information

Review

Construction of Knowledge Graphs: Current State and Challenges

Marvin Hofer [1,*], Daniel Obraczka [1], Alieh Saeedi [2], Hanna Köpcke [1,3] and Erhard Rahm [1,2]

[1] Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI) Dresden/Leipzig, 04105 Leipzig, Germany; obraczka@informatik.uni-leipzig.de (D.O.); koepcke@hs-mittweida.de (H.K.); rahm@informatik.uni-leipzig.de (E.R.)
[2] Department of Computer Science, Leipzig University, 04109 Leipzig, Germany; saeedi@informatik.uni-leipzig.de
[3] Faculty Applied Computer Sciences & Biosciences, University of Applied Sciences Mittweida, 09648 Mittweida, Germany
* Correspondence: hofer@informatik.uni-leipzig.de

Abstract: With Knowledge Graphs (KGs) at the center of numerous applications such as recommender systems and question-answering, the need for generalized pipelines to construct and continuously update such KGs is increasing. While the individual steps that are necessary to create KGs from unstructured sources (e.g., text) and structured data sources (e.g., databases) are mostly well researched for their one-shot execution, their adoption for incremental KG updates and the interplay of the individual steps have hardly been investigated in a systematic manner so far. In this work, we first discuss the main graph models for KGs and introduce the major requirements for future KG construction pipelines. Next, we provide an overview of the necessary steps to build high-quality KGs, including cross-cutting topics such as metadata management, ontology development, and quality assurance. We then evaluate the state of the art of KG construction with respect to the introduced requirements for specific popular KGs, as well as some recent tools and strategies for KG construction. Finally, we identify areas in need of further research and improvement.

Keywords: Knowledge Graphs; data integration; data science

Citation: Hofer, M.; Obraczka, D.; Saeedi, A.; Köpcke, H.; Rahm, E. Construction of Knowledge Graphs: Current State and Challenges. *Information* **2024**, *15*, 509. https://doi.org/10.3390/info15080509

Academic Editor: Ryutaro Ichise

Received: 5 July 2024
Revised: 15 August 2024
Accepted: 19 August 2024
Published: 22 August 2024

1. Introduction

Aggregated machine-readable information in the form of Knowledge Graphs (KGs) serves as the backbone of numerous data science applications nowadays, ranging from question-answering [1] through recommendation systems [2] to predicting drug–target interactions [3]. The ever-changing nature of information necessitates the design of a KG construction pipelines that are able to continuously incorporate new information. In developing such a system, knowledge engineering teams must address various challenges, from tackling scalability and heterogeneous data sources to tracking data provenance. Given the usually large volume of data that need to be integrated, such pipelines have to be automatized as much as possible while aiming at a high degree of data quality.

Knowledge graphs generally integrate heterogeneous data from a variety of sources with unstructured and semi-structured data of different modalities (e.g., pictures, audio, text) as well as structured data such as databases or other KGs in a semantically rich way. Therefore, constructing a KG encompasses a multidisciplinary effort requiring expertise from research areas such as natural language processing (NLP), data integration, knowledge representation, and knowledge management.

Knowledge graphs are at the center of numerous use cases for data analysis and decision support. In the clinical setting, enriching patient data with medical background knowledge enables improved clinical decision support [4]. Sonntag et al. [5] argue that properly aligning the semantic labels attached to patient data with medical ontologies is

crucial in creating meaningful access to heterogeneous patient data. Knowledge graphs are also used to organize the relevant information for fast-emerging global topics, such as pandemics (e.g., COVID-19) or natural disasters [6]. Machine learning also benefits from KGs as a source of labeled training data or other input data [7,8], thereby supporting the development of knowledge- and data-driven AI approaches [9]. KGs can further be combined with Large Language Models (LLMs) to improve factual correctness and explanations in question-answering, e.g., with ChatGPT, thereby promoting quality and interpretability of AI decision-making [10–12].

Improving the pipelines that build such Knowledge Graphs and enabling them to efficiently keep current and semantically meaningful aggregated knowledge is, therefore, an effort that benefits a wide range of application areas [13,14]. However, KGs are often created in a batch-like manner so that the respective pipelines are unfit to incorporate new incoming facts into a KG without the full re-computation of the individual tasks. Furthermore, different pipeline steps often require manual intervention, thereby limiting the scalability to large data volumes and increasing the time required to update a KG.

There are a growing number of surveys about Knowledge Graphs, especially on their general characteristics and usage forms [7–9]. An excellent tutorial-style overview of the construction and curation of KGs is provided in [15] with a focus on integrating data from textual and semi-structured data sources such as Wikipedia. Other surveys focus on KG construction with specific technologies [16,17] or only a single domain such as for geographical data [18]. We discuss related surveys on KG construction in Section 7 and contrast them with our approach.

This survey article provides a concise yet comprehensive entry into the current state of the art in KG construction for readers new to the topic, as well as contributing guidance for researchers, engineers, and experts by highlighting existing solution approaches and tools and identifying open gaps in the areas. We first outline the main requirements for the construction and continuous maintenance of KGs distilled from the literature, as well as our experience and reasoning. Next, we give an overview of the concrete subtasks of KG construction and current solution approaches. Furthermore, we select 27 KG-specific construction approaches, as well as generic toolsets based on the criteria discussed in Section 5 and evaluate and compare them with respect to the requirements introduced in Section 3. Finally, we identify open challenges and current limitations and, thus, areas for further research.

Our survey builds on previous studies for KG construction but differs in essential aspects, as will be explained in detail in Section 7. In contrast with most other surveys, we explicitly specify the main requirements for KG construction and use these as a guideline for evaluating current solutions and identifying open challenges. We are also more comprehensive in several important aspects as we cover different graph data models (RDF and property graphs) and deal with incremental KG construction and data integration, including incremental entity resolution in much more detail. We also provide a comparison between many carefully selected KG-specific construction approaches and toolsets and identify open challenges that go beyond those discussed in previous surveys.

The structure of the remainder of this survey is as follows: Section 2 deals with KG aspects and includes a definition of KGs, a discussion of graph data models for KGs, the distinction between domain-specific and general KGs, with the impact of the final use case on the construction methods, and the relevancy of dynamic adaptations in the pipeline. Furthermore, In Section 3, we introduce and categorize the general requirements for incremental KG construction. In Section 4, we provide an overview of the main tasks in incremental KG construction pipelines and proposed solution approaches for them. We then investigate and compare existing construction efforts for selected KGs as well as within recent tools for KG construction with respect to the requirements introduced earlier. This also allows us to identify tasks that are not yet supported well. Section 6 discusses open challenges for KG construction. Section 7 contains a more in-depth comparison with the related work. Finally, we provide concluding remarks and a summary.

