institute in the UK, `https://theodi.github.io/interactive-data-ethics-canvas/`, visited December 2022

Do we know how the data came about? Even when automation is at play, knowledge construction is a social process. For this reason, we analysed the link between quality of knowledge graph entities and the make-up of the editor teams that worked on those entities [11, 12]

Do we know how the data is used? One application of knowledge graphs is natural language processing. In [13] we evaluated a natural language generation system that takes Wikidata triples and creates Wikipedia articles in different languages. We ran user studies to understand if and when the presence of automation changes editor perceptions and practices.

Knowledge engineering remains as exciting of a field as ever, with a range of human-centric challenges that cannot and should not be overlooked given the advanced in the field and in related fields such as machine learning, natural language processing, and computer vision. Looking ahead, I would like to see more work into establishing user-centric design and empirical methods more firmly into the ways we build our tools and applications. In particular, we need to ensure the way we make knowledge graphs today is interpretable and trustworthy, and ongoing research in the area of responsible AI, including transparency, accountability, and fairness can deliver new impulses for interdisciplinary research.

References

1 Alon Jacovi, Ana Marasović, Tim Miller, and Yoav Goldberg. Formalizing trust in artificial intelligence: Prerequisites, causes and goals of human trust in ai. In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 624–635, 2021.

2 Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé Iii, and Kate Crawford. Datasheets for datasets. *Communications of the ACM*, 64(12):86–92, 2021.

3 Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. Model cards for model reporting. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 220–229, 2019.

4 Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller. *Explainable AI: interpreting, explaining and visualizing deep learning*, volume 11700. Springer Nature, 2019.

5 Alexandra Chouldechova and Aaron Roth. A snapshot of the frontiers of fairness in machine learning. *Communications of the ACM*, 63(5):82–89, 2020.

6 Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743, 2017.

7 Alessandro Piscopo and Elena Simperl. What we talk about when we talk about wikidata quality: a literature survey. In *Proceedings of the 15th International Symposium on Open Collaboration*, pages 1–11, 2019.

8 Alessandro Piscopo, Lucie-Aimée Kaffee, Chris Phethean, and Elena Simperl. Provenance information in a collaborative knowledge graph: an evaluation of wikidata external references. In *International semantic web conference*, pages 542–558. Springer, 2017.

9 Gabriel Amaral, Alessandro Piscopo, Lucie-Aimée Kaffee, Odinaldo Rodrigues, and Elena Simperl. Assessing the quality of sources in wikidata across languages: a hybrid approach. *Journal of Data and Information Quality (JDIQ)*, 13(4):1–35, 2021.

10 David Abián, Albert Meroño-Peñuela, and Elena Simperl. An analysis of content gaps versus user needs in the wikidata knowledge graph. In *International Semantic Web Conference*, pages 354–374. Springer, 2022.

11 Alessandro Piscopo, Chris Phethean, and Elena Simperl. What makes a good collaborative knowledge graph: group composition and quality in wikidata. In *International Conference on Social Informatics*, pages 305–322. Springer, 2017.
12 Alessandro Piscopo and Elena Simperl. Who models the world? collaborative ontology creation and user roles in wikidata. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW):1–18, 2018.
13 Lucie-Aimée Kaffee, Pavlos Vougiouklis, and Elena Simperl. Using natural language generation to bootstrap missing wikipedia articles: A human-centric perspective. *Semantic Web*, 13(2):163–194, 2022.

3.7 Everything is Expensive

Denny Vrandečić (Wikimedia Foundation – San Francisco, US, denny@wikimedia.org)

The creation of ontologies is expensive. It is hard to achieve the initial buy-in from developers, and often developers are mandated into the use of a specific ontology. This often leads to less than enthusiastic support. The ontologist often has to ”cat-herd” technical leads and product managers across several departmenets and organizations. Generating agreements often takes a long time and many meetings with discussions and not directly tangible outcomes. And meetings are expensive. Even in Wikidata, property creation is one of the major bottlenecks.

The trouble with triples. Single triples cannot express complex statements (known as n-ary statement, but also not frames or events). So patterns of triples are required to represent such complex statements. But for users of a triple store, these are atomic statements. Tools, user experience, metrics, processes all become much more complex and expensive due to this mismatch.

Will large language models lead to cheap ontologies? It is expected that a technology-driven companies there will be an initial surplus of trust in large language models, which may backfire when these models lead to expensive errors. On the other side, technology-skeptical organizations such as in journalism or in Wikidata, may start with a deficit of trust, which may hamper the usage of these technologies. The biggest problem is actually the same as with handmade ontologies: how to make people understand, commit to, and trust the created ontologies? The cost is not in creating the ontologies, but the agreement.

Knowledge acquisition is expensive. Once we have the ontology, how do we efficiently populate it? How do we let humans efficiently check a large amount of data before product launch? Important are the possibility to sample parts of the knowledge graph, which are either particularly impactful or particularly interesting. Rules have been very good at discovering inconsistencies and incompleteness. Machine learning has also been well used to suggest anomalies.

Knowledge maintenance is expensive. Now that we have large lists of inconsistencies and incomplete data, what do we do with that? We also need to keep and maintain metadata about exceptions (because the world is always more complex than your rules). If we allow for feedback from end users, how do we capture and classify that feedback? If we don't allow for feedback, what is the point of the knowledge graph? How do we channel feedback in order to maintain the knowledge?

3.8 Tools and User Experience for KG Engineering

Filip Ilievski (University of Southern California – Marina del Rey, US, ilievski@isi.edu)

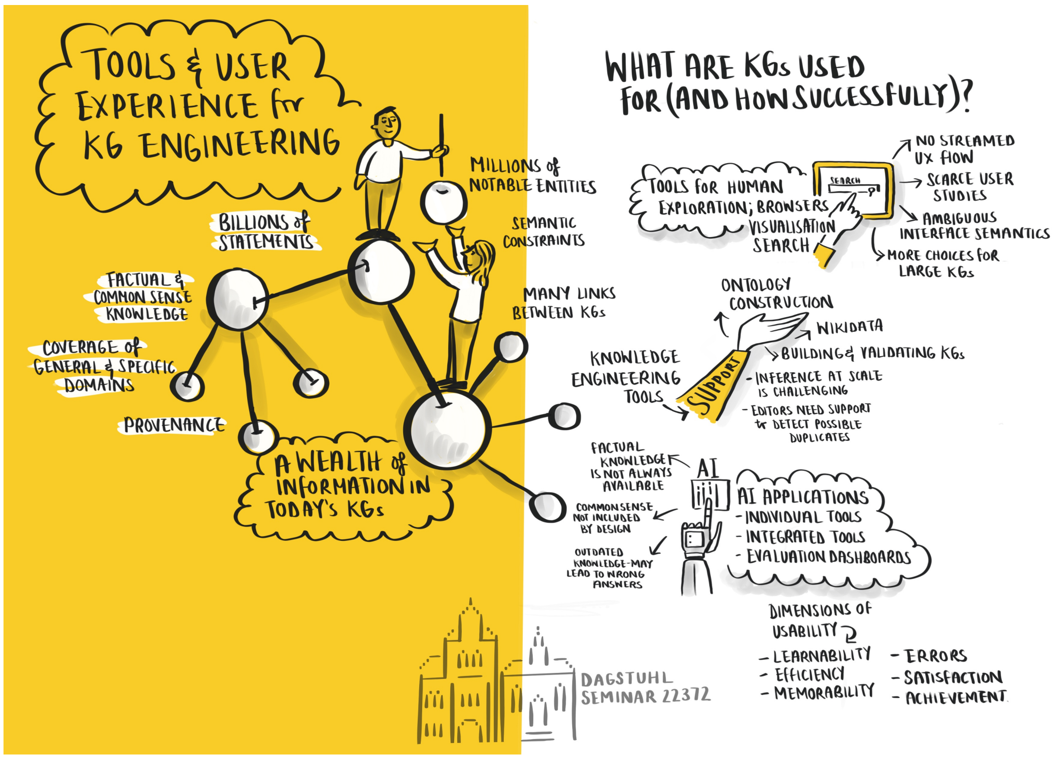
TOOLS & USER EXPERIENCE for KG ENGINEERING
BILLIONS of STATEMENTS
MILLIONS of NOTABLE ENTITIES
SEMANTIC CONSTRAINTS
FACTUAL & COMMON SENSE KNOWLEDGE
COVERAGE of GENERAL & SPECIFIC DOMAINS
MANY LINKS BETWEEN KGs
PROVENANCE
A WEALTH of INFORMATION IN TODAY'S KGs
WHAT ARE KGs USED FOR (AND HOW SUCCESSFULLY)?
TOOLS FOR HUMAN EXPLORATION; BROWSERS VISUALISATION SEARCH
SEARCH
NO STREAMED UX FLOW
SCARCE USER STUDIES
AMBIGUOUS INTERFACE SEMANTICS
MORE CHOICES FOR LARGE KGs
ONTOLOGY CONSTRUCTION
WIKIDATA
BUILDING & VALIDATING KGs
KNOWLEDGE ENGINEERING TOOLS
SUPPORT
- INFERENCE AT SCALE IS CHALLENGING
- EDITORS NEED SUPPORT to DETECT POSSIBLE DUPLICATES
FACTUAL KNOWLEDGE IS NOT ALWAYS AVAILABLE
AI
COMMONSENSE NOT INCLUDED BY DESIGN
OUTDATED KNOWLEDGE-MAY LEAD TO WRONG ANSWERS
AI APPLICATIONS
- INDIVIDUAL TOOLS
- INTEGRATED TOOLS
- EVALUATION DASHBOARDS
DIMENSIONS OF USABILITY
- LEARNABILITY
- EFFICIENCY
- MEMORABILITY
- ERRORS
- SATISFACTION
- ACHIEVEMENT
DAGSTUHL SEMINAR 22372

Figure 6 Tools and User Experience.

Today's knowledge graphs contain a wealth of information. For instance, Wikidata has billions of statements for millions of notable entities, with recorded provenance and semantic constraints. In this talk, I ask the question: What are KGs used for and how successfully? I consider three user profiles: an end user, an AI/DS engineer, and a knowledge engineer.
1) An end user might use knowledge graphs to explore knowledge, browse answers to their questions, or develop new ideas. These tasks can be supported by browsers, visualization tools, or tools for textual and faceted search. Key pain pointers from an end-user perspective are the lack of streamed workflow from high- to micro-level, the lack of user studies, the ambiguity of interface semantics, and issues with compositionality and data quality.
2) An AI/DS engineer might use knowledge technologies to perform automatic question answering, recommendation, search, or content enrichment. These tasks can be pursued with a pipeline of existing tools that perform operations like entity linking, semantic similarity, lexicalization, and embedding learning. Integrated tools, databases, or libraries allow developers to perform a set of these operations in the same framework, avoiding the need to compose different toolkits, formats, and standards themselves. Evaluation dashboards for tasks like knowledge graph completion and question answering enable fine-grained auditing of system performance. Pain points for AI engineers include: sparsity of factual and commonsense knowledge, consistency of ontological knowledge, the lack of decision support tooling, and potentially outdated knowledge.

3) A knowledge engineer might contribute or edit new knowledge or provenance, perform semantic modeling and validation, infer new knowledge, or engineer a new knowledge graph. Key tools for knowledge engineers include ontology editors, tooling for Wikidata contributors, and knowledge construction and validation tools. The key pain points for knowledge engineers are that inference at scale is challenging, identity is hard to establish, different is-a flavors are difficult to distinguish, the lack of tool integration, and the lack of user studies and logging practices.

3.9 Social and Technical Biases in Knowledge Graphs

Harald Sack (FIZ Karlsruhe – Leibniz Institute for Information Infrastructure, DE & Karlsruhe Institute of Technology (KIT), DE, harald.sack@fiz-karlsruhe.de)

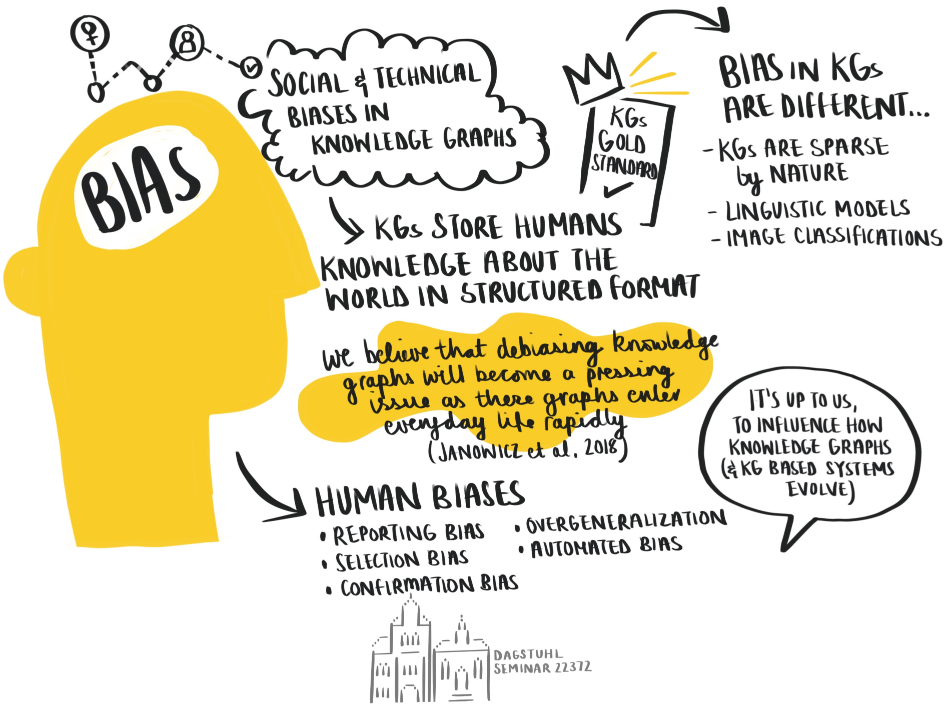
BIAS
SOCIAL & TECHNICAL BIASES IN KNOWLEDGE GRAPHS
KGs GOLD STANDARD
BIAS IN KGs ARE DIFFERENT...
- KGs ARE SPARSE by NATURE
- LINGUISTIC MODELS
- IMAGE CLASSIFICATIONS
KGs STORE HUMANS KNOWLEDGE ABOUT THE WORLD IN STRUCTURED FORMAT
We believe that debiasing knowledge graphs will become a pressing issue as these graphs enter everyday life rapidly (JANOWICZ et al. 2018)
IT'S UP TO US, TO INFLUENCE HOW KNOWLEDGE GRAPHS (& KG BASED SYSTEMS EVOLVE)
HUMAN BIASES
• REPORTING BIAS
• SELECTION BIAS
• CONFIRMATION BIAS
• OVERGENERALIZATION
• AUTOMATED BIAS
DAGSTUHL SEMINAR 22372

Figure 7 Biases in Knowledge Graphs.

Knowledge graphs as a key tool for organizing and presenting information in the modern world constitute networks of interlinked data that help us to make sense of the vast amounts of information available to us. Once constructed, knowledge graphs are often considered as "gold standard" data sources that safeguard the correctness of other systems. Thereby, objectivity and neutrality of the represented information have become an important issue. Biases inherent to knowledge graphs may become magnified and spread through knowledge graph based systems. Traditionally, bias can be defined as "*a disproportionate weight in favor* *of or against an idea or thing, usually in a way that is closed-minded, prejudicial, or unfair*"[4]. Taking into account the bias networking effect for knowledge graphs, it is crucial that we acknowledge and address various types of bias already in knowledge graph construction [1].

Biases in knowledge graphs as well as potential means to address them, are different from those in linguistic models or image classification. Knowledge graphs are sparse by nature, i.e. only a small number of triples are available per entity. In difference, linguistic models learn the meaning of a term from its context within large corpora and image classification learns classes from millions of labeled images. Biases in knowledge graphs may originate in the very design of the knowledge graph, in the source data from which it is created (semi-)automatically, and in the algorithms used to sample, aggregate, and process that data. These source biases typically appear in expressions, utterances, and text sources, and can carry over into downstream representations such as knowledge graphs and knowledge graph embeddings. Furthermore, we also have to consider a large variety of human biases, as e.g. reporting bias, selection bias, confirmation bias, overgeneralization, etc.

Biases in knowledge graphs can arise from multiple sources. Data bias occurs already in the data collection process for the knowledge graph or simply from the available source data. Schema bias depends on the chosen ontology for the knowledge graph or simply is already embedded within the used ontologies [1]. Inferential bias might result from drawing inferences on the represented knowledge. Ontologies are typically defined by a group of knowledge engineers in collaboration with domain experts and consequently (implicitly) reflect the worldviews and biases of the development team. Ontologies are also prone to encoding bias depending on the chosen representation language (fragment of description logics). Moreover, biases in knowledge graph embeddings may also arise from the embedding method. Inferential biases in knowledge graphs arise at inferencing level, such as reasoning, querying, or rule learning. A simple example might be the different SPARQL entailment regimes, which in consequence, might be responsible for different results that different SPARQL endpoints deliver despite containing the same knowledge graph.

Collaboratively built knowledge graphs, as e.g. DBpedia or GeoNames also exhibit social bias, often arising from the western centered world view of their main contributors [2]. In addition, some "truths" represented in those knowledge graphs might be considered as controversial or opinionated, which underlines the importance of provenance information.

For knowledge graph embeddings that represent a vector space based approximation of the structural and semantic information contained in a knowledge graph, one of the main sources of bias lies in the sparseness and incompleteness of most knowledge graphs. Thereby, knowledge graph embeddings trained on incomplete knowledge graphs might favour entities for which more information is available [3]. However, if the underlying knowledge graph is biased, then also knowledge graph embeddings trained on this base data. De-biasing of knowledge graph embeddings requires methods for detecting as well as removing bias in knowledge graph embeddings. Depending on the underlying embedding model, this task might become complex and requires finetuning of embeddings with respect to certain sensitive relations [4, 5, 6].

References

1 Krzysztof Janowicz, Bo Yan 0003, Blake Regalia, Rui Zhu, and Gengchen Mai. Debiasing knowledge graphs: Why female presidents are not like female popes. In Marieke van Erp, Medha Atre, Vanessa López, Kavitha Srinivas, and Carolina Fortuna, editors, *Proceedings*

[4] Wikipedia article on Bias, `https://en.wikipedia.org/wiki/Bias`

of the ISWC 2018 Posters & Demonstrations, Industry and Blue Sky Ideas Tracks co-located with 17th International Semantic Web Conference (ISWC 2018), Monterey, USA, October 8th - to – 12th, 2018, volume 2180 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2018.

2 Gianluca Demartini. Implicit bias in crowdsourced knowledge graphs. In *Companion Proceedings of The 2019 World Wide Web Conference*, WWW '19, page 624–630, New York, NY, USA, 2019. Association for Computing Machinery.

3 Wessel Radstok, Melisachew Wudage Chekol, and Mirko Tobias Schäfer. Are knowledge graph embedding models biased, or is it the data that they are trained on? In *Wikidata@ISWC*, 2021.

4 Joseph Fisher. Measuring social bias in knowledge graph embeddings. *ArXiv*, abs/1912.02761, 2019.

5 Joseph Fisher, Arpit Mittal, Dave Palfrey, and Christos Christodoulopoulos. Debiasing knowledge graph embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7332–7345, Online, November 2020. Association for Computational Linguistics.

6 Mario Arduini, Lorenzo Noci, Federico Pirovano, Ce Zhang, Yash Raj Shrestha, and Bibek Paudel. Adversarial learning for debiasing knowledge graph embeddings. *ArXiv*, abs/2006.16309, 2020.

4 Lightning Talks

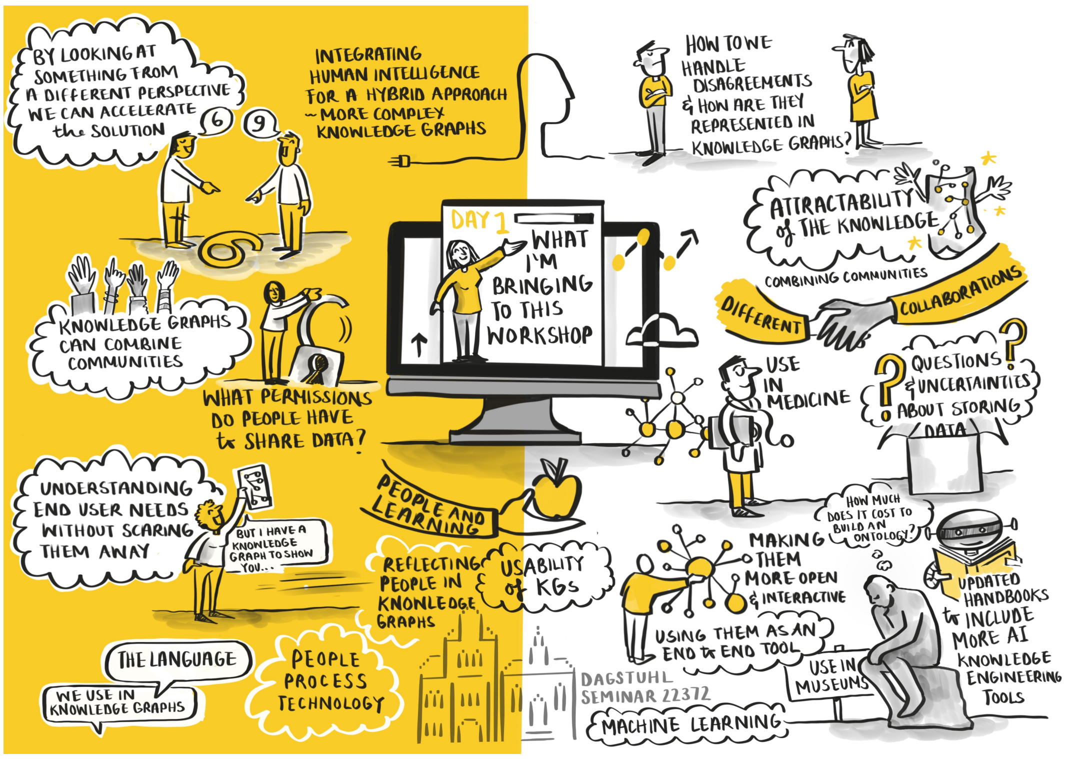
BY LOOKING AT SOMETHING FROM A DIFFERENT PERSPECTIVE WE CAN ACCELERATE the SOLUTION
6
9
INTEGRATING HUMAN INTELLIGENCE FOR A HYBRID APPROACH ~ MORE COMPLEX KNOWLEDGE GRAPHS
HOW TO WE HANDLE DISAGREEMENTS & HOW ARE THEY REPRESENTED IN KNOWLEDGE GRAPHS?
DAY 1
WHAT I'M BRINGING TO THIS WORKSHOP
ATTRACTABILITY of THE KNOWLEDGE
COMBINING COMMUNITIES
DIFFERENT
COLLABORATIONS
KNOWLEDGE GRAPHS CAN COMBINE COMMUNITIES
WHAT PERMISSIONS DO PEOPLE HAVE to SHARE DATA?
USE IN MEDICINE
QUESTIONS & UNCERTAINTIES ABOUT STORING DATA
UNDERSTANDING END USER NEEDS WITHOUT SCARING THEM AWAY
BUT I HAVE A KNOWLEDGE GRAPH TO SHOW YOU...
PEOPLE AND LEARNING
REFLECTING PEOPLE IN KNOWLEDGE GRAPHS
USABILITY of KGs
HOW MUCH DOES IT COST TO BUILD AN ONTOLOGY?
MAKING THEM MORE OPEN & INTERACTIVE
USING THEM AS AN END to END TOOL
UPDATED HANDBOOKS to INCLUDE MORE AI KNOWLEDGE ENGINEERING TOOLS
THE LANGUAGE
WE USE IN KNOWLEDGE GRAPHS
PEOPLE PROCESS TECHNOLOGY
DAGSTUHL SEMINAR 22372
USE IN MUSEUMS
MACHINE LEARNING

Figure 8 Participant Perspectives – Lightning Talks.

4.1 Organizing Scientific Contributions in the Open Research Knowledge Graph

Sören Auer (TIB - Hannover, DE)

The transfer of scholarly knowledge has not changed fundamentally for many hundreds of years: It is usually document-based-formerly printed on paper as a classic essay and nowadays as PDF. With around 2.5 million new research contributions every year, researchers drown in a flood of pseudo-digitized PDF publications. As a result, research is seriously weakened. We argue for representing scholarly contributions in a structured and semantic way as a knowledge graph. The advantage is that information represented in a knowledge graph is readable by machines and humans. As an example, we give an overview of the Open Research Knowledge Graph (ORKG[5]), a service implementing this approach. For creating the knowledge graph representation, we rely on a mixture of manual (crowd/expert sourcing) and (semi-)automated techniques. Only with such a combination of human and machine intelligence, we can achieve the required quality of the representation to allow for novel exploration and assistance services for researchers. As a result, a scholarly knowledge graph such as the ORKG can be used to give a condensed overview of the state-of-the-art addressing a particular research quest, for example as a tabular comparison of contributions according to various characteristics of the approaches. Further possible intuitive access interfaces to such scholarly knowledge graphs include domain-specific (chart) visualizations or answering natural language questions.

A detailed presentation including screenshots of the demo can be found here.

4.2 dblp as a Knowledge Graph

Marcel R. Ackermann (Schloss Dagstuhl LZI – Trier, DE)

For more than 20 years, a full XML dump of the *dblp computer science bibliography*[6] has been available as open data for download and reuse. Snapshots of the dblp XML dump have been converted to RDF before by members of the community. However, these snapshots are usually severely out of sync with the continuously curated dblp data, in some cases up to several years. To remedy this problem, the dblp team has now started to release its data also in RDF via APIs and as a full dump download. The goal is to provide a semantically rich knowledge graph of bibliographic information that is up to date and in sync with the curated and disambiguated dblp data. Just as with any other data provided by dblp, the RDF data is made available under CC0 1.0 Public Domain Dedication license.

In its initial release, the *dblp knowledge graph*[7] forms a simple person-publication graph, consisting (as of October 2022) of more than 3 million person entities, 6.3 million publication entities, and 340 million RDF triples in total. More than 15 million external resource URIs

[5] `https://orkg.org`
[6] `https://dblp.org`
[7] `https://dblp.org/rdf/`

are linked in the data set. Numerous metadata aspects, like journals/conference series or the affiliation of an author, are currently provided only as string literals. Future iterations of the schema will see these and further aspects being added as true entities, together with their own metadata, persistent IDs, and links to external resources. Hence, we don't see the current dblp knowledge graph as final, but rather as a first step in providing the semantics of the dblp dataset in a more structured way. We also aim to provide a proper SPARQL endpoint in the near future.

4.3 Triples are not Enough

Denny Vrandečić (Wikimedia Foundation – San Francisco, US)

Abstract Wikipedia aims to cover the whole breadth of knowledge that is in a usual Wikipedia article. Wikidata cannot comfortably represent the kind of knowledge necessary for the natural language text of such a Wikipedia article. We decided to work with two knowledge representations beyond triples: functions, in order to generate natural language text, and frames, in order to capture n-aries and other complex statements [1].

See [1] for more details.

4.4 Making Knowledge Graph Embeddings a First Class Citizen

Heiko Paulheim (Universität Mannheim, DE)

Knowledge graph embeddings have become a major area in the knowledge graph research landscape. There are quite a few downstream applications which do not consume the knowledge graph per se, but only the embeddings.

At the same time, embeddings are not very well integrated in current tool stacks. In many cases, developers download a dump of a knowledge graph, compute embeddings, and then feed them into the application at hand. Such a model can neither incorporate any knowledge graph dynamics, nor is it suitable if only a small excerpt of a large knowledge graph is of interest for an application at hand. [2].

Services which serve knowledge graph embeddings like KGvec2go [3] are still rare. Moreover, embeddings are rarely integrated with other KG toolstack services, such as query interfaces. For those reasons, if we want to unleash the full potential of knowledge graph embeddings, we have to integrate them more tightly into our current knowledge graph tool stacks.