2. Prerequisites

We first outline the notion of Knowledge Graph (KG) used in this paper and differentiate between domain-specific and general KGs. We then briefly introduce and compare the two most popular graph data models for KGs, namely RDF and property graphs. Finally, we outline the main requirements or desiderata for largely automatic construction and maintenance of KGs.

2.1. Knowledge Graph

KGs typically realize physical data integration, where the information from different sources is combined in a new graph-like representation (while we will focus on the predominant physical data integration of KGs, there are also some virtual data integration approaches, e.g., to keep data sources more autonomous [19,20]). KGs are schema-flexible and can thus easily accommodate and interlink heterogeneously structured entities. This is in contrast to the use of data warehouses as a popular approach to physical data integration. Data warehouses focus on integrating data within a structured (relational) database with a relatively static schema optimized for certain multi-dimensional data analyses. Schema evolution is a manual and tedious process, making it difficult to add new data sources or new kinds of information not conforming to the schema. KGs are less restricted and can better deal with heterogeneous information derived from semi- and unstructured data from potentially many sources.

Although the term *Knowledge Graph* goes back as far as 1973 [21], it gained popularity through the 2012 blog post (https://blog.google/products/search/introducing-knowledge-graph-things-not/ (accessed on 15 August 2024)) about the Google KG. Afterward, several related definitions of Knowledge Graphs were proposed, either in research papers [8,15,22–24] or by companies using or supporting KGs (OpenLink, Ontotext, Neo4J, TopQuadrant, Amazon, Diffbot (https://blog.diffbot.com/knowledge-graph-glossary/ (accessed on 15 August 2024)), Google). Ehrlinger et al. [23] give a comprehensive overview of KG definitions and provide their own: "A Knowledge Graph acquires and integrates information into an ontology and applies a reasoner to derive new knowledge". Hogan et al. [25] argue that this definition is very specific and excludes various industrial KGs that helped popularize the concept.

We, therefore, define KGs more inclusively as a graph of data consisting of semantically described entities and relations of different types that are integrated from different sources. Entities have unique identifiers. KG entities and relations can be semantically described by an ontology [26]. A KG's ontology defines the concepts, relationships, and rules governing the *semantic structure* within a KG of one or several domains that also include the types and properties of entities and their relationships. To structure data in a KG, common ontology relationships such as *is-a* and *has-a* are used to represent taxonomic hierarchies and possessive relations between entities.

Furthermore, a KG ontology combined with reasoning engines can infer new implicit knowledge from the explicitly represented information in the KG [27–29]. These sources cover hidden connections or knowledge that can be logically deduced from the given data

Figure 1 visualizes a simplified KG example with integrated information from several domains where ontological information such as types or is-a relations are dashed. There are ten entities of the following eight types: Country (*Ireland*), City (*Limerick*), Artist (*Aphex Twin*), Album (*Selected Ambient Works 85-9*), Record Label (*R & S*), Genre (*Techno*, *Ambient Techno*), Song (*Xtal*, *Ageispolis*), and Year (*1992*). Ontological *is-a* (sub-class) relations interrelate City and Country with Place, Artist with Person, Album with Music Release Type, and Record Label with Organisation. The domain is further described by the named relationships: country, birthPlace, artist, label, writtenBy, yearReleased, founded, broader, genre, yearProduced, partOf. Based on the given relationships and typing, further information is inferable. For example, Aphex Twin's broader birthplace is Ireland, the song Xtal is also of the genre Techno, and Aphex Twin being of the type Artist means this instance is also of the type Person (for readability, not all possible inferences are denoted).

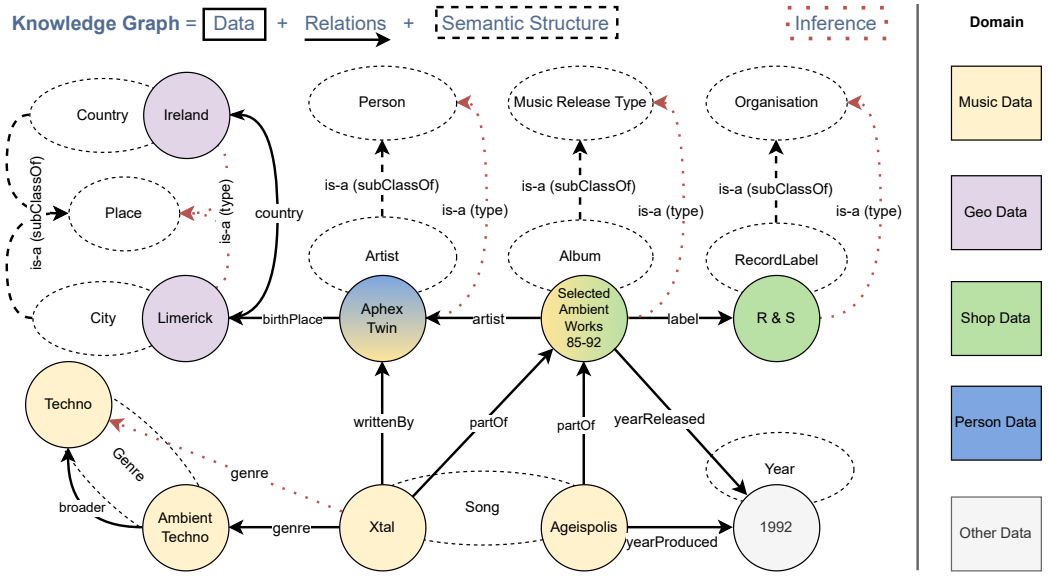

Figure 1. Simplified Knowledge Graph (KG) example demonstrating integrated information from five domains, showcasing ten entities of eight types connected by twelve relationships (two distinct is-a relations). Dashed lines indicate semantic structures (ontology or graph schema), such as entity types. Inferences can be made based on the relationships and typing, revealing additional information such as the broader birthplace of Aphex Twin being Ireland and Xtal belonging to the Techno genre (Not all possible inferences are shown for clarity).

2.2. Importance of Application Domain and Use Cases

KGs can vary significantly in their domain scope, ranging from those encompassing a broad spectrum of general, open-world knowledge, integrating insights from multiple disciplines, to specialized *domain-specific KGs* that focus on the nuanced details of a specific application domain or area of interest [13,30]. This diversity in scope reflects the varied applications of Knowledge Graphs and the importance of tailoring the ontology to meet the specific needs of its intended use case. Domain-specific KGs use specialized ontology terminology, often curated by experts in the targeted domain, whereas general KGs provide a broader view and connect multiple domains with a typically shallow representation of the individual fields. Use cases of open-world KGs include the building of search engines [31], general question-answering [32], or recommendation systems [33]. Use cases of domain-specific KGs range from healthcare [34,35] and medical diagnosis support [36], financial analysis and risk management [37,38] to customer insights [39] and products [40].