4.5 Knowledge Graph Completion using Embeddings

Mehwish Alam (FIZ-Karlsruhe, Leibniz Institute for Information Infrastructure, DE & Karlsruhe Institute of Technology, DE)

Knowledge Graphs (KGs) constitute a large network of real-world entities and relationships between these entities. KGs have recently gained attention in many tasks such as recommender systems, question answering, etc. Due to automated generation and open-world assumption, these KGs are never complete. Recent years have witnessed many studies on link prediction using KG embeddings which is one of the mainstream tasks in KG completion. These KG completion methods also include methods for entity type prediction [4], i.e., given the structural, textual, or another kind of information about an entity the task is to predict the type of an entity. Over the past few years, many methods have been proposed that also utilize language models, as well as a few benchmark datasets, have also been proposed [5]. A challenge remains as to how these methods can further be applied to real-world problems such as the biomedical domain, materials sciences, cultural heritage, scholarly data [6], etc. How do these existing methods scale to a particular domain? Moreover, multilingualism is also an important aspect that is under explored, i.e., how different language chapters of a KG such as Wikidata or DBpedia can help complete a KG in one language?

4.6 Knowledge Engineering for Semantic Web Machine Learning Systems

Marta Sabou (Vienna University of Economics and Business, AT)

In line with the general trend in artificial intelligence research to create intelligent systems that combine learning and symbolic components, a new sub-area has emerged that focuses on combining machine learning (ML) components with techniques developed by the Semantic Web (SW) community – Semantic Web Machine Learning (SWeML for short). Of particular interest are the emerging variations of processing patterns used in these SWeML systems in terms of their inputs/outputs and the order of the processing units. While several such neuro-symbolic system patterns were identified previously, we performed a systematic study and analyzed nearly 500 papers published in the last decade in this area. Overall we discovered 41 different system patterns, which we categorized into six pattern types. We observed that simple patterns that only incorporate one ML module are the most frequent, however the number of modules used in SWeML Systems is growing over time leading to increasingly complex and sophisticated system architectures for these novel systems. This development raises interesting questions for our community: What does the emergence of these complex systems mean for knowledge engineering? Do we need to rethink how we create, evaluate and evolve knowledge resources to better fit the requirements of such systems? What are typical SWeML systems patterns that can be used for various knowledge engineering tasks? Can we make use of these system patterns to guide the development of knowledge-based intelligent systems?

4.7 Shifting from a Triple-centric View to a Knowledge Components View in KGs

Eva Blomqvist (Linköping University, SE)

Tool support and partial automation is essential in today's Knowledge Engineering (KE) practices. This is true both for creating schemas, e.g. ontologies, and corresponding knowledge graphs. It is rarely the case that a single triple in a KG answers a user's query, rather, users of knowledge intensive systems most often need much more complex knowledge structures. An example from our previous experience is the notion of a crime, in the policing domain. A naive look at the concept of may lead to modelling a direct relation between a crime concept and a person that committed that crime. While this may be to some extent valid in a historic record, for an ongoing police investigation however, there are only suspects that to a certain degree can be connected to the crime, based on specific evidence. Even the crime in itself may need to be represented not as a single event, but as a series of actions, that could lead to certain charges being applicable in court. On the other hand, end users, in this case the police investigator also need ways to abstract from highly complex relations, to get an overview of the main connections between events and people involved in the investigation. Hence, it becomes essential that the knowledge engineering process captures all these end-user relevant levels of granularity, i.e. not only the triple-level but as more complex knowledge components. Some previous work on ontology design patterns, and recently conceptual components, point in this direction. However, this has not yet been fully brought into KE methodologies, tools, visualisations, and reasoning methods. Even further, when automating parts of the KE methodologies, such as the population of KGs, there is a need for knowledge extraction not only at the triple level, but at the level of detecting and extracting such complex components, e.g. from natural language text, where many open challenges exist.

4.8 A Normative Knowledge Graph for Verified Identity Applications

Bradley Allen (Merit International, Inc. – Millbrae, US)

The Merit Graph is a commercial example of a knowledge graph. However, in contrast to other commercial knowledge graphs, and because of the sensitivity of its areas of application in licensing, certification, and emergency services, the Merit Graph takes special care to address the problem of ensuring that the data it contains is managed to the highest standards of truth and trust. The Merit Graph maintains metadata about the provenance of statements about relations and entities, and uses that information to establish access control over data in the graph. This metadata supports verifiable and fine-grained policies that are meant to ensure the trustworthiness of the data, as well as to prevent improper sharing of personal data with third parties. The normative specification of these policies uses principles derived from action and deontic modal logic, allowing the control of not only who can access certain data, but also who is permitted to share data they have access to with whom, a capability necessary to provide organizations the tools needed to ensure that data that they are responsible for is not compromised or abused by others with whom they share that data. The Merit Graph is formally defined in a way that it can be transformed into a set of logical statements which, combined at processing time with rules, can be used to perform automated reasoning about the data in the graph. Rules are managed as part of the schema associated with the graph, through user interfaces used by system administrators to establish policies and provide domain expertise for specific use cases. This capability is used to automatically perform syntactic and semantic validation, transformation, and enhancement of data during the ingestion and issuance of merits, personas, and folios. It can also be used to perform advanced analytics, for example, link prediction in support of the recommendation of career or educational opportunities for licensed individuals, or normative reasoning to establish additional permissions and obligations of entities represented in the graph.

4.9 Semantic Interoperability at Conceptual Level: Not Easy but Necessary

Valentina Presutti (University of Bologna, IT)

Knowledge graphs (KG) have the potential of enabling meaning-aware artificial intelligence (AI) as opposed to statistically-aware AI. Let us consider recommending systems as an example. Most of them rely on features such as popularity: if I like a song, I will be suggested to listen to other music that is very "popular" among consumers who listened to the same song before me. To act differently and being able to personalise recommendations and motivate them, AI systems need to be aware of the meaning associated with the music (or any other item) they recommend and of the preference that emerge from a consumer's previous behaviour. Encoding the meaning of music or of other subjects is a hard problem but knowledge graphs and their ability to capture and formalise domain knowledge can push AI systems toward this achievement. One main issue is that specialised, domain knowledge is often overlooked. We are literally sitting on an unprecedented global, distributed source of knowledge addressing all sorts of specialised domains (Linked Open Data – LOD) but most KG-related research is limited to analyse and reuse encyclopedic knowledge. From a study that analyses the alignment between LOD ontologies [7] it emerges that LOD is poorly linked at the conceptual level (and I speculate that these alignments are mostly based on labels and common sense). There is an opportunity and a challenge to analyse LOD's knowledge from specialised domains, to enrich it and properly link it at the conceptual level. We shall resume the Semantic Web agenda about alignment and reuse of distributed ontologies, which opens to numerous research paths: to define more expressive and flexible knowledge representation languages, informed by empirical semantics; to standardise ontology design patterns (ODP); to provide tool support that makes it easy to reuse ODP, to perform ODP-based ontology alignment, to document (automatically) ontologies and KGs, to perform ontology testing, to lexicalise ontologies, etc. Only with semantic interopertability at the conceptual level and by properly addressing specialised domains shall we make a step towards meaning-aware AI systems.

4.10 Modelling Complex Concepts

Marieke van Erp (KNAW Humanities Cluster - Amsterdam, NL)

The success of AI technologies on standardised benchmark datasets, invites us to move towards more difficult and more complex concepts and tasks. The digital humanities domain presents many opportunities for investigating the recognition and modelling of complex concepts thanks to massive digitisation efforts that have made available large and varied datasets, in multiple modalities. My work now specifically highlights the complexities in modelling a concept such as smell, dealing with its representations in various media, and how the temporal dimension of historical and linguistic research forces us to deal with issues such as changing social norms and our colonial history.

4.11 KG Magic Requires KE Magic

Stefan Schlobach (VU University Amsterdam, NL)

Expert Systems have been among the initial success stories of Knowledge Representation, showing the potential of (mostly rule-based) formalised knowledge in a variety of tasks in various domains. The enormous costs of producing such high-quality knowledge led to the development of a variety of Knowledge Engineering (KE) methodologies in the Nineties and the decades after, which focused on the challenge of creating systematic processes to formalise tacit and tribal knowledge that, while being essential for the success of a system, is very often neither explicit, nor formalised. Nowadays, Knowledge Graphs (KG) are often considered to be some kind of magic wands of modern AI with the promise to extend purely statistical, learning-based, approaches by more generalisability and explainability. This has lead to increased interest in the development of Knowledge Graphs by commercial partners. The engineering challenges for constructing such high-quality knowledge remain the same as 10, 20 or 30 years ago; tribal and tacit knowledge is still as non-explicit and non-formalised as it used to be then. My research ambition is to extend the proven socio-technical KE methodologies with recent technological advances, e.g. based on Language Models or other statistical learning-based methods, to scale-up to the required complexity of modern AI-based systems.

References

1 Denny Vrandečić. Building a multilingual Wikipedia. *Commun. ACM*, 64(4):38–41, 2021.
2 Jan Portisch, Michael Hladik, and Heiko Paulheim. Rdf2vec light–a lightweight approach for knowledge graph embeddings. *arXiv preprint arXiv:2009.07659*, 2020.
3 Jan Portisch, Michael Hladik, and Heiko Paulheim. Kgvec2go–knowledge graph embeddings as a service. *arXiv preprint arXiv:2003.05809*, 2020.
4 Russa Biswas, Radina Sofronova, Harald Sack, and Mehwish Alam. Cat2type: Wikipedia category embeddings for entity typing in knowledge graphs. In Anna Lisa Gentile and Rafael Gonçalves, editors, *K-CAP '21: Knowledge Capture Conference, Virtual Event, USA, December 2-3, 2021*, pages 81–88. ACM, 2021.
5 Genet Asefa Gesese, Mehwish Alam, and Harald Sack. Literallywikidata – A benchmark for knowledge graph completion using literals. In Andreas Hotho, Eva Blomqvist, Stefan

Dietze, Achille Fokoue, Ying Ding, Payam M. Barnaghi, Armin Haller, Mauro Dragoni, and Harith Alani, editors, *The Semantic Web – ISWC 2021 – 20th International Semantic Web Conference, ISWC 2021, Virtual Event, October 24-28, 2021, Proceedings*, volume 12922 of *Lecture Notes in Computer Science*, pages 511–527. Springer, 2021.
6 Cristian Santini, Genet Asefa Gesese, Silvio Peroni, Aldo Gangemi, Harald Sack, and Mehwish Alam. A knowledge graph embeddings based approach for author name disambiguation using literals. *Scientometrics*, 127(8):4887–4912, 2022.
7 Luigi Asprino, Wouter Beek, Paolo Ciancarini, Frank van Harmelen, and Valentina Presutti. Observing LOD using equivalent set graphs: It is mostly flat and sparsely linked. In Chiara Ghidini, Olaf Hartig, Maria Maleshkova, Vojtech Svátek, Isabel F. Cruz, Aidan Hogan, Jie Song, Maxime Lefrançois, and Fabien Gandon, editors, *The Semantic Web – ISWC 2019 – 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part I*, volume 11778 of *Lecture Notes in Computer Science*, pages 57–74. Springer, 2019.

5 Breakout Groups

5.1 Integration of Language Models and Structured Data

Juan Sequeda (data.world – Austin, US)
Mehwish Alam (FIZ-Karlsruhe, Leibniz Institute for Information Infrastructure, DE & Karlsruhe Institute of Technology, DE)
Soren Auer (TIB - Hannover, DE)
George Fletcher(Eindhoven University of Technology, NL)
Harald Sack (FIZ-Karlsruhe, Leibniz Institute for Information Infrastructure, DE & Karlsruhe Institute of Technology, DE)

This group focused on how Large Language Models (LLMs) can be integrated or used for structure data.

5.1.1 Discussed Problems

Large-scale Language Models (LLMs) have shown impressive results in terms of language generation, question answering, but also software source code generation or translation. Following the presentation of the overview of the state of the art in Automated Knowledge Graph Construction, it was observed that the surveyed methods focused on automated approaches to construct knowledge graphs from unstructured sources. The question is whether these results can be applied for automatically constructing knowledge graphs from structured data (e.g. tabular, relational), and mapping structured data to ontologies and knowledge graphs.

The following initial observations were made:

- There are two streams to consider: 1) Automatic Knowledge Graph Construction from structured data, namely given structured data as input, the output is a knowledge graph, and 2) Automatically Mapping structured data to Knowledge Graph, namely, given structured data and an existing knowledge graph as input, the output is an augmented knowledge graph.

- The second stream considers the traditional data integration challenges of schema matching and entity linking.
- Language models have common sense knowledge. However, to do the mapping, it would also need to have specific business/domain knowledge, which may not exist in language models today.

5.1.2 Possible Approaches

There have been few recent studies focusing on tabular data to KG matching [1] along with many recent efforts by the semantic web community which designs a group benchmark dataset for this problem, i.e., the SemTab [2, 3, 4] challenge where the community joins forces and presents their systems targeting the problem of tabular data to KG. None of the existing studies so far utilize LLMs for performing this matching.

On the other hand, there have been several efforts where the latent representations are learned directly from the tabular data such as Tab2Vec [5], TaBert [6], etc. Tab2Vec is then evaluated on row completion, table completion, and table retrieval tasks. TaBERT is a pre-trained model that learns representations for natural language sentences and tabular data. These efforts should further be explored and exploited for mapping structured data to ontologies and KGs. A brief collection of methods following this line of research has been discussed in [7]. A deep dive into Machine or Deep Learning methods for tabular data is required.

5.1.3 Open Research Questions

The high-level questions to consider are:

- How do we automatically construct a knowledge graph from structured data?
- How do we automatically construct mappings from structured data to a knowledge graph?

Diving deeper into these questions, we discussed the following questions:

- Do we even need LLMs for this problem? It seems that we are turning this problem into a nail for the Language Model hammer, thus we should try to use this tool. As observed, language models consist of common-sense/domain-agnostic knowledge and may lack specific domain/business knowledge, thus the limits of existing language models need to be investigated. On the other hand, if we look at schema.org, we have evidence of a manual, low-effort, distributed, community-driven, and scalable approach to adding semantics to web pages.
- What is the cost/benefit tradeoff to using LLMs? What do we do with the results of a language model? A user will most likely need to review the result. For this approach to be cost beneficial, it would need to be drastically reduced to the cost of creating the knowledge graph manually/non-language model approaches.
- How can mappings be learned with additional context provided as input, for example, mapping patterns?
- What happens when the input is just tabular data vs relational data (SQL DDL, constraints)?
- Would there be a need to denormalize the data into a single flat table?
- What are the frameworks for evaluation?

5.1.4 Next Steps

Some anecdotal evidence arising from some experiments done during the breakout session[8] showed that while LLMs are able to perform some form of data mapping on typical textbook examples, they quickly fail when data structures are more original (and thus less likely to be included in the LLM training data). Also, due to the lacking explainability of LLM results, it is extremely cumbersome and thus not feasible to manually verify the results, since the required effort for this task might easily exceed a manual mapping. However, LLMs could possibly be used for generating smaller (e.g. property) mapping or documentation suggestions.

An interesting question is whether the experience of LLMs can be applied to generate novel large-scale structured data models, which are trained with millions of data schemata, ontologies, and mappings and will thus be better suited for mapping generation tasks. However, this might not be practically feasible since many of the required artifacts are private. Possibly some federated learning of such large-scale structured data models could alleviate this problem.

When considering applying Language Models for the problems of schema mapping, it's key to understand the state of the art in order to create bridges. For example, there is formal work on learning schema mappings [8] and queries from examples of structured data.

One of the possible solutions could be to align two embeddings generated from different structures such as tabular data and the knowledge graphs and perform alignments or make use of both kinds of embeddings and use it in the downstream task of matching.

What would prompt engineering for data integration look like? This may be an extension of the existing SemTab challenge.

Finally, we should be careful and not just jump on using the Language Model hammer and start pounding on that hammer to see what works. It is paramount to have a systematic approach to understanding and evaluating how language models and structured data can be combined to automatically construct knowledge graphs.

References

1 Jixiong Liu, Yoan Chabot, Raphaël Troncy, Viet-Phi Huynh, Thomas Labbé, and Pierre Monnin. From tabular data to knowledge graphs: A survey of semantic table interpretation tasks and methods. *Journal of Web Semantics*, page 100761, 2022.

2 Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Vasilis Efthymiou, Jiaoyan Chen, Kavitha Srinivas, and Vincenzo Cutrona, editors. *Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2020) co-located with the 19th International Semantic Web Conference (ISWC 2020), Virtual conference (originally planned to be in Athens, Greece), November 5, 2020*, volume 2775 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2020.

3 Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Vasilis Efthymiou, Jiaoyan Chen, Kavitha Srinivas, and Vincenzo Cutrona. Results of semtab 2020. In Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Vasilis Efthymiou, Jiaoyan Chen, Kavitha Srinivas, and Vincenzo Cutrona, editors, *Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2020) co-located with the 19th International Semantic Web Conference (ISWC 2020), Virtual conference (originally planned to be in Athens, Greece), November 5, 2020*, volume 2775 of *CEUR Workshop Proceedings*, pages 1–8. CEUR-WS.org, 2020.

[8] `https://beta.openai.com/playground`

4 Vincenzo Cutrona, Jiaoyan Chen, Vasilis Efthymiou, Oktie Hassanzadeh, Ernesto Jiménez-Ruiz, Juan Sequeda, Kavitha Srinivas, Nora Abdelmageed, Madelon Hulsebos, Daniela Oliveira, and Catia Pesquita. Results of semtab 2021. In Ernesto Jiménez-Ruiz, Vasilis Efthymiou, Jiaoyan Chen, Vincenzo Cutrona, Oktie Hassanzadeh, Juan Sequeda, Kavitha Srinivas, Nora Abdelmageed, Madelon Hulsebos, Daniela Oliveira, and Catia Pesquita, editors, *Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching co-located with the 20th International Semantic Web Conference (ISWC 2021), Virtual conference, October 27, 2021*, volume 3103 of *CEUR Workshop Proceedings*, pages 1–12. CEUR-WS.org, 2021.

5 Li Zhang, Shuo Zhang, and Krisztian Balog. Table2vec: Neural word and entity embeddings for table population and retrieval. In Benjamin Piwowarski, Max Chevalier, Éric Gaussier, Yoelle Maarek, Jian-Yun Nie, and Falk Scholer, editors, *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 1029–1032. ACM, 2019.

6 Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. Tabert: Pretraining for joint understanding of textual and tabular data. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8413–8426. Association for Computational Linguistics, 2020.

7 Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. Deep neural networks and tabular data: A survey. *CoRR*, abs/2110.01889, 2021.

8 Balder ten Cate, Víctor Dalmau, and Phokion G. Kolaitis. Learning schema mappings. *ACM Trans. Database Syst.*, 38(4):28, 2013.

5.2 Knowledge Engineering with Language Models and Neural Methods

Elena Simperl (King's College London, GB, elena.simperl@kcl.ac.uk)
Paul Groth (University of Amsterdam, NL, p.t.groth@uva.nl)
Steffen Staab (Universität Stuttgart, DE, steffen.staab@ipvs.uni-stuttgart.de)
Marta Sabou (Vienna University of Economics and Business, AT, marta.sabou@wu.ac.at)
Eva Blomqvist (Linköping University, SE)
Bradley Allen (Merit International, Inc. – Millbrae, US, bradley.p.allen@gmail.com)

5.2.1 Discussed Problems

Knowledge engineering remains expensive. The emergence of new powerful automation tools (e.g. large language models) opens new avenues for exploration to bring down the cost of knowledge engineering. While there is much work using machine learning for knowledge engineering (e.g. ontology learning, curation), we know much less about the overall picture of the incorporation of these new machine learning (ML) techniques.

5.2.2 Open Research Questions

- How does state of the art machine learning, including large language models augment knowledge engineering processes and projects?
- What is the existing user / developer experience of machine learning tools?

- What are the roles of people and machines in current knowledge engineering process?
- How do you evaluate the added value of automation to knowledge engineering processes?
- How do you control the outputs of ML-based systems are what you need for knowledge engineering?
- What is the interplay knowledge graph engineering (with or without AI) and system engineering?

5.2.3 Possible Approaches

Classic knowledge engineering approaches (e.g. NEON [1]) tend to distinguish between management, support, development activities while knowledge graph engineering approaches are still in their infancy and tend to focus largely on development activities (e.g. relation extraction, learning class representations, refinement). Therefore, the potential for automation is much bigger when considering the management and support activities, for instance, repurposing existing knowledge graphs in new contexts, search, automatic generation of documentation (e.g. labels in multiple languages, entity and relation descriptions), or process optimization. These are just some examples, hence, what is needed is a systematic study of current practices, roles, and tools that support them. This can be achieved in several points:

- analyze emerging knowledge engineering processes to assess their automation potential building on [2, 3, 4] (e.g. through literature reviews or empirical analysis of existing code);
- run task-based studies in which the tasks would be to build knowledge graphs following established knowledge engineering methodologies using existing out-of-the-box automation tools (e.g. HuggingFace);
- case study analysis of existing knowledge engineering projects that include an AI element.

Within this analysis, a key emerging technology, is prompt engineering [5], whose outputs, based on large language models, could inform knowledge engineering activities in several ways. Here, a mapping to between the state-of-the-art in prompt engineering and knowledge engineering would be beneficial. In particular, there is a question as to how these technologies can be suitably controlled for knowledge engineering processes.

Evaluating and understanding the impact of technology is an established field with its own methodologies and approaches. In particular, there has been considerable work by researchers, practitioners and regulators around the use of machine learning in a range of applications, which resulted in frameworks for responsible/trustworthy AI [6, 7]. Studies with technical users of AI seem to suggest data scientists and other technical roles tend to over-trust the outcomes of machine learning systems and do not always fully grasp how they work, or, where applicable, their explanations [8]. End-users of downstream applications need means to provide feedback and adjust the outputs of the application, which often involves improving the underlying data – often, knowledge graph embeddings are a source of such data, hence it is important when evaluating the added value of automation in building a graph to consider questions of end-user agency and control from the start. Here, extending approaches [9] that look at machine teaching[9] to examples from knowledge graphs appears promising.

Any way to understand to knowledge engineering with AI systems should be based on the existing extensive work with designing and planning for AI systems. This includes a series of practices including, following human-centered design, identifying multiply evaluation

[9] `https://github.com/cleanlab/cleanlab`

metrics, extensive testing and continued monitoring while in deployment.[10] This also includes reflecting on fairness and interpretability, which come with their own set of best practices and techniques.

Furthermore, often knowledge graph engineering is described as one-off activity, where the project is finished when a knowledge graph is complete. In practice, this is not the case as seen by the examples discussed at the seminar. Therefore, there is to study the ongoing maintenance of knowledge graphs the roles and automation involved. This should be done with a grounding in the current thinking around data-centric AI and MLOps[10].

5.2.4 Next Steps

- Perform the user studies, case studies and reviews mentioned above;
- Organizie a workshop bringing together prompt engineering and knowledge engineering experts.

References

1. María del Carmen Suárez de Figueroa Baonza. *NeOn Methodology for Building Ontology Networks: Specification, Scheduling and Reuse.* PhD thesis, Universidad Politécnica de Madrid, Madrid, Spain, June 2010.
2. Anna Breit, Laura Waltersdorfer, Fajar J. Ekaputra, Tomasz Miksa, and Marta Sabou. A lifecycle framework for semantic web machine learning systems. In *Database and Expert Systems Applications – DEXA 2022 Workshops*, pages 359–368, Cham, 2022. Springer International Publishing.
3. Gytė Tamašauskaitė and Paul Groth. Defining a knowledge graph development process through a systematic review. *ACM Transactions on Software Engineering and Methodology*, 2022.
4. Lucie-Aimée Kaffee, Kemele M. Endris, and Elena Simperl. When humans and machines collaborate: Cross-lingual label editing in wikidata. In *Proceedings of the 15th International Symposium on Open Collaboration*, OpenSym '19, New York, NY, USA, 2019. Association for Computing Machinery.
5. Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *CoRR*, abs/2107.13586, 2021.
6. Lu Cheng, Kush R Varshney, and Huan Liu. Socially responsible ai algorithms: Issues, purposes, and challenges. *Journal of Artificial Intelligence Research*, 71:1137–1181, 2021.
7. Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information fusion*, 58:82–115, 2020.
8. Harmanpreet Kaur, Harsha Nori, Samuel Jenkins, Rich Caruana, Hanna Wallach, and Jennifer Wortman Vaughan. Interpreting interpretability: Understanding data scientists' use of interpretability tools for machine learning. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–14, New York, NY, USA, 2020. Association for Computing Machinery.
9. Curtis G Northcutt, Anish Athalye, and Jonas Mueller. Pervasive label errors in test sets destabilize machine learning benchmarks. *arXiv preprint arXiv:2103.14749*, 2021.

[10] `https://ai.google/responsibilities/responsible-ai-practices/`, `https://www.microsoft.com/en-us/ai/responsible-ai-resources`

10 Hima Patel, Shanmukha Guttula, Ruhi Sharma Mittal, Naresh Manwani, Laure Berti-Equille, and Abhijit Manatkar. Advances in exploratory data analysis, visualisation and quality for data centric ai systems. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 4814–4815, New York, NY, USA, 2022. Association for Computing Machinery.

5.3 Explainability of Knowledge Graph Engineering Pipelines

Axel-Cyrille Ngonga Ngonga (Universität Paderborn, DE, axel.ngonga@upb.de)
Diana Maynard (University of Sheffield, GB, d.maynard@sheffield.ac.uk)
Marcel R. Ackermann (Schloss Dagstuhl LZI – Trier, DE, marcel.r.ackermann@dagstuhl.de)

5.3.1 Discussed Problems

There is currently no standard definition for explanation and explainability. We consider the following explanation scenario [1]: An explainer is to provide an explanation for an explanandum to an explainee via explanans. In knowledge engineering for knowledge graphs, the explainer is commonly a system driven by some background knowledge. The explanandum could consist of the graph as a whole or a single statement, but also the process. The explainee can be a human or another system. Finally, the explanans can range from natural language to a set of assertions in a formal language. In this setting, explanation is clearly an iterative process within which the explainee can request supplementary information (e.g., pertaining to previous explanans) to reach the explanation goal. When modelled as such, the function of an explanation is to empower the explainee to understand enough about the explanandum to take action. It is rather unclear how explanations and the accompanying processes are to be tailored and evaluated.

Explainability is central for several aspects of the knowledge engineering process including building trust, quantifying uncertainty, hypothesis exploration, due diligence support, compliance and liability, data and process audits, and data usage agreements. *Trust* is a key element of explainability, enabling the explainee to evaluate the correctness / usefulness and/or actionability of the output. *Quantifying uncertainty* ensures that the explainee has a measurement for the reliability of the explanation process and hence of the explanandum. Devising pareto-optimal explanation processes that can cater for several of these aspects is a challenge which is currently not widely addressed.

5.3.2 Possible Approaches

The body of works on interpretability and explainability covers various disciplines ranging from psychology [1] to machine learning theory [2]. The ML community has developed post-hoc methods for explainability, including approaches such as LIME [3], SHAP [4], and MVU [5]. Ante-hoc solutions such as verbalization techniques for class expressions [6, 7] serve a similar purpose in inductive logic programming based on description logics. Still, these are one-shot explanations, which do not fully implement the iterative explanation process described in Section 5.3.1. Currently, there seems to be no detailed study encompassing the state of the art in theory and practice, but rather a number of piecemeal attempts to solve various issues in tackling explainability.

5.3.3 Open Research Questions

Explaining is an intrinsically challenging task, as a good explanation for a given explanandum must fit potentially very different user and application requirements. The computation of explanations must hence be carried out in collaboration with the explainee – a process which is widely unexplored in knowledge engineering. Still, it seems obvious that one-shot explanations will rarely be enough to satisfy user needs. A clear specification of the relation between explanations and interpretations (e.g., as described in [2]) must be at the core of future research as the distinction between them is unclear and often misrepresented. Further key challenges include the need to provide measures for explanation, methods to evaluate them and to quantify their trustworthiness (both intrinsically and extrinsically) and to allow for measures of uncertainty. On the other hand, since the field of explainability in this context is both fast-evolving and application-dependent, it is therefore difficult and perhaps undesirable – especially in the near future – to develop rigid standards.