Different use cases require distinct KG requirements, influencing its design and construction to meet specific needs [30]. A KG's purpose significantly impacts the applied methods and steps during construction, with domain-specific data necessitating tailored approaches for effective integration. For instance, a medical KG employs different methods and tools compared to a geographical or financial KG. For example, in the biomedical domain, specialized machine learning models are designed to extract entities, such as gene names, protein interactions, or medical terminologies, from scientific publications [41]. Similarly, entity-matching algorithms can be customized for domain-specific data [42], such as geographical entities [43,44]. This targeted approach ensures that the KG is optimized for its intended purpose, providing more accurate and relevant insights.

Further, when constructing KGs, it is essential to consider the specific data requirements of the intended methods or applications. For example, temporal graphs are essential for incremental recommendations [45], tracking event types and user interactions, medical diagnoses and treatments [34], and fraud-detection systems relying on real-time transaction data and user behavior patterns [38,46]. Other applications may require including metadata such as entity origin [47] or spatial information [18]. These examples highlight how specific requirements influence the design and data composition of knowledge graphs.

Understanding the targeted application domain and intended use cases is crucial in constructing a Knowledge Graph. This understanding guides the selection of appropriate methods and tools, ensuring the KG meets the specific needs and requirements of its intended application. A tailored approach not only enhances the effectiveness of the KG but also maximizes its utility across various applications.

2.3. Graph Models

To represent and use KGs as informally defined above, a powerful graph data model is needed that supports entities and relations of different types, as well as their ontological description and organization [48]. Moreover, the graph data model should provide a comprehensive query language and possibly more advanced graph analyses or mining capabilities, e.g., for clustering of similar entities or determining graph embeddings for machine learning tasks. Support for integrity constraints is also desirable to automatically control the consistency and, therefore, quality of graph data to some extent. Moreover, the model should allow for the annotation of metadata for KG entities, such as information about their origin and transformations during KG construction. It should also support temporal analysis, enabling the representation of the KG's evolution over time. This can be achieved through a temporal graph data model that includes temporal metadata for each entity and relation, along with temporal querying capabilities to determine previous states of the KG or identify changes within specific time intervals. The temporal development of a KG might alternatively be reflected in a versioning concept where new KG versions are periodically released. Finally, the graph data model should facilitate the KG construction process and its different tasks for acquiring, transforming, and integrating heterogeneous data from different sources. This can be supported by suitable formats to seamlessly exchange data of the chosen graph model between different steps and processing nodes of a KG construction pipeline.

The most common graph models used for KGs are the Resource Description Framework and the Property Graph Model. Both offer pros and cons for specific use cases. In the following, we briefly describe both and discuss how they meet the introduced desiderata. Table 1 summarizes some of the key differences between the two models. At the end, we also contrast the different terminology of the models and specify the terms used in the rest of this paper.

Table 1. Comparison of RDF and property graphs.

	Resource Description Framework (RDF)	**Property Graph Model (PGM)**
base constructs	triples <subject, predicate, object>	labeled vertices and edges and their properties
entity identity	IRI-based	local (implementation-specific)
node classification	rdf:type triples	type labels
ontology support	RDFS, OWL2 vocabularies	limited, e.g., schema graph
reasoning/inference	supported, RDFS/OWL-based and other languages	limited, custom queries and procedures
integrity constraints	SHACL, SHEX	PG-Keys, PG-Schema
query language	SPARQL(-Star)	Cypher, Gremlin, G-Core, PGQL
exchange format	N-Triples, N-Quads, (RDF/XML, JSONLD)	application specific e.g., PGEF, GDL
meta information	reification, singleton-property, (RDF-Star)	dedicated properties

Resource Description Framework (RDF) is a framework or data model to present data in a graph-like fashion and was initially developed to describe metadata of web resources (namely the Semantic Web) [49]. Today, the W3C proposes many technologies around RDF that help build and use Knowledge Graphs either as part of the Linked Data Cloud or in an encapsulated environment. KGs are represented by a set of <subject, predicate, object> triples that uniformly represent named relations (predicates) of entities (subjects) to either attribute values (literals) or other entities (objects). Entities are usually assigned an IRI (Internationalized Resource Identifier) that can refer to either a global or local namespace. In addition to IRIs, entities can be identified by a blank node identifier

that is only unique inside an RDF dataset. Sets of triples may also be grouped within named graphs to aggregate more information by extending the triple structure to quads of the form <subject, predicate, object, named-graph>. Standard RDF does not support edge properties, although the RDF-Star extension (https://www.w3.org/2022/08/rdf-star-wg-charter/ (accessed on 15 August 2024)) includes a similar feature, where single triples are usable in the subject or object part of another triple.

The RDF standard also defines vocabularies such as the RDF Schema (RDFS) to further express semantic structure by allowing the definition of classes, properties, and their hierarchies. An RDF resource's type (or entity class) is assigned by using the standard RDF vocabulary (http://www.w3.org/1999/02/22-rdf-syntax-ns# (accessed on 15 August 2024)) to define triples of the form `<s rdf:type o>`, where o is the class (type) of the resource. In addition to RDFS, a widely used approach for defining ontologies is the Web Ontology Language (OWL), more specifically, the current version OWL 2, which adds semantics to the data using a variety of axioms.

RDF KGs using vocabularies like RDFS, OWL2 and SWRL [50], when used alongside reasoning engines [51–54], enable the deduction of new knowledge by inferring implicit relationships from explicitly stated data [27–29]. Further, reasoning engines allow for consistency checking and classification. Custom rule languages like Datalog [55] and SPIN (SPARQL Inferencing Notation) enable rule-based inferencing in RDF data, with SPIN specifically using SPARQL queries for flexible and dynamic reasoning.

Besides syntax validation (triples/quads, URIs, datatypes) RDF triple stores do not provide a standard method to define and validate graph data integrity or shape constraints (similar to the relation database schemata). Therefore, overlaid solutions such as SHACL (Shape Constraint Language [56]) or ShEx (Shape Expressions [57]) are developed, which can be used to validate the semantic correctness of the graph structure, node or property constraints, cardinalities, and other constructs.

While some RDF stores or triple stores are built from scratch to optimize the management of RDF triples, others might use existing SQL or NoSQL systems in the underlying database processing layer. The primary query language for RDF (moreover, the Semantic Web) is the standardized language SPARQL (https://www.w3.org/TR/sparql11-overview/ (accessed on 15 August 2024)), with an extended version for RDF-Star called SPARQL-Star. Standard exchange formats for RDF are N-Triples, N-Quads, Turtle, or adapted syntax formats like RDF/XML and JSON-LD.