5.3.4 Next Steps

We plan to write a survey of the state of the art in explainability, with the aim of understanding better the current limitations and future directions. Several workshops on explainability are currently organized at major AI conferences including XAI4CV at CVPR, and SemEx at ISWC. We plan to support these efforts and contribute requirements and solutions from knowledge engineering. Our ultimate goal is to write a reference book on methods and applications of explainable AI for knowledge engineering.

References

1 Katharina J Rohlfing, Philipp Cimiano, Ingrid Scharlau, Tobias Matzner, Heike M Buhl, Hendrik Buschmeier, Elena Esposito, Angela Grimminger, Barbara Hammer, Reinhold Häb-Umbach, et al. Explanation as a social practice: Toward a conceptual framework for the social design of ai systems. *IEEE Transactions on Cognitive and Developmental Systems*, 13(3):717–728, 2020.

2 Christoph Molnar. *Interpretable machine learning.* Lulu. com, 2020.

3 Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144, 2016.

4 Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

5 Kilian Q Weinberger and Lawrence K Saul. Unsupervised learning of image manifolds by semidefinite programming. *International journal of computer vision*, 70(1):77–90, 2006.

6 Norbert E Fuchs, Kaarel Kaljurand, and Tobias Kuhn. Attempto controlled english for knowledge representation. In *Reasoning web*, pages 104–124. Springer, 2008.

7 Axel-Cyrille Ngonga Ngomo, Diego Moussallem, and Lorenz Bühmann. A holistic natural language generation framework for the semantic web. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 819–828, Varna, Bulgaria, September 2019. INCOMA Ltd.

5.4 Construction with Modalities and Types

Antoine Isaac (Europeana Foundation - Den Haag, NL, antoine.isaac@europeana.eu)
Marieke van Erp (KNAW Humanities Cluster - Amsterdam, NL, marieke.van.erp@dh.huc.knaw.nl)

This working group focused on investigating the gap between what is currently captured in Knowledge Graphs and what information is contained in other sources and modalities. The impetus for this break-out group came from the observation that most current KGs focus on factoid-information that is easily captured in triples such as information about entities and properties.

More freely structured data such as text and images contain information that may be difficult to capture in triple format. Procedural knowledge or other knowledge that has a clear sequence (e.g. word order in text) does not naturally fit into KGs. Solutions such as the NLP Interchange Format (NIF) have been proposed but lead to bulky modelling.

Information concerning more abstract concepts such as opinions or perspectives are often implicit and have a social and contextual dimension – what is acceptable in one context may not be acceptable in another. This type of information intersects with commonsense knowledge as well as social norms. Something that, to the best of our knowledge, is currently not captured in KGs.

This break-out group therefore poses the following questions:

- What can (or should be) be included knowledge graph?
- Which source can it come from?
- What is the purpose of (elements of) the knowledge graph?

5.4.1 Discussed Problems – Towards a Typology of Knowledge for KGs

The main problems that we discussed are the identification and characterization of the types above, as well as the representation techniques that can be used to handle them.

Table 1 Types of knowledge typically found in knowledge graphs.

Type of knowledge	**Examples**	**(Typical) representation level**
Conceptual knowledge	Entities: classes, properties. Typically the knowledge written in OWL	T-Box, R-Box
Factual knowledge	Entities and statements: objects, relationships	A-Box
Procedural knowledge		
Rules		
Commonsense and encyclopedic	Naive physics, “knives are used for cutting”, “dinner is at 6pm”	
Causality		
Sentiment		
Arguments, claims	Political, scientific arguments	Qualifiers, named graphs, nanopublications
Beliefs	“No man landed on moon”	
Provenance, references		Qualifiers, named graphs
Perspectives, narratives, frames, interpretations	Colonial perspectives on objects being “given”, Tonality of a music piece (which changes as the applied theory changes), Wikidata has different entries for Jesus (in which he may be the last or second-to-last prophet)	Representations of situations, (trans-)actions and (sociological) roles
Moral and ethical judgements	“Abortion is a crime” according to certain groups of people in the US	

Definitions (and to some extent, terminology) need to be confirmed and refined for these types. For example our group discussed commonsense knowledge, only to conclude that while we have a general idea of the notion – e.g., it includes what is needed to understand the newspapers – it remains extremely vague and the term is quite overloaded.

5.4.2 Possible Approaches

A first way to refine and better structure the notions (roughly) laid down above would be to identify suitable dimensions of analysis and position the various types of knowledge along these dimensions. This idea follows upon the example of the “expressiveness spectrum” produced at AAAI99, which was presented by McGuinness at the Seminar (see fig. 9).

A complementary, more bottom-up approach, would be to inventorise the elements of knowledge used in actual KGs. As a first attempt, and recognizing that the types of knowledge present in KGs heavily depends on the domain or the application considered, the group embarked on identifying knowledge elements that are typically (or less typically) found in a few selected domains.

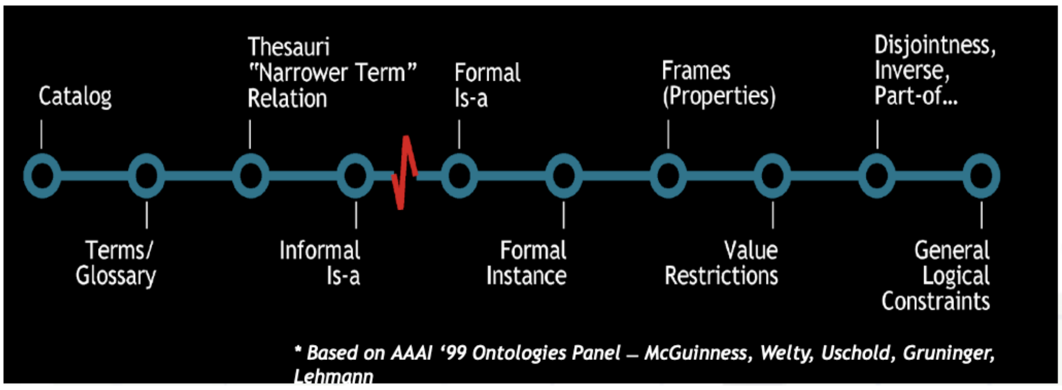
Catalog
Terms/ Glossary
Thesauri "Narrower Term" Relation
Informal Is-a
Formal Is-a
Formal Instance
Frames (Properties)
Value Restrictions
Disjointness, Inverse, Part-of...
General Logical Constraints
* Based on AAAI '99 Ontologies Panel – McGuinness, Welty, Uschold, Gruninger, Lehmann

Figure 9 Expressiveness spectrum as presented by McGuinness during the seminar (see Section 3.3 for an overview of the talk).

Table 2 Matrix in which the break-out group brainstormed types of knowledge that may be included in a knowledge graph for a particular use case.

	Conceptual	Facts	Rules	Procedural	Common sense	Sentiment, moral/ ethical judgment	Arguments	Beliefs	Provenance	Narratives/Perspectives	Causality
Contested heritage	x	x	x	(x)	x?	x	x	x	x	x	
Musicology	x	x	x	?	(x)	x	x	?	x	x	
Engineering	x	x	(x)	x	(x?)		x		x		
Agriculture	x	x		x	x		x		x	x	
Medical											
Wikidata											
Astronomy				(x)							
Bibliography											

5.4.3 Next Steps

The discussion in our group was only a first attempt at charting the landscape of the various types of knowledge that can appear in KGs. This effort needs to be continued, especially on:

- Surveying of cases and the types of knowledge that can or should be relevant for them.
- Working on one or several "spectra" of expressiveness and other dimensions, for the knowledge that can be represented in KGs.

This work, which could be progressed in a workshop-like setting (especially in order to agree on types and dimensions) and long-term community outreach effort (especially for the surveying), should be eventually presented in a written form that can benefit researchers and practitioners on the longer term – either as a separate paper or part of a wider book on knowledge engineering methodology.

5.5 Knowledge Graphs vs. Other Forms of Knowledge Representation

Paul Groth (University of Amsterdam, NL, p.t.groth@uva.nl)
Aidan Hogan (University of Chile – Santiago de Chile, CL, ahogan@dcc.uchile.cl)
Lise Stork (Vrije Universiteit Amsterdam, NL, l.stork@vu.nl)
Katherine Thornton (Yale University Library – New Haven, US, katherine.thornton@yale.edu)
Denny Vrandečić (Wikimedia Foundation – San Francisco, US, denny@wikimedia.org)

This working group focused on how knowledge graphs relate to, complement, and could be combined with, or even replaced by, other forms of knowledge representation, including traditional forms of knowledge representation like text and tables, as well as novel forms of knowledge representation such as (large) language models.

5.5.1 Discussed Problems

- How do knowledge graphs relate to other types of knowledge representation?
- What kinds of knowledge are knowledge graphs good at representing?
- Will knowledge graphs still be needed given the advancements in large language models?
- Are knowledge graphs the best target for knowledge extraction from large language models?

5.5.2 Discussion

Despite the growing popularity of knowledge graphs, it is not always clear for what sorts of knowledge (or knowledge-centric applications) such graphs are appropriate representations. Our discussion thus covered different forms of knowledge representation and knowledge, how knowledge graphs relate to modern forms of knowledge graphs like large language models, and what forms of representation are useful in what settings.

5.5.2.1 Different Representations

Knowledge graphs are a particular class of knowledge representation; some members of this class include RDF, RDF*, property graphs, Wikidata, etc. Knowledge graphs have garnered a lot of attention for their ability to integrate knowledge from diverse sources at large scale. However, they are only one instance of a particular "modality" of knowledge representation used by humans. One may thus ask: How do knowledge graphs relate to other modalities of knowledge representation? When are they more or less useful than the other alternatives? Can we model all knowledge within a knowledge graph, or do we need different representations for different types of knowledge? How could knowledge graphs be combined with, or interact with, these other representations?

With respect to modalities of knowledge representation, we can identify, for example, the following:

- *Textual*: books, literature, rich text, emails
- *Lexicographical*: thesauri, lexemes, vocabulary, dictionaries
- *Tabular*: CSV, spreadsheets, relational tables
- *Temporal*: edit histories, chronologies, stock tickers, temporal databases
- *Graph*: (social/transport/biological) networks, knowledge graphs
- *Hierarchical*: taxonomies, classifications, XML, JSON

- *Logical*: rules, ontologies, first-order logic, frames, scripts, schemas
- *Procedural*: code, instructions, workflows, tutorials
- *Multimedia*: video, audio, images
- *Diagrammatic*: UML, ER, pie charts, Sankey diagrams
- *Numeric*: embeddings, language models, matrices
- *Mental*: human memory, epigenetic memory
- *Social*: word of mouth, gossip, stories, songs, institutional memory

It is clear that knowledge graphs are not intended to replace all of these modalities, nor is it clear that they even can. To understand this in more detail, we identify some different types of knowledge:

- Factual: expressing declarative statements representing claims of truth (e.g., *the capital of Nigeria is Abuja*).
- Quantified: expressing statements for existential or universally quantified elements (e.g., *all countries have a capital*).
- Contextual: expressing statements that are claimed to be true within a certain context, such as a probability or fuzzy quantification of truth (e.g., *a country probably only has one capital*); a temporal context (*the capital of Nigeria has been Abuja since 1991*), etc.
- Procedural: expressing ways of doing things, often involving a sequence of actions and their effects (e.g., *how to prepare the Nigerian dish Tuwo shinkafa*).
- Narrative: expressing a series of statements building a model and working within that model to communicate knowledge
- Tacit: implicit knowledge often gained through lived experience; may involve qualia, such as taste, smell, touch, sight (e.g., *what Tuwo shinkafa tastes like*); socially-acquired knowledge relating to customs, values, etc. (e.g., *that it would be strange to eat Tuwo shinkafa with marmalade*), and so forth.
- Counterfactual: expressing statements of possible world states, representing what would be true under varying circumstances and often including modal terms such as "possibly" (*if I would take the bike, I would possibly not be on time).*

Knowledge graphs are perhaps strongest when representing factual knowledge, particularly when such knowledge is expressed as binary relations. When combined with rules or ontologies, they can further represent quantified knowledge. When combined with techniques such as annotated logic, reification, named graphs, RDF* or property graphs, etc., they can also be used to capture contextual knowledge. For reasoning over such complex objects, however, new formalisms would be required [1]. Though knowledge graphs are only one possible representation, they have shown certain advantages and disadvantages when compared with tables (SQL, CSV, etc.), trees (JSON, XML, etc.), images, and so forth.

How knowledge graphs can be used to capture procedural knowledge is less clear. If, for example, we wanted to represent the sequence of steps in a recipe, while we could potentially structure the recipe as a graph of dependent steps or causal relations, the steps themselves will likely be described in natural language, such as "*mash the rice with a wooden spoon*". While such unstructured steps could potentially be decomposed (potentially recursively) into a structured sequence of sub-steps, and the instruments they involve, etc., and while a more fine-grained structure might help to later find recipes satisfying certain criteria, the resulting representation will not convey very well how to actually make the recipe (a video would be better).

Moreover, it is nontrivial how to represent hypothetical knowledge – such as counterfactual knowledge – for which statements can be equally likely depending on different world states. RDF* does allow for contextualised statements without any truth value assigned to them.

These are *quoted triples*,[11] which are statements not asserted and thus not evaluated in the knowledge graph. However, keeping track of the epistemic status of a contextualised statement, given situational facts, is not yet supported.

Likewise tacit knowledge is often inherently difficult to express, particularly as structured knowledge (including knowledge graphs).

5.5.2.2 Use for Knowledge Graphs in the Age of Large Language Models

Knowledge graphs will inevitably begin to compete with – and potentially complement – language models [2, 3], which have recently captured not only interest within academia, but also the more general public, in terms of their ability to seemingly understand and communicate knowledge through natural language. Such language models thus increase machine interpretability of human language. However, many of the modalities of knowledge representation introduced previously – including knowledge graphs – were primarily introduced as a way to make knowledge available in "machine-readable" structured formats. If language models are paving the way for human natural language to become "machine-readable", and if natural language is a more intuitive way for humans to capture, express, communicate and conceptualise knowledge (including procedural and tacit knowledge), then a key question arises: assuming that language models continue to improve over time, will we even need structured representations like knowledge graphs in the future?

Knowledge graphs require knowledge to be structured, while language models are designed to capture unstructured knowledge. Thus the question of how knowledge graphs relate to language models is contained within the broader scope of how structured knowledge relates to unstructured knowledge. Viewed in this light, knowledge graphs are more efficient for query answering, are more reliably modifiable, are more transparent, cover the long tail better, and lend themselves better to explainability than large language models (which we expect to hold, also, for the medium-term future). In more detail:

- For **query answering**, looking up a triple in a triple store is more efficient than retrieving text from a large generative model with billions of parameters; e.g. it is less expensive to look up the capital of France in Wikidata than to generate that answer from GPT-3.
- If the world changes, or if an error in the knowledge is discovered, we can easily **fix, edit and update** the knowledge graph, but it is currently an open research question how a language model would need to update its weights to reflect such a change in the world or correct an existing error; e.g. if the capital of Kazakhstan gets a new name, or if the British monarch dies, how do we update a language model to incorporate that change?
- Knowledge graphs can be more **transparent** and can have clearer **provenance** as they can contain references, sources, or other ways to establish trust in the knowledge in the graph. Conversely, language models do not currently capture the connection between the weights and the textual sources used to learn these weights in a fine-grained way.
- Relatedly, knowledge graphs allow for more **explainability** than large language models. With a symbolic system we can display the involved ground statements, and the inferences that took place, whereas with language models, generating explanations is a very popular and challenging topic of active research [4].
- Knowledge graphs also **cover the long tail** better, and can be more easily extended to cover the long tail. A naive approach to increasing coverage for a language model

[11] `https://w3c.github.io/rdf-star/cg-spec/editors_draft.html#dfn-quoted`

is to retrain or refine it with more text about the topics to be covered; in a structured knowledge base you can just explicitly add the required structure. Anecdotal experience indicates that if we want to increase coverage of, e.g. different file types, we can either write or search for documents about these file types – and writing a new document may take dozens of minutes if not hours – or we can create a new item in a knowledge base, which may take half a minute.

The aforementioned advantages of knowledge graphs versus language models have clear parallels elsewhere in terms of the advantages and disadvantages of structured/deductive/symbolic methods vs. unstructured/inductive/numeric methods. In the context of Natural Language Processing and Information Extraction, for example, while machine learning methods have led to major advances in the state-of-the-art, more traditional rule- or pattern-based approaches are still often preferred for certain applications (particularly in domain-specific scenarios) as they provide more control over the process, provide more transparent and explainable results, and can work better for the long tail or for emerging knowledge (where training data is sparse).

In conclusion, we think that knowledge graphs will not become redundant due to language models, but rather both can clearly complement each other.

5.5.2.3 Representations in Practice

Large language models are good at understanding (the distribution of) language and can therefore be used in a variety of downstream tasks such as named entity recognition. However, they also come with some important draw-backs and challenges, such as the lack of provenance and explainability (as discussed in the previous section). Humans express and record a lot of knowledge in unstructured form, but even before the advent of digital computers, humans were applying structure to knowledge for the purpose of understanding as well as communication (e.g., the Periodic Table).

Existing diverse representational structures (as enumerated in Subsection 5.5.2.1) each have their specific merits. In some cases, working with just a bunch of screenshots decreases cognitive load over working with free text. In others, we need formal rules or ontologies where transparency and clarity are key, whereas in other cases a table will suffice.

The downside of using a variety of data structures alongside one another is the lack of integration and harmonisation. The question then arises, how do we enable data federation in the case of heterogeneous data structures?

One solution would be a single data structure for heterogeneous data, such as the multi-modal knowledge graph described in [5], as the go-to data structure. Such a data structure integrates multi-modal data such as lists, images, etc., queryable through a single query language. Potential pitfalls of such a heterogeneous data structure could be the added modeling complexity, resulting in data silos that would be hard to query/use, or structures that are difficult to query or understand by users. Another way to go forward would be a single knowledge based system integrating multiple types of knowledge, with a single unified query interface.

5.5.3 Open Research Questions

- How to allow for a knowledge based system that integrates different kinds of knowledge representation, but yet allow for a unified query interface?
- What types of tasks require which kind of knowledge representations?

- Language models are good in smoothly dealing with the brittleness problem of symbolic knowledge representations. How can we combine language models with knowledge graphs to gain the advantage of language models?
- Would increased use of datatypes be advantageous for knowledge graph engineering? Is that a potential approach for combining knowledge graphs with more knowledge representations?
- Can we separate individual facts or knowledge out of a language model, and store it in a more efficient representation, and thus save on parameters that would encode that knowledge, making them smaller and more efficient, while allowing them to access a knowledge graph?
- How could we track provenance for language models? How could we represent and explain where this response came from? How would we trace the lineage of statements in large language models?
- How can language models be updated?
- How can language models be adapted to better cover the long tail, emerging knowledge, etc.?

5.5.4 Next Steps

- Invite collaboration on a prototypical infrastructure that demonstrates the usefulness of combining a knowledge graph with other modalities, e.g., images and a language model.
- Setting up tasks or challenges that are expected to be very difficult for certain types of knowledge representations, and easy for others. Often benchmarks are biased towards tasks that are solvable with a given approach; for example, benchmarks for question answering over knowledge graphs will include questions that are answerable over the target knowledge graph, and might tend to exclude questions like *how can I mash rice?* or *what is EUR12.53 in USD?* that knowledge graphs are not well-suited for, even if users may often like to answer such questions. These challenges should include tasks that are easy for, say, knowledge graphs, but very challenging for language models; and vice versa. The challenges should also consider "meta-tasks", like updating the knowledge, curating answers, explaining them, etc. The challenges should be promoted within the wider machine learning communities. The best-performing approaches will likely require combining different forms of representation; thus the challenges might stimulate research on hybrid approaches.

References

1 Fabian Suchanek. The need to move beyond triples. *CEUR Workshop Proceedings*, 2593:95–104, 2020.
2 Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.
3 Badr AlKhamissi, Millicent Li, Asli Celikyilmaz, Mona Diab, and Marjan Ghazvininejad. A review on language models as knowledge bases. *arXiv preprint arXiv:2204.06031*, 2022.
4 Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144, 2016.
5 Xander Wilcke, Peter Bloem, and Victor De Boer. The knowledge graph as the default data model for learning on heterogeneous knowledge. *Data Science*, 1(1-2):39–57, 2017.

5.6 Bias in Knowledge Graph Systems

Mehwish Alam (FIZ Karlsruhe, DE)
George Fletcher (Eindhoven University of Technology, NL)
Antoine Isaac (European Foundation – Den Haag, NL)
Aidan Hogan (University of Chile – Santiago de Chile, CL)
Diana Maynard (University of Sheffield, GB)
Heiko Paulheim (Universität Mannheim, DE)
Harald Sack (FIZ Karlsruhe, DE)
Elena Simperl (King's College London, GB)
Lise Stork (VU University Amsterdam, NL)
Marieke van Erp (KNAW Humanities Cluster – Amsterdam, NL)
Hideaki Takeda (National Institute of Informatics – Tokyo, JP)

The starting point of this discussion was the overview lecture on social and technical bias in knowledge graphs presented by Harald Sack. Bias often is characterised as a disproportionate weight in favour of or against a person, group, an idea or thing, usually in a way that is considered closed-minded, prejudicial, or unfair, especially one that is preconceived or unreasoned. Biases in Knowledge Graphs (KGs) as well as potential means to address them are different from those in other AI systems, as e.g. in large language models or in image classification. KGs store human knowledge about the world in structured format, e.g., triples of facts or graphs of entities and relations, to be processed by AI systems. In the past decade, extensive research efforts have gone into constructing and utilising KGs for tasks in natural language processing, information retrieval, recommender systems, and many more. In difference to language models and image classification systems, KGs are sparse, i.e. typically only a small number of triples exist per entity. Once constructed, KGs are often considered as objective and neutral reference data sources that safeguard the correctness of other systems. In reality this is often not the case, since KGs are created with specific application context in mind. This has the undesirable effect that biases inherent to KGs may become magnified and spread through KG based systems (Bias Network Effect).

Basically, biases in KGs may arise from the following sources [1]:

Data Bias: Bias may be already inherent in the source data from which the KG is created in an automated or semi-automated way. For KGS that are collaboratively created or based on collaboratively collected information, all forms of human biases might be already incorporated. Furthermore, bias can also be introduced by the algorithms used to sample, aggregate, and process that data.

Schema Bias: Bias may be introduced via the chosen ontology as the basis for a KG, or simply be embedded within ontologies. Most times, ontologies are developed in a top-down manner with application needs or certain philosophical paradigms in mind Typically defined by a group of knowledge engineers in collaboration with domain experts, ontologies consequently (though often implicitly) reflect the worldviews and biases of the development team (human bias and anthropocentric thinking). In addition, the ontology and its modelling often depends on the chosen representation language, i.e. typically a fragment of DL, and not the other way around.

Inferential Bias: Inferential biases in KGs arise at inferencing level, such as reasoning, querying, or rule learning.

Bias inherent in KGs will directly carry over into downstream representations such as KG embeddings (KGE). Also errors and incompleteness in KGs might cause bias in KGEs due to an unbalanced distribution of facts or attributes that does not reflect an objective worldview. Furthermore the chosen KGE model might be the source of additional bias induced by application-specific loss functions.

5.6.1 Discussed Problems

Bias as a signal problem

One way in which representation bias might surface in knowledge graphs is that information which can be inferred is not explicitly represented in a knowledge graph. For example, the relation *is married to* is symmetric, and from *A is married to B*, one can infer that *B is married to A* also holds. From a logical standpoint, it is therefore sufficient to encode one of the two statements in the knowledge graph.

In [2], it was reported that a vast majority of *is married to* relations in DBpedia are only present in one direction, and there are far more statements where the subject is female and the object is male than vice versa [3]. This can be considered a gender-related representation bias in the knowledge graph, since the editors (of Wikipedia infoboxes, which DBpedia is created from) find this information more noteworthy for females than for males.

The same paper [2] also discussed logical inference as a means to cancel the representation bias. In this example, it would mean filling the slots for the symmetric relation in both directions, i.e., adding *B is married to A* for every occurrence of *A is married to B*. However, in an experimental setup, they showed that the performance of using the debiased knowledge graphs in a few downstream tasks actually leads to worse performance.

One interpretation of this outcome is that bias can actually be a signal, which can help downstream applications. The fact that a human editor considered the fact *A is married to B* noteworthy, but not *B is married to A*, actually conveys some information about A and B – mainly that B is better known for other things. Removing the bias here also implies removing the corresponding signal.

Bias as a legal and professional problem

The EU commission distinguishes between fair and unfair bias.[12] In general, national and international law, as well as the standards of professional bodies [4], provide norms regarding the development and use of knowledge and data-based systems and applications. What are the relationships between legal and professional norms such as (un)fairness, responsibility, accountability and bias in knowledge graph construction, maintenance, and use? How can we build and use knowledge graphs which reflect legal or professional guidelines regarding bias? To what extent can auditing and compliance checking of knowledge graphs be automated?

Bias as a context problem

Bias as an ethical and societal problem is another important aspect, rooted in the context of the knowledge graph, since the knowledge graph cannot be generated without context, which is usually implicit. Typical examples include political and cultural statements. The serious issue of such ethical/societal bias can be exacerbated by the naive use of a knowledge graph,

[12] `https://digital-strategy.ec.europa.eu/en/library/assessment-list-trustworthy-artificial-intelligence-altai-self-assessment`

and may even cause the denial of a whole knowledge graph (for example, some nations forbid the use of Wikipedia). Data and knowledge graph quality methodologies and methods are also typically context-driven. Questions to explored include understanding the relationships between bias and quality of knowledge graphs.

5.6.2 Possible Approaches

Bias detection

Detecting biases is not a straightforward task, requiring knowledge of the world. For example, observing in the data that there are more females married to males than vice versa is an observed representation bias. On the other hand, observing in the data that there are far more male than female Nobel prize winners is not a representation bias in the data, but an accurate rendition of the state of the world.

Moreover, identifying bias requires some intuition of what to look for. While some sensitive attributes (e.g., gender or nationality) are quite straightforward, others are not, and may require several iterations of observing downstream behaviour in a system using a knowledge graph. For example, in [5], it was found that different language editions of DBpedia have a different information density of movies with respect to their genre – a representation bias that would be hard to anticipate without observing it in a downstream task.

Methods for detecting bias suggested in the literature so far often anticipate that the user knows which bias to look for, and then query the knowledge graph to get some statistics out (e.g., the proportion of male and female subjects in statements with a given property). A more open approach to this would be to learn patterns from the graphs, and then let a user decide whether those patterns represent biases in the data or distributions in the real world.

Representing and documenting bias

In contrast to language models or image classification systems, where bias can only be detected implicitly, and explicit bias descriptions have to be added separately, KGs offer the means for an explicit internal representation of bias, legal norms, and further guidelines by definition. Once bias is detected, it would be helpful to document it. If the bias occurs in the form of some pattern, this could be done using a pattern description language, such as SHACL.[13] Moreover, some statistical information would be required, as, e.g., defined by the VoiD vocabulary.[14]

Handling bias

Documenting bias is a first step to handling bias, but it is not the end of the line. Depending on the requirements and task at hand, different ways of further handling bias are possible. Applying negotiation protocols is an option for dealing with conflicting information, but may not be possible for truly controversial information. In such cases, the authors of [6] suggest allowing controversial information with additional metadata. Depending on the task at hand, bias may also be removed or handled by means of resampling methods. However, as the experiment reported above shows, this might not always be an efficient method.

[13] `https://www.w3.org/TR/shacl/`
[14] `https://www.w3.org/TR/void/#statistics`

5.6.3 Open Research Questions

Handling Bias

The *bias as a signal* observation gives rise to the conclusion that bias – although a mostly negatively connoted term – can also be helpful, and that blindly removing bias is therefore not the only or necessarily best option. Instead, more sophisticated ways of dealing with bias are required.

Detecting bias

As discussed above, bias detection requires world knowledge. For a fully automated detection of bias, one would therefore need a fully objective and bias-free knowledge base of world knowledge. This presumption – which, at least today, is impossible to meet – shows that fully automatic bias detection is currently impossible. Therefore, manual intervention will be required to detect bias in knowledge graphs, and the processes and models to do so in the best and most efficient way are still to be explored. This is not a solely computational issue. Rather, people with various disciplines should commit to the whole life cycle form generation to use of knowledge graph (diversity and inclusion issue).