Different methods exist for assigning metadata to entities, relations, and properties, such as using RDF-Star or named graphs, as discussed and evaluated in [58,59]. However, incorporating support constructs for metadata management generally increases the complexity of the graph structure and queries, which can potentially lead to increased processing time.

There is some work around the representation, querying storage, and other aspects of temporal information in RDF [60]. The investigated methods focus on different temporal granularity and dimensions, including approaches that target querying single snapshots and time windows or inspect the evolution of temporal graphs.

As many Knowledge Graphs are in RDF, several frameworks have been developed to perform graph analytics, algorithms, or mining tasks using RDF as input [61].

Property Graph Model (PGM) [62]. The property graph data model, also called Labeled Property Graph (LPG), supports the flexible definition of graph structures with heterogeneous nodes (vertices) and directed edges to represent entities of different kinds and the relationships between them. Both nodes and edges can have multiple (type) labels expressing their role in the modeled domain, e.g., *User* as a node label and *follows* as an edge label. Additionally, properties (in the form of key–value pairs) can be assigned to both nodes and edges. Further, in the most common implementations, vertices and edges are specified by a unique identifier. While label and property names represent some schema-like information, there is intentionally no predefined schema to allow the flexible incorporation of heterogeneous entities and relations of different kinds (although a schema

graph can be inferred from the type information [63]). There is no built-in support for ontologies, e.g., to provide is-a relations between entity categories. Embedded metadata can be relatively easily maintained for entities and relationships by using dedicated properties, e.g., for provenance or time annotations.

In contrast to RDF, the PGM, with its vertices, edges, and properties, is more related to graph models in graph theory, thereby contributing to their good understandability. As there is not yet a global (defacto) standard for PGM, its capabilities highly depend on its implementation. The PGM is increasingly popular in research and practice and supported by several graph database systems, such as Neo4j [64], JanusGraph [65] or TigerGraph [66], and processing frameworks, such as Oracle Labs PGX [67] or Gradoop [68]. It is further the base data model for several graph query languages [69], such as G-Core [70], Gremlin [71], PGQL [72], and Cypher [73], as well as SQL/PGQ and GQL [74], the upcoming ISO standard language for property graph querying. Efforts on a standardized PGM serialization format comprise the JSON-based Property Graph Exchange Format (PGEF) [75], YARS-PG [76], and the Graph Definition Language (GDL) (https://github.com/dbs-leipzig/gdl (accessed on 15 August 2024)).

While most PGM engines do not natively support reasoning in the way RDF systems do with OWL ontologies, they can perform pattern matching, traversal-based reasoning, and inferencing through custom queries and procedures. In Neo4J, the neosemantics plugin (https://neo4j.com/labs/neosemantics/4.0/inference/ (accessed on 15 August 2024)) offers RDF-like reasoning capabilities and an extension to Janusgraph allows reasoning in the Gremlin query language [77].

Similar to RDF stores, the data integrity of PGM databases is generally limited to syntax or basic value constraints. A first effort about the aspects of property graph *key constraints* is proposed by Angles et al. [78] by identifying four natural key types: identifier, exclusive mandatory, exclusive singleton, or exclusive. Additionally, PG-Schema offers a robust formalism for specifying property graph schemas [79].

There are several extensions to the PGM for supporting temporally evolving graph data [68,80] and graph streams [81], often with advanced analysis capabilities for graph mining.

Discussion. The intensive use of RDF in the Semantic Web and Linked Open Data communities has led to its widespread application for KGs; in fact, most KGs we will consider in Section 5 use RDF. The triple-based graph representation of RDF is quite flexible and allows a uniform representation of entities and relationships. But it is also hard to understand without additional processing or inference as the information of an entity is distributed over many triples. While RDF-Star greatly improves the formal meta-expressiveness of RDF, specific cases are still not presentable as in PGM without utilizing support constructs. In the PGM, we can have two relations with the same name that can be addressed independently. Each relation has its own distinct properties. However, in RDF-Star, relations (triples) are identified based on their associated elements <<s1,p1,o1>,p2,o2>, and it is not possible to attach different sets of information to equally named relations (triples) without causing incorrect connections or relying on support constructs (e.g., singleton properties) [82]. While RDF is older and has gone through extensive standardization during the last 25 years, the PGM has become increasingly popular for advanced database and network applications, such as graph traversal and network analysis [83].

Besides RDF (direct graphs) or property graphs, in some cases, custom models or special high-arity representation could be used to cover specific features, such as access levels, temporal information, or multihop relations in one record (node–edge–edge–node) [84]. However, the usage of such custom models will lower interoperability with existing tools (requiring transformation) and complicate its own reusability by others.

The decision between RDF or PGM (or a custom data model) depends on the targeted application or use case of the final Knowledge Graph. Lassila et al. [82] conclude that both formats are qualified to meet their challenges, and neither of the two is perfect for every use case. They thus recommend increasing interoperability between both models to reuse existing techniques of both approaches. Various efforts to address this problem have been

made in recent years. The Amazon Neptune (https://aws.amazon.com/en/blogs/aws/amazon-neptune-a-fully-managed-graph-database-service/ (accessed on 15 August 2024)) database service allows users to operate PGM and RDF interchangeably. Hartig et al. [85] and Abuoda et al. [86] discuss transformation strategies between RDF and PGM to lower usage boundaries. GraphQL (https://graphql.org/ (accessed on 15 August 2024)) provides a unified approach to query both RDF and the PGM, although with fewer features compared to query languages dedicated to these graph formats. GraphQL-LD [87] aims at simplifying querying Linked Data via GraphQL.

Terminology

Due to the different communities around PGM and RDF, many similar but differently named terms are used. Table 2 lists some of the terms that we will use synonymously in this paper with the underlined ones used preferably. Furthermore, we refer to the smallest unit of information as *statement* or *fact*. For RDFm this would describe a triple; for PGM, this can be assigning a property (value), adding a type label to an entity, or adding a relation between two nodes.

Table 2. Synonymously used KG terms in RDF and PGM.

Terms	Description
entity, instance, subject and object and resource (RDF), individual	KG nodes that represent a specific real-world or abstract thing
relation, property (RDF)	A relationship (edge, link) between two KG entities.
type, class, label, concept	Identifier that represents the same kind or group of entities or relations.
property (PGM), attribute (RDF)	An entity feature identifier pointing to a value
property value, literal, attribute value	Any value that is not referable to as an entity.

2.4. Dynamic Adaptations in Knowledge Graph Construction

Understanding the dynamic nature of Knowledge Graph (KG) construction is crucial for developing robust and scalable systems. This section focuses on the potential changes that can occur in data sources, target domains, and integration methods over time and emphasizes the interplay and dependencies among these changes. For example, the introduction of new data sources might necessitate changes in integration methods, and shifts in target domains could require updates to both data sources and integration techniques.