Representing bias

As discussed above, a bias that is detected in a knowledge graph should at least be documented. However, to the best of our knowledge, no standards for documenting biases exist so far. Therefore, the representation of bias is still an open research issue.

Bias needs not only to be documented for humans, but also machines. Once a standard for bias representation has been defined, it would be another open question of how subsequent steps, e.g., machine learning operations, may be informed about that bias, and then how to carry out appropriate remedies (e.g., by internally re-sampling the data).

Compliance and auditing

How could we support KG engineers in building legally compliant KGs, and how could we support government bodies in (semi-)automated auditing of KGs for legal compliance (e.g., EU regulations on responsible data and AI)? How would these goals be balanced with methods and tools for bias negotiation (e.g., legal compliance vs. protecting personal safety)? More generally, further work at the intersection KG engineering and Computational Law is called for.

5.6.4 Next Steps

Targeted directions for continuing this discussion include:

- A vision paper, fleshing out the roadmap sketched above.
- Activities to bridge the knowledge graph engineering community and scholars working in the legal, professional, and cultural, societal aspects of data and knowledge. Avenues here include multidisciplinary workshops, panel discussions, Dagstuhl Seminars, research consortia (e.g., EU COST action), and working groups.
- Standardization of vocabularies and standards for bias representation.

References

1 Krzysztof Janowicz, Bo Yan, Blake Regalia, Rui Zhu, and Gengchen Mai. Debiasing knowledge graphs: Why female presidents are not like female popes. In Marieke van Erp, Medha Atre, Vanessa López, Kavitha Srinivas, and Carolina Fortuna, editors, *Proceedings of the ISWC 2018 Posters & Demonstrations, Industry and Blue Sky Ideas Tracks co-located with 17th International Semantic Web Conference (ISWC 2018), Monterey, USA, October 8th – to – 12th, 2018*, volume 2180 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2018.

2 Andreea Iana and Heiko Paulheim. More is not always better: The negative impact of a-box materialization on rdf2vec knowledge graph embeddings. *arXiv preprint arXiv:2009.00318*, 2020.

3 Eduardo Graells-Garrido, Mounia Lalmas, and Filippo Menczer. First women, second sex: Gender bias in wikipedia. In *Proceedings of the 26th ACM conference on hypertext & social media*, pages 165–174, 2015.

4 Urs Gasser and Carolyn Schmitt. The Role of Professional Norms in the Governance of Artificial Intelligence. In *The Oxford Handbook of Ethics of AI*. Oxford University Press, 2020.

5 Michael Matthias Voit and Heiko Paulheim. Bias in knowledge graphs–an empirical study with movie recommendation and different language editions of dbpedia. *arXiv preprint arXiv:2105.00674*, 2021.

6 Gianluca Demartini. Implicit bias in crowdsourced knowledge graphs. In *Companion Proceedings of The 2019 World Wide Web Conference*, pages 624–630, 2019.

5.7 Generating User and Developer Buy-in

Filip Ilievski (University of Southern California – Marina del Rey, US, ilievski@isi.edu)
Lydia Pintcher (Wikimedia Deutschland – Berlin, DE, lydia.pintscher@wikimedia.de)
Florian Reitz (Schloss Dagstuhl LZI - Trier, dblp group, DE)
Bradley P. Allen (Merit International, Inc. – Millbrae, US, bradley.p.allen@gmail.com)
Axel-Cyrille Ngonga Ngomo (Universität Paderborn, DE, axel.ngonga@upb.de)
Katherine Thornton (Yale University Library – New Haven, US, katherine.thornton@yale.edu)
Paul Groth (University of Amsterdam, NL, p.t.groth@uva.nl)
Denny Vrandečić (Wikimedia Foundation – San Francisco, US, denny@wikimedia.org)

This working group focused on how knowledge graphs can be made more attractive for regular developers of applications and services. We wanted to figure out how to make the vast amount of value created in resources like Wikidata accessible to a broad developer audience.

5.7.1 Discussed Problems

We discussed how to make accessing the data available in knowledge graphs quicker, cheaper and more efficient for developers of applications and services, especially those who have not been in contact with knowledge graphs before. This is becoming especially relevant as artificial intelligence and machine learning systems are becoming more prevalent and knowledge graphs can be a powerful tool to improve them.

Developers trying to work with knowledge graphs are facing a number of pain points. We discussed the various pain points we encountered in our own work with developers building applications and services on top of different knowledge graphs. Some of these pain points relate to the data in the knowledge graph and some to the tooling around that data. The following pain points were identified:

- Getting incorrect answers to queries: Developers are getting incorrect answers to their queries, which directly harms adoption and trust. This may be caused by issues in the data, the modeling of the data or the query itself.
- Dislike of identifiers, especially opaque identifiers: Many developers seem to dislike the prevalent use of identifiers in knowledge graphs, especially opaque ones like they are used for example in Wikidata. Developers want to use human-readable labels in their code instead.
- Schema discovery: The same data can often be modeled in different ways in a knowledge graph. To write queries that give them the answers they need, developers need to first get an understanding of how the data they are interested in is modeled. This can be challenging, especially if exploratory tools are not at hand.
- Adapting to new interfaces: There are various user interfaces developers are expected to work with when developing with data from a knowledge graph such as a query UI. These have a learning curve.
- Unclear and unhelpful error messages: When writing queries developers make mistakes and sometimes produce syntactically invalid queries. The errors they get back from the query systems are often not helpful for them to identify the problem and improve their query.

During the discussion, it became clear that more work is needed to define the exact target group of this developer outreach to make it successful. We need to better understand their needs, motivations, additional pain points and the environment they are working in. We also need to articulate more clearly what problem areas knowledge graphs are particularly well equipped to solve. A list of prototypical example use cases was considered particularly helpful in addition.

It might help to analyze positive existing use cases for KGs in a commercial setting, including data unit testing, content enrichment, geographical visualization, easy access to multilingual labels, and infobox extraction with a single query. It also seems helpful to understand the experience and the motivation of the library community, which has bought into knowledge graphs, perhaps after being shown how Wikidata can answer questions that could not be answered before. However, applying the same approach to other communities and use cases might bring novel challenges.

5.7.2 Possible Approaches

To facilitate value creation for software developers brought by knowledge technologies, we propose a combination of the following eight approaches:

1. **Conduct user studies** to better understand the target group. Software developers should be asked to perform representative tasks, and monitoring tools should be included to understand their mental model. Users should be asked to provide feedback on what was easy, what was difficult, what went wrong, and what could be improved.
2. **Log user actions** to help us understand typical user needs that are expressed through their queries.
3. **Provide useful knowledge subsets** from Wikidata that users can easily download and plug in their tools. These subsets should be provided in a developer-friendly format, like TSV or JSON.
4. **Provide users with atomic functions for common operations,** based on the persona needs derived from user studies and logging (points 1 and 2). Some operations would

include getting labels and aliases for an item, describing an item, query for similar items, text search for entities or events, fact extraction, and extracting a subset for reuse.

5. **Provide example use cases** as Jupyter and Colab notebooks, to illustrate the simplicity, effectiveness, and efficiency of including knowledge technologies in developer tasks. This should include a discussion on why this technology is the one to use.
6. **Enable users to develop a proof of concept (POC)** quickly, based on the atomic functions and the example use cases. This POC is primarily meant to convince developers, but it is also essential to secure management buy-in, as people are best convinced by showing, not telling.
7. **Make knowledge technologies relevant to the developer world,** by designing them to follow developer best practices as closely as possible, including graphical interfaces, APIs, data unit testing, and GitHub actions and releases. We should not expect developers to change their habits and make sacrifices to adopt knowledge technologies.
8. **Solicit developer feedback** to understand remaining pain points and listen to their suggestions for further improvement.

5.7.3 Open Research Questions

There seems to be a limited understanding of knowledge graph adaptations by developers. As a first step, we need a better understanding of why developers are adopting knowledge graphs and – most importantly – why not. For this research, we first need a clear understanding of the developer role and the different applications for which knowledge graphs can be adopted. The developer role needs to be clearly distinguished from other roles, such as data engineer and end user. In a second step, we need to determine what categories of information are needed and how the pain points outlined above hinder the adoption of knowledge graphs. Amongst others, we have to look into aspects such as:

Data access and presentation: What types of interfaces do developers require and how are their requirements fulfilled by current tools? E.g., how are SPARQL endpoints perceived as points of access? What are advantages and what are problems that developers face when using them? In what ways should the data be represented, i.e., triple-based formats vs. other data formats?. A possible result would be a list of atomic interaction patterns (such as API calls) that are considered to be beneficial or to be avoided when providing a knowledge graph data interface.

Tool requirements: What tools are needed to access a knowledge graph – again, specifically from the perspective of a developer? E.g., what data inspection/visualization tools are needed and how do requirements for these tools differ from requirements of other roles?

Data ownership: What role does data ownership play in knowledge graph adoption? What are the hurdles/concerns in using a shared knowledge graph such as Wikidata, particularly in a commercial setting? What roles do data quality and trust issues play in adopting shared knowledge graphs?

In a second step we need to develop new or improve existing tools to better align with the needs of developers and to provide an overall better experience. These tools need to be evaluated based on the understanding of user requirements we developed. We also need to evaluate the usefulness of the possible approaches mentioned above to increase the buy-in of developers.

5.7.4 Next Steps

1. **Research on persona descriptions and needs** – The most urgent challenge with developer buy-in is social rather than technical. We need to understand what we mean exactly by developers, what are their knowledge needs, and what is their current workflow. A possible entry point is an existing user group in the library community, which has seen the value of knowledge and has gradually embraced knowledge technologies in its practices.
2. **Map of different knowledge types in relation to representation formats** – Knowledge graphs are likely not the optimal format to store every kind of data. Geo-coordinates, for instance, might be stored more efficiently in a database that supports numeric querying with high precision, and text-heavy knowledge might be better captured by language models or ElasticSearch indices. It is essential to provide a rule of thumb for which kind of representation and resource should be the primary source for which kind of knowledge. A comprehensive figure or webpage would be a great initial format for such a map.
3. **Cookbook style documentation for developers** – Performing a knowledge technology task is overwhelming without understanding the landscape of available knowledge graphs and tooling. This could be a challenge at the beginning of using this technology, but also later in the process. To improve the developer experience, we aim to develop cookbook-style documentation that will enable developers to find relevant knowledge sources and tools as efficient as possible. The cookbook should ideally also include example use cases with code as supplementary material.
4. **Release data subsets with high reuse potential** – Well-understood datasets like MNIST have been key drivers of user-friendly tools in data science, like scikit-learn. Similarly, developing high-quality Wikidata subsets with high reuse potential, like a list of all countries or English labels, will provide an attractive playground for developers and inspire them to include ready knowledge in their frameworks. The downloads of the published subsets should be tracked to understand which, if any, are adopted by developers.

5.8 A Core Knowledge Engineering Methodology for Knowledge Graphs

Eva Blomqvist (Linköping University, SE)
Deborah McGuinness (Rensselaer Polytechnic Institute – Troy, US)
Valentina Presutti (University of Bologna, IT)
Marta Sabou (Vienna University of Economics and Business, AT)
Juan Sequeda (data.world – Austin, US)
Steffen Staab (Universität Stuttgart, DE)

This working group focused on the state of Knowledge Engineering (K) methodologies today, their relation to ontology engineering methodologies and other types of methodologies, as well as identifying open issues and needs in this domain.

5.8.1 Discussed Problems

The discussion started with an inventory of methodologies used by the participants and experiences of using such methodologies for knowledge engineering in the past. The inventory of methodologies included mainly ontology engineering methodologies [1], [2], [3], [4], [5] and others. Some of the issues discussed were:

- Lack of transparency, e.g. provenance data, in many projects. Auditing the project can be an important tool to understand what went well and what did not.
- Methodologies need to be agile, to some extent. Waterfall-style processes do not work. Also the setup of the projects are different, i.e. centralized of decentralized.
- Methodologies need to start from use cases, and most of them do, but they differ in how much guidelines are given on how to actually capture and describe use cases, and to elicit requirements from them. A good template is essential, and it should include Competency Questions. It is also important to be able to pick or tailor the methodology base on the type of use case. An enterprise data integration project has different needs than a collaborative open data project. In addition, most projects also need to cater for the unknown use cases of tomorrow – how can a methodology incorporate that?
- Costs of the methodology should be considered – KE is expensive. There needs to be guidelines on how to reduce the costs, and adapt methodology to the available resources. Human-machine collaboration, and crowdsourcing can be such means to reduce costs.
- Types of stakeholders and users, and different roles of users, is another important aspect that a methodology should cover.
- Reuse, e.g. both of existing data artefacts, code lists, and existing ontologies, are not in focus of most methodologies, but often an ad-hoc add-on activity. Also design patterns is an important kind of reuse of best practices and proven solutions.
- Evaluation and testing is often overlooked, or restricted only to assessing basic sanity criteria. Very few test-driven methodologies, and to some extent activities such as ontology testing are still under explored, compared to in for instance software engineering. Evaluation needs to be more structured and with better tool support. However, human-centric evaluation methodologies are also crucial.
- Focusing only on ontology engineering is too restrictive. A KE methodology needs to include also data curation, data integration/mapping, population of the knowledge (graph), and should put the project into its context, e.g. software engineering.
- Current tool support is far from perfect, and new tools are emerging to automate additional steps in KE methodologies, e.g. through ML approaches and language models. Most tools operate on the triple level, but Knowledge Engineers, and in particular domain experts, think in terms of other conceptual units, i.e. more complex structures.
- Although methodologies should not be too prescriptive, knowledge engineers that are not experts need a good cookbook, with rules of thumb etc.

An observation of the group was that at a high level existing methodologies are quite similar, and can be updated and consolidated to give a more coherent view of the KE processes. However, they are also to some extent lacking in that they do not cover the whole process, and do not take into account the relation to, for instance, whether the project takes place in a software engineering context, is conducted more independently, or in an open collaborative setting, such as crowd-sourcing.

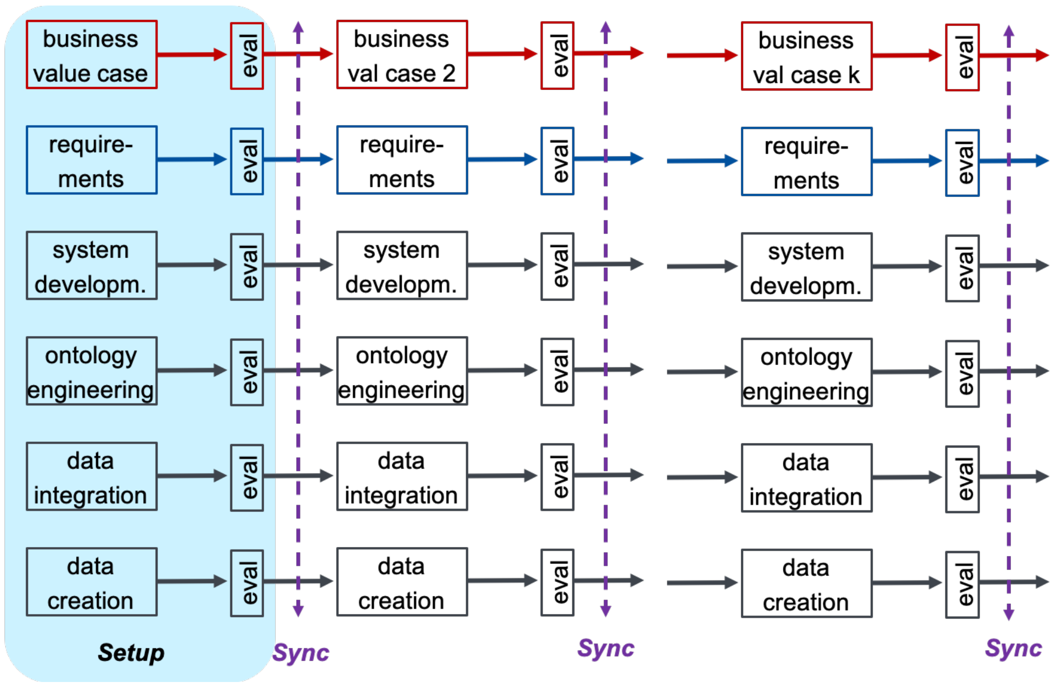

Figure 10 A blueprint knowledge engineering process that connects multiple methodologies for developing knowledge graphs.

5.8.2 Possible Approaches

There are many existing methodologies, both earlier KE methodologies and current ontology engineering methodologies. At an abstract level these are often quite similar, e.g. iterative processes to incrementally build up the knowledge model, but they are also often too narrow in scope, since they do not take into account the interaction with the context in which the KE process happens. Such context can for instance be a software project, intended to provide some business value to a company. In addition, many such methodologies also do not take into account the population of the knowledge model being built, i.e. the data integration and curation efforts needed to put the knowledge into use. Therefore some work is needed that considers the overall picture of KE in context, as illustrated in Figure 10.

Each step in such a process, i.e. the boxes in the figure, can then be more or less automated, and supported by various tools and detailed methodologies. However, the overall core KE methodology will still remain largely the same. A similar effort was also presented in [6].

5.8.3 Open Research Questions

- How does the sync between the methodologies in Figure 10 actually happen?
- What kind of evaluations are to be performed in each step, and overall?
- How can certain steps be automated or crowdsourced, and to what extent? What are the quality implications?
- How do current Knowledge Engineers actually perform these steps in practice? What are the bottlenecks and challenges?

5.8.4 Next Steps

Several possibilities for follow-up publications are discussed and will be pursued by the working group.

References

1 S. Staab, R. Studer, H.-P. Schnurr, and Y. Sure. Knowledge processes and ontologies. *IEEE Intelligent Systems*, 16(1):26–34, 2001.

2 Juan F. Sequeda and Daniel P. Miranker. A pay-as-you-go methodology for ontology-based data access. *IEEE Internet Computing*, 21(2):92–96, 2017.

3 Eva Blomqvist, Karl Hammar, and Valentina Presutti. Engineering ontologies with patterns-the extreme design methodology. *Ontology Engineering with Ontology Design Patterns – Foundation and Applications*, (25):23–50, 2016.

4 James L Benedict, Deborah L McGuinness, and Peter Fox. A semantic web-based methodology for building conceptual models of scientific information. In *AGU Fall Meeting Abstracts*, volume 2007, pages IN53A–0950, 2007.

5 María Poveda-Villalón, Alba Fernández-Izquierdo, Mariano Fernández-López, and Raúl García-Castro. Lot: An industrial oriented ontology engineering framework. *Engineering Applications of Artificial Intelligence*, 111:104755, 2022.

6 Gytė Tamašauskaitė and Paul Groth. Defining a knowledge graph development process through a systematic review. *ACM Transactions on Software Engineering and Methodology*, 2022.

6 Conclusions and Open Questions

Advances in neural and symbolic AI approaches [1], including knowledge graphs, prompted us to organise a Dagstuhl Seminar to chart the next frontiers of knowledge engineering in this brave new world. Participants reviewed the past, current, and emerging landscape of approaches, practices, and tools in knowledge base and knowledge graph construction, identified open research questions, and proposed next steps to address them.

The knowledge graph life cycle was a focal point of discussion. There was consensus that we need a sustained effort to update and upgrade classical ontology engineering methodologies [2] and develop end-to-end open-source infrastructure to make the most of the latest neurosymbolic technologies and tools, hence taking knowledge engineering and knowledge graphs beyond structured and semi-structured data to other modalities.

There are several canonical examples of knowledge graph architectures in use today. Within organisations, knowledge graphs are instrumental to data, content, and knowledge management [3]. KG projects essentially follow variants of classical ontology engineering methodologies, supported by a range of platforms and specialist tools e.g., taxonomy/ontology editors, graph databases, semantic mappers etc. In conjunction with machine learning, knowledge graphs are also used in semantic search, zero-shot learning, dialogue systems and recommender systems as a source of knowledge and explanations. Some of the best known knowledge graphs today, for instance in web search (Google, Microsoft), social networks (LinkedIn), and intelligent assistants (Siri, Alexa) achieve scales that were inconceivable decades ago – this is possible only with the help of automation, in particular using the latest developments in machine learning including generative models pre-trained on huge amounts of online data. It was recognised at the seminar that this AI-centric architecture with human-in-the-loop is not well supported in terms of methodologies and end-to-end tools. Finally, a third category of knowledge graphs are open-source and built by lively online communities [4]. While they have found considerable adoption in research and practice, their success is difficult to replicate in closed, proprietary settings, though they do provide invaluable insight into the sociotechnical ecosystem in which knowledge is created and shared.

As knowledge graph construction is making use of increasingly sophisticated, yet opaque AI capabilities, knowledge engineering must, like any other community using AI face its fairness, accountability, and transparency challenges. Several break-out workshops during the seminar considered common trustworthy AI concerns such as interpretability, biases, as well as human-AI interaction more generally, arguing for the need for bespoke solutions that target a range of end-users and stakeholders unique to knowledge engineering settings.

Finally, participants shared best practices and ideas to continue the knowledge and technology transfer efforts of the last two decades that have made knowledge graphs the backbone of systems as diverse as search engines, recommenders, chatbots, and enterprise data management platforms. They suggested activities to build capabilities and skillsets to use the latest neurosymbolic technologies and tools in knowledge graph construction, including tutorials, workshops, and hackathons, and agreed to work on joint frameworks and knowledge engineering methodologies. They also recognised the sustained need to promote knowledge graphs to the wider developer community and communicate their benefits, for instance, alongside neural methods.

As a community invested in knowledge representation and engineering, the participants embrace neural solutions such as language models for the step change they brought about in automating knowledge graph construction. At the same time, and looking back at decades of projects and experience with capturing knowledge in computational representations within organisations and on the open web, they are convinced that the use of such solutions will require human-in-the-loop approaches that are trusted and trustworthy. One of the reasons why enterprise knowledge graphs have become so successful is their ability to combine efficient, flexible storage of data with tractable representations of domain knowledge, while guaranteeing data integrity. If enterprise knowledge graph platforms are to adopt the latest advances in machine learning these guarantees will be as critical as ever. [15]

6.1 Continuing the Conversation

To continue the conversation, we provided organizers of EKAW 2022 the 23rd International Conference on Knowledge Engineering and Knowledge Management input for a *walkshop*. We prompted them with the following questions coming from the seminar:

- What ways does knowledge engineering deliver value today? What should be the requirements for knowledge production processes?
- What does user centric knowledge engineering look like including does it integrate into standard software engineering processes?
- How can new technologies help automate manual tasks such as knowledge elicitation, documentation, etc?
- How and to what extent do we integrate language models and knowledge engineering

[15] For an individual perspective of the seminar, we refer the reader to the trip report by Juan Sequeda: `http://www.juansequeda.com/blog/2022/09/20/knowledge-graphs-and-their-role-in-the-knowledge-engineering-of-the-21st-century-dagstuhl-trip-report/`

6.2 Open Questions

Finally, throughout this report we have identified many open questions for futher study. We itemize them here:

Knowledge Engineering Practice

- How does state of the art machine learning, including large language models augment knowledge engineering processes and projects?
- What is the existing user / developer experience of machine learning tools?
- What are the roles of people and machines in current knowledge engineering process?
- How do you evaluate the added value of automation to knowledge engineering processes?
- How do you control the outputs of ML-based systems are what you need for knowledge engineering?
- What is the interplay knowledge graph engineering (with or without AI) and system engineering?
- How doe we synchronize knowledge engineering methodologies?
- What kind of evaluations are needed to be performed in each step, and overall for knowledge engineering methodologies?
- How can certain steps in knowledge engineering be automated or crowdsourced, and to what extent? What are the quality implications?
- How do current Knowledge Engineers actually perform methodological steps in practice? What are the bottlenecks and challenges?

Types of Knowledge

- What are cases and the types of knowledge that can or should be relevant for people.
- Provide one or several "spectra" of expressiveness and other dimensions, for the knowledge that can be represented in KGs.
- How to allow for a knowledge based system that integrates different kinds of knowledge representation, but yet allow for a unified query interface?
- What types of tasks require which kind of knowledge representations?
- Language models are good in smoothly dealing with the brittleness problem of symbolic knowledge representations. How can we combine language models with knowledge graphs to gain the advantage of language models?
- Would increased use of data types be advantageous for knowledge graph engineering? Is that a potential approach for combining knowledge graphs with more knowledge representations?
- Can we separate individual facts or knowledge out of a language model, and store it in a more efficient representation, and thus save on parameters that would encode that knowledge, making them smaller and more efficient, while allowing them to access a knowledge graph?
- How could we track provenance for language models? How could we represent and explain where this response came from? How would we trace the lineage of statements in large language models?
- How can language models be updated?
- How can language models be adapted to better cover the long tail, emerging knowledge, etc.?

Explanations and Bias

- What is a clear specification of the relation between explanations and interpretations?
- What are measures for explanation and methods to evaluate them and to quantify their trustworthiness (both intrinsically and extrinsically) and to allow for measures of uncertainty?

- How doe we detecting bias using knowledge?
- How to representing bias?
- How can steps, e.g., machine learning operations, be informed about bias, and then how to carry out appropriate remedies?
- How could we support KG engineers in building legally compliant KGs, and how could we support government bodies in (semi-)automated auditing of KGs for legal compliance (e.g., EU regulations on responsible data and AI)?
- How would these goals be balanced with methods and tools for bias negotiation (e.g., legal compliance vs. protecting personal safety)?

Developer Experience

- What types of interfaces do developers require and how are their requirements fulfilled by current tools?
- What are advantages and what are problems that developers face when using tools?
- In what ways should the data be represented, i.e., triple-based formats vs. other data formats?
- What tools are needed to access a knowledge graph – again, specifically from the perspective of a developer? E.g., what data inspection/visualization tools are needed and how do requirements for these tools differ from requirements of other roles?
- What role does data ownership play in knowledge graph adoption? What are the hurdles/concerns in using a shared knowledge graph such as Wikidata, particularly in a commercial setting?

References

1 Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

2 Elisa F Kendall and Deborah L McGuinness. Ontology engineering. *Synthesis Lectures on The Semantic Web: Theory and Technology*, 9(1):i–102, 2019.

3 Juan Sequeda and Ora Lassila. Designing and building enterprise knowledge graphs. *Synthesis Lectures on Data, Semantics, and Knowledge*, 11(1):1–165, 2021.

4 Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

Participants

- Marcel R. Ackermann
Schloss Dagstuhl – Trier, DE
- Mehwish Alam
FIZ Karlsruhe, DE
- Bradley Allen
Merit – Millbrae, US
- Sören Auer
TIB – Hannover, DE
- Eva Blomqvist
Linköping University, SE
- George Fletcher
Eindhoven University of Technology, NL
- Paul Groth
University of Amsterdam, NL
- Aidan Hogan
University of Chile – Santiago de Chile, CL
- Filip Ilievski
USC – Marina del Rey, US
- Antoine Isaac
Europeana Foundation – Den Haag, NL
- Diana Maynard
University of Sheffield, GB
- Deborah L. McGuinness
Rensselaer Polytechnic Institute – Troy, US
- Axel-Cyrille Ngonga Ngomo
Universität Paderborn, DE
- Heiko Paulheim
Universität Mannheim, DE
- Lydia Pintscher
Wikimedia – Germany, DE
- Valentina Presutti
University of Bologna, IT
- Florian Reitz
Schloss Dagstuhl – Trier, DE
- Marta Sabou
Wirtschaftsuniversität Wien, AT
- Harald Sack
FZ Karlsruhe, DE
- Stefan Schlobach
VU University Amsterdam, NL
- Juan F. Sequeda
data.world – Austin, US
- Elena Simperl
King's College London, GB
- Steffen Staab
Universität Stuttgart, DE
- Lise Stork
VU University Amsterdam, NL
- Hideaki Takeda
National Institute of Informatics – Tokyo, JP
- Katherine Thornton
Yale University Library – New Haven, US
- Marieke van Erp
KNAW Humanities Cluster – Amsterdam, NL
- Denny Vrandečić
Wikimedia – San Francisco, US