- **Source Changes.** Data can be updated continuously or regularly. Sources, like relational databases, can receive regular updates, enriching their content with new entries and revisions. Platforms like Wikipedia and various websites continually expand their repositories with fresh information. At the same time, new forms of social or multimedia, such as text, image, and video posts, contribute to the vast amount of continuously populated data. Beyond traditional datasets, new sources emerge with different structures, access methods, and specialized domains. These changes broaden the scope of available information and require fitting methods for integrating diverse and evolving sources into knowledge graphs.
- **Target Changes.** The domain or ontology is often precipitated by shifts in organizational priorities, advancements in knowledge within a particular field, or emerging trends in application requirements. These changes can be driven by evolving business strategies that necessitate a reevaluation of the scope and focus of the domain. External factors such as regulatory changes or market dynamics shifts can also influence ontology adjustments, ensuring alignment with current standards and practices. Ultimately, the dynamic nature of the target domain or ontology reflects an ongoing effort to adapt and refine knowledge representation to best serve the evolving needs of stakeholders and the broader environment in which the system operates.
  For example, initially centered on company data, Knowledge Graphs may later incorporate geographical information or delve into other specialized domains. Each

new source brings unique opportunities and complexities, demanding flexible approaches to integration that can accommodate diverse data structures and formats. The evolution of data sources necessitates corresponding changes in the final data representation and ontology within Knowledge Graphs. As the scope of domains shifts, adjustments to the ontology become imperative to accurately reflect the underlying data. Ontologies can range from predefined structures that provide a consistent framework to flexible frameworks that evolve semi-automatically based on incoming data. This adaptability ensures that Knowledge Graphs remain relevant and effective in capturing and organizing the intricacies of evolving datasets.

- **Method Changes.** Integrating heterogeneous data sources into a Knowledge Graph involves a series of critical steps: extraction, resolution, fusion, completion, and quality assurance. Changes in these integration steps are essential to accommodate updates in both source data and the target domain Knowledge Graph to improve the KG quality. Methods may need to be adjusted or augmented to handle new data sources effectively. Specific steps may be introduced to enhance the integration process, ensuring that the Knowledge Graph remains robust and up-to-date amidst evolving data landscapes. For instance, if the domain of the Knowledge Graph shifts to a specialized area like biomedical data, it might be necessary to incorporate a different method or tool for text extraction. In the biomedical field, specific techniques are often required to extract entities accurately, such as gene names, protein interactions, or medical terminologies. Additionally, as advancements in machine learning continue, updating the integration pipeline with more powerful algorithms can significantly enhance the quality and efficiency of the Knowledge Graph. For example, a new deep-learning model for entity recognition in biomedical texts could replace an older model to achieve better precision and recall.

By understanding and managing these dynamic adaptations, KGs can remain relevant, accurate, and effective over time. This section sets the stage for understanding the essential dynamic adaptations, while the following section provides requirements on data processing methods to handle these changes effectively.

3. Requirements

The development and maintenance of KGs encompass several steps to integrate relevant input data from different sources. While the specific steps depend on the input data to be integrated and the intended usage forms of the KG, it is generally desirable that the steps are executed within pipelines with only a minimum of manual interaction and curation. However, a completely automatic KG construction is not yet in reach since several steps (e.g., identification of relevant sources, development of the KG ontology), as we will see, typically require human input by individuals, expert groups, or entire communities [88].

The KG construction process should result in a high-quality KG based on an expressive *KG data model*, as discussed above. The quality of a KG (and data sources) can be measured along several dimensions such as correctness, freshness, comprehensiveness, and succinctness [89,90]. The correctness aspect is crucial to the validity of information (accuracy) and implies that each entity, concept, relation, and property is *canonicalized* by having a unique identifier and being included exactly once (consistency) [15]. The freshness (timeliness) aspect requires continuously updating the instances and ontological information in a KG to incorporate all relevant changes in data source. The comprehensiveness requirement asks for good coverage of all relevant data (completeness) and that complementary data from different sources are combined [90]. Finally, the succinctness criterion asks for a high focus in the data (e.g., on a single domain) [8] and the exclusion of unnecessary information, which also improves resource consumption and scalability of the system (availability). A Knowledge Graph that meets high standards in these areas can be considered a confident and reliable resource (trustworthiness) [91].

Below, we discuss requirements for KG construction and maintenance in more detail, as they should guide the realization of suitable implementation approaches. We group these requirements into four aspects related to (1) input consumption, (2) incremental data processing capabilities, (3) tooling/pipelining, and (4) quality aspects, whereas some essential prerequisites can affect multiple parts of the workflow (e.g., supportive metadata). Please note that we outline the desired functionality for defining arbitrary KG pipelines and that, depending on its purpose, only a subset of it is typically needed for a specific KG project.

3.1. Input Data Requirements

Integrating a large number of data sources and a high amount of data (data scalability) should be possible. There should also be support for heterogeneous and potentially low-quality input data of different kinds, such as structured, semi-structured, and multimodal unstructured data (textual documents, web data, images, videos, etc.). As a result, KG construction requires scalable methods for the acquisition, transformation, and integration of these diverse kinds of input data. The processing of semi-structured and unstructured data introduces the need for Knowledge Extraction methods to determine structured entities and their relations, as well as their transformation into the KG graph data model. Data integration and canonicalization involve methods to determine corresponding or matching entities (entity linking, entity resolution) and their combination into a single representation (entity fusion), as well as matching and merging ontology concepts and properties. For incremental KG construction, the input is not limited to the new data to be added but also includes the current version of the KG and reusable data artifacts, such as previously determined mappings specifying how to transform input data into the format of the KG graph model.

3.2. Support for Incremental KG Updates

It should be possible to process the input data in a batch-like mode, where all (new) input data are processed at the same time, or in a streaming manner, where new data items can continuously be ingested. The initial version of the KG is typically created in a batch-like manner, e.g., by transforming a single data source or by integrating several data sources into an initial KG. After the initial KG version has been established, it is necessary that the KG be updated to incorporate additional sources and information. A simple approach would perform these updates by a complete recomputation of the KG with the changed input data, similar to the creation of the initial KG. However, such an approach would result in an enormous amount of redundant computation to repeatedly extract and transform the same (unchanged) data and to perform data integration and the removal of inconsistencies again, possibly with repeated manual interactions. These problems increase with the number and size of input sources and thus limit or prevent scalability. Hence, we require support for incremental KG updates that can either periodically be performed in a batch-like manner or in a more dynamic, streaming-like fashion. The batch approach would not require completely rebuilding the KG but focus on adding the new information without reprocessing previously integrated data. A given KG can also be continuously updated with new data in a streaming manner to always provide the most current information for high data freshness. Batch- and stream-oriented updates may also be applied in combination [84]. As a result, several pipelines may be needed for the creation of the initial KG, the integration of sources with heterogeneous structures, and different forms of incremental KG maintenance. While incremental KG maintenance is important in general, specific KG use cases, such as research projects, may only need a one-time or batch creation of a KG. Hence, the posed requirement would not apply in such cases.

3.3. Pipeline and Tool Requirements

It should be easy to define and run powerful, efficient, and scalable pipelines for creating and incrementally updating a KG. This requires a set of suitable methods or tools

for the different steps (discussed in the next section) that should have good interoperability and a good degree of automation but still support high customizability and adapt to new domain requirements. While using a uniform KG data model (or serialization) can lower the workflow's debugging complexity, reusing existing toolsets might require transformation/mapping between data formats and processing steps. Moreover, a pipeline tool should be provided that can integrate the different tools and manage intermediate results and common metadata, e.g., about provenance. The pipeline tool should further provide administration functionality to design and execute pipelines, support error handling, performance monitoring and tuning, etc. Pipeline processes should scale horizontally as new input data are ingested and the KG's size increases over time. Modular processing workflows with transparent interfaces can increase the reusability of alternative tools (implementations).

3.4. KG Quality Requirements

Quality assurance is a cross-cutting topic that plays an important role throughout the whole KG construction process. Quality problems in the KG can be multifaceted and relate to ontological consistency, the data quality of entities and relations (comprehensiveness), or domain coverage. The coverage aspect may focus on the inclusion of relevant data and the exclusion of unnecessary data. In some scenarios, the timeliness of data can play a critical role in real-time-oriented use cases. If not handled, quality problems might aggravate over time due to the continuous integration of additional data. Therefore, methods are needed to evaluate the quality of each step of the construction pipeline and of the resulting KG. A specific quality aspect is to validate the KG's data integrity concerning its underlying semantic structure (ontology). Another relevant criterion could be to optimize data freshness to guarantee up-to-date results in upstream applications. Debugging capabilities based on sufficient metadata are helpful in locating the exact points in the construction pipeline where quality problems arise. Methods are then required for fixing or mitigating the detected quality issues by refining and repairing the KG.

4. Construction Tasks

We give an overview of the main tasks for Knowledge Graph construction with a focus on (semi-)automatic and incremental solutions. In particular, we cover the following tasks that often involve several subtasks:

- Metadata Management: The acquisition and management of different kinds of metadata, e.g., about the provenance of entities, structural metadata, temporal information, quality reports, or process logs.
- Data Acquisition and Preprocessing: The selection of relevant sources, acquisition, and transformation of relevant source data, and initial data cleaning.
- Ontology Management: The creation and incremental evolution of a KG ontology.
- Knowledge Extraction (KE): The derivation of structured information and knowledge from unstructured or semi-structured data using techniques for named entity recognition, entity linking, and relation extraction. If necessary, this also entails canonicalizing entity and relation identifiers.
- Entity Resolution (ER) and Fusion: Identification of matching entities and their fusion within the KG.
- Knowledge Completion: Extending a given KG, e.g., by learning missing type information, predicting new relations, and enhancing domain-specific data (polishing).
- Quality Assurance (QA): Possible quality aspects, their identification, and repair strategies for data quality problems in the KG.

Figure 2 illustrates a generic pipeline to incrementally incorporate updates from several sources into a KG that may result in a sequence of distinct KG versions. It is important to note that a construction pipeline does not necessarily follow a fixed execution order for the individual tasks and that not all steps may be required depending on the KG use case. This is also because the required tasks depend on the type of source input. Knowledge extraction is commonly applied to unstructured data inputs like text and

may not be needed for structured data, e.g., from databases or other Knowledge Graphs. Furthermore, the entity linking part of Knowledge Extraction can make an additional entity resolution step unnecessary. As a result, there may be different KG construction pipelines for different use cases and data sources. The steps of Quality Assurance and KG completion to improve the current version of the KG are not needed for every KG update but may be executed asynchronously, e.g., within separate pipelines (although QA actions such as data cleaning also apply to individual tasks). Furthermore, data and metadata management play a special role compared to the other tasks since they are necessary throughout the entire pipeline, therefore representing a cross-cutting task, as indicated by the central position of metadata management in Figure 2.

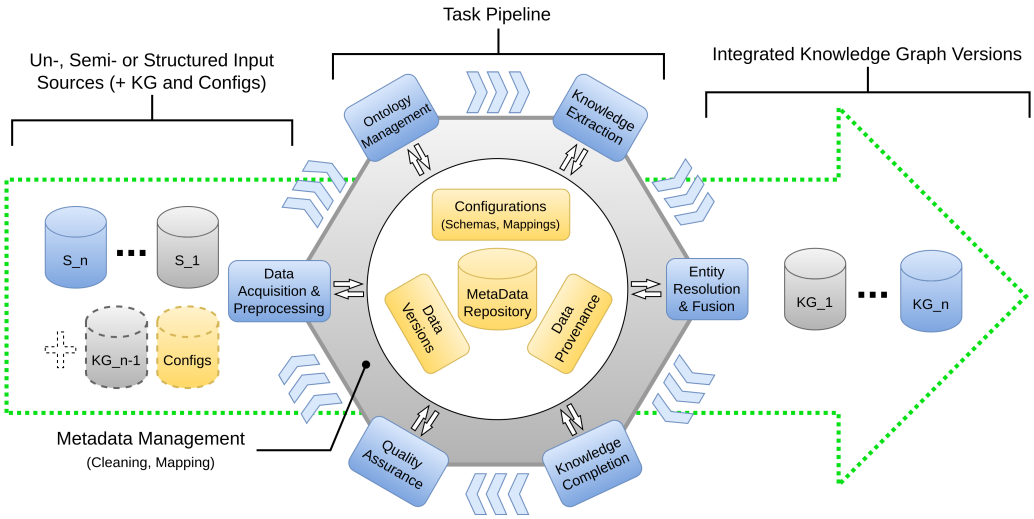

Figure 2. Incremental knowledge graph construction pipeline.

In the following overviews of the different tasks, we will discuss the major solution approaches. We will also refer to some of the recent methods in ML, including Large Language Models (LLMs) that show promise in some relevant areas, such as for data integration [92,93]. However, the use of LLMs for KG construction is still at an early stage and a topic for further research, as discussed in Section 6.

4.1. Metadata Management

Metadata describe data artifacts and are important for the findability, accessibility, interoperability, and (re-)usability of these artifacts [18,94,95]. There are many kinds of metadata in KGs, such as descriptive metadata (content information for discovery), structural metadata (e.g., schemas and ontologies), and administrative metadata concerning technical and process aspects (e.g., provenance information and mapping specifications) [96–98]. It is thus important that KG construction supports the comprehensive representation, management, and usability of the different kinds of metadata. From the perspective of KG construction pipelines, this includes metadata for each data source (schema, access specifications), each processing step in the pipeline (inputs including configuration and outputs including log files and reports), intermediate results, and, of course, the KG and its versions. Moreover, for each fact (entity, relation, property) in the KG, there can be metadata such as about provenance, i.e., information about the origin of data artifacts. Such fact-level provenance is sometimes called *deep* or *statement-level provenance*. Examples of deep provenance include information about the creation date, confidence score (of the extraction method), or the original text paragraph from which the fact was derived. Such provenance can help to make fact-level changes in the KG without re-computing each step or to identify how and from where wrong values were introduced into the KG [95].

Metadata can be created either manually by human users (e.g., to specify a license for data usage or a configuration of a pipeline step) or by a computer program based on a

heuristic or an algorithm [97]. In the latter case, the results may be exact or only approximate. For example, data profiling computes accurate statistical information (e.g., about the distribution of values), while the use of machine learning (e.g., for type recognition) usually does not provide perfect accuracy. Advancements in LLMs for data management have shown capabilities in extracting specific values for given documents or generating code snippets of functions to perform this kind of task [99,100].

4.1.1. Metadata Repositories

To use metadata effectively for KG construction, it is beneficial to maintain a metadata repository (MDR) to store and organize the different kinds of metadata in a uniform and consistent way [94]. Either the MDR can be separate from the sources and the KG with reference to data artifacts, or combined solutions can be provided for both the data and their metadata. While several metadata repositories may exist for the different sources and processing steps, a central solution can simplify access to all KG-relevant metadata. In contrast, using multiple metadata solutions might allow more flexibility in selecting specialized solutions that suit specific needs or types of metadata. This approach can also introduce complexity or inconsistencies and hinder the process of discovery and exploration due to information being scattered across various repositories. Specific implementations of MDRs are CKAN (https://ckan.org/ (accessed on 15 August 2024)), Samply [101], or the DBpedia Databus [102], all using specific vocabularies, standard query languages, and databases to implement their relevant features. Concerning metadata exchange, the Open Archives Protocol for Metadata Harvesting (https://www.openarchives.org/OAI/openarchivesprotocol.html (accessed on 15 August 2024)) framework also allows the acquisition of structured metadata.

4.1.2. Graph Embedded Metadata

Fact-level or annotation metadata in the KG can be stored either together with the data items (embedded metadata) or in parallel to the data and referenced using unique IDs (associated metadata) [98]. For example, fact-level metadata can support the selection of values and sub-graphs [103] or the compliance with licenses used in target applications. Such annotations are also useful for other kinds of metadata. Temporal KGs can be realized by temporal annotations to record the validity time interval (the period during which a fact was valid) and transaction time (the time when a fact was added or changed) [68,80]. The possible implementations for fact-level annotations depend on the used graph data model (see Section 2.3). From another perspective, provenance metadata can also capture the steps of the applied schema and data transformations in the pipeline [104,105].

While some RDF stores or triple stores are built from scratch to optimize the management of RDF triples, others might use existing SQL or NoSQL systems in the underlying database processing layer. The primary query language for RDF (moreover, the Semantic Web) is the standardized language SPARQL (https://www.w3.org/TR/sparql11-overview/ (accessed on 15 August 2024)), with an extended version for RDF-Star called SPARQL-Star. Standard exchange formats for RDF are N-Triples, N-Quads, Turtle, or adapted syntax formats like RDF/XML and JSON-LD. There are different possibilities to assign metadata to entities, relations, and properties, like using RDF-Star or named graphs as explained and evaluated in [58,59]. The usage of support constructs for metadata management generally increases the complexity of the graph structure and queries and can possibly increase processing time.

The PGM natively allows for adding properties to entities or relations, mostly as key–value pairs, which makes it easy to add additional metadata. In some implementations, the values of these fields can also contain nested data structures, such as JSON, allowing for expressing and querying complex metadata objects.

4.1.3. Versioning

Versioning is a highly relevant concept for incremental Knowledge Graph construction, as it helps to identify specific states and queries between versions of the KG after each change and incremental update. Previous works distinguish three different version concepts [106,107]:

Independent Copies in Knowledge Graph (KG) management captures the entire KG at regular intervals (daily, weekly, or monthly). Each snapshot represents the KG at a specific timestamp, providing clear versioning and aiding in audits and historical analysis. However, this approach can be resource-intensive, especially for large KGs with frequent updates, requiring significant storage capacity for multiple versions. Efficient storage management and retrieval strategies are crucial to reduce overhead.

A simple solution is to store each snapshot in a file system and keep version information separately [108]. Another option is to store the entire graph (edges and nodes or triples) as tables with an added column for the version [109].

Change-based or differential (diff) versioning captures and stores only the changes (additions, updates, deletions) between successive KG versions. This method saves storage by maintaining only deltas, with each new version derived from its predecessor through these changes. While storage-efficient, it requires robust mechanisms for computing and applying changes. As changes accumulate, retrieving specific versions can become complex, needing careful version history management and efficient querying strategies.

Example systems supporting change-based revisions are R43ples [110] and Stardog (http://stardog.com (accessed on 15 August 2024)) Further, QuitStore [111] uses the Git version control system, a triple-based diff algorithm, and fragmentation to incorporate version capabilities for collaborative RDF editing.

Timestamp-based solutions use additional metadata representation or native temporal graph models, where versioning information is integrated into the KG structure through annotations on entities, relationships, or the entire KG. Each part is tagged with version details (timestamps or version numbers), allowing detailed queries and analyses based on metadata. This approach requires careful management to maintain metadata consistency and may require additional processing to retrieve specific versions, balancing enhanced version control with the complexity of managing metadata integrity. Examples of implementations are Dydra [112] for RDF and the TPGM [80] model of Gradoop for PGM.

There is some work around the representation, querying storage, and other aspects of temporal information in RDF [60]. The investigated methods focus on different temporal granularity and dimensions, including approaches that target querying single snapshots and time windows or inspecting the evolution of temporal graphs. Similar work was performed for PGM to support temporal capabilities natively [68,113].

4.2. Data Preprocessing

Building a Knowledge Graph (KG) involves meticulous planning and strategic decision-making when selecting and integrating data sources. This section delves into the critical initial phases of KG construction: Source Selection and Filtering, Data Acquisition, Data Transformation, and Data Cleaning.

4.2.1. Source Selection and Filtering

In order to integrate data into a KG, relevant sources must first be identified. Furthermore, relevant subsets of a data source have to be determined as it is generally unnecessary to integrate all information of a source for a given KG project. For example, for a pandemic-specific KG, only health-related parts of existing KGs, such as DBpedia, may be needed. If the system is not supposed to integrate data sources in their entirety, it can determine a relevant subset by the quality or trustworthiness of a source [114], as well as the importance of single entities [84]. Formulating the quantification [115] of all these criteria alongside computing the cost of integrating a source leads to saving a considerable amount of unnecessary effort while producing a high-quality KG.

Selecting relevant data sources and their subsets is typically a manual step, but data catalogs describing metadata about sources and their contents can support this process. Common approaches to determine such metadata are to employ techniques for data profiling, topic modeling, keyword tagging, and categorization [116,117]. In [15], it is recommended to start KG construction with large curated ("premium") data sources such as Wikipedia and other KGs such as DBpedia. Then, further data sources should be identified and integrated to cover additional entities and their relations, especially rather special entities in the "long tail" (e.g., less prominent persons). Given that sources can differ enormously in size and quality, the order in which sources (and their updates) are integrated can have a strong influence on the final quality [118–120]. Especially for creating the initial KG, these choices are often crucial. To limit these effects, it is advisable to first integrate the sources of the highest quality [118] such as the mentioned premium sources. Nevertheless, ideally, the integration order should not matter in a high-quality pipeline.

4.2.2. Data Acquisition

A KG's data sources may be in many different data formats, such as CSV, XML, JSON, or RDF, to meet the requirements of different originating environments and applications. Furthermore, there are different technologies to exchange or acquire data artifacts by providing downloadable files, deploying databases, or individual application program interfaces (APIs). Hence, KG construction has to deal with these heterogeneous data formats and access technologies to acquire the data to be integrated. A common access approach is the use of an adapter component for each source dataset. Such an adapter approach is typical for data integration, and there are also supporting tools for use in KG management [121–123].

In addition, KG construction has to deal with continuously changing sources, which necessitates the recognition of such changes and possibly maintaining snapshots of already acquired versions of source data. Possible solutions for change detection include manual user notifications over email, accessing a change API using publish–subscribe protocols [124], or computing diffs by repeatedly crawling external data and comparing them with a previously obtained snapshot.

For RDF stores, several strategies for maintaining versions of extracted data have been proposed. With full materialization, complete versions (snapshots) of source data are maintained [125]. With the delta-based strategy, only one full version of the dataset needs to be stored, and for each new version, only the set of changes or deltas has to be kept [110,126,127]. The annotated triples strategy is based on the idea of augmenting each triple with its temporal validity [128]. Hybrid strategies have also been considered [129,130]. Another approach to synchronize changes in a data source is Linked Data Event Streams (LDESs) [131]. Van Assche et al. [132] use LDES to continuously update a KG with changes from the underlying data sources.

Other KGs are important sources for data acquisition. However, only a limited number of KGs provide a queryable interface, and such interfaces can be expensive to host at high availability [133–135]. To address this problem, recent proposals suggest decentralization, either of the data themselves or of the query processing tasks. Decentralization (distribution) of the data across multiple sources [136–138] can increase their availability but tends to provide less efficient query processing compared to centralized servers or approaches that provide full data dumps to powerful clients for local processing. Alternatively, recent studies [133,139–142] have focused on decentralizing the query-processing tasks by dividing the processing workload between servers and clients. WiseKG is a system to dynamically distribute the load between servers and clients based on a cost model [143].

4.2.3. Transformation and Mapping

A KG construction pipeline has to transform the input data into the final KG data format, such as RDF or a property graph format. Furthermore, the different pipeline steps may consume and produce different formats, so additional data format transformations or

conversions may become necessary. For example, knowledge-extraction methods typically process document data such as HTML or Unicode-encoded text, while an entity resolution task may require input data in CSV or JSON format.

Data transformations have especially been addressed for (semi-)structured data, and many tools exist for this purpose [144]. Depending on the required input format, the transformation can be performed automatically using generic approaches or requires the manual specification of mappings. Mapping languages allow the specification of complex and reusable mappings, for example, to transform relational databases (RDBs) into an equivalent RDF representation, e.g., using the R2RML language [145]. RML [146] (RDF Mapping Language) extends R2RML and allows defining mappings not only from RDB but also from other semi-structured data formats such as XML, TSV/CSV, and JSON. Systems implementing such mapping languages include SDM-RDFizer [147] for RML and Karma [148] for an R2RML alternative called K2RML. Relatively little work has so far investigated the transformation of structured data into property graphs [149,150], although the conversion between RDF and property graphs has received some attention [85,86,151]. In the case of an existing RDF-based KG as input, a simple solution for RDF to RDF mappings is to use SPARQL-CONSTRUCT (https://www.w3.org/TR/rdf-sparql-query/#construct (accessed on 15 August 2024)) queries, which return a single RDF graph by substituting variables of a given graph pattern with the results of the SPARQL query. As an extension of SPARQL, SPARQL-Generate supports the transformation of streaming and binary data sources [152]. The graph query language GQL will support a similar feature for the PGM [74]. An extensive survey on state-of-the-art RDF mapping languages for schema transformation, data transformation, and systems was conducted by Van Assche et al [20]. A major issue the authors point out is the lack of tools supporting the (semi-)automatic definition of mappings. In their survey, only 3 out of 30 analyzed systems support the semi-automatic definition of mappings (including human-in-the-loop methods) [153–155].

4.2.4. Data Cleaning

Data cleaning deals with detecting and removing errors and inconsistencies from data in order to improve the quality of data. Whenever possible, data quality problems within the input sources should be handled already during the import process to avoid wrong or low-quality data being added to the KG. Data cleaning has received a large amount of interest, especially for structured data, in both industry and research, and there are numerous surveys and books about the topic, e.g., [156–159].

Various types of data errors and quality problems, ranging from structural to semantic, need to be handled. Dealing with structural problems requires consolidating different data structures and formats and ensuring consistent naming conventions for entities and attributes. For instance, if "USA", "United States", and "U.S." are all used to represent the same country, they should be standardized into a single form. Semantic data cleaning focuses on addressing issues related to the meaning and relationships within the data. One example is handling conflicting or contradictory information present in the dataset. For instance, if one source indicates that a person was born in 1980 while another source suggests 1985, this inconsistency needs to be resolved. Another example is handling entities either in one source or different sources that represent the same real-world object, e.g., a certain customer or product.

Data cleaning typically involves several subtasks to address these problems. These include data profiling to identify quality issues [160], data repair to correct identified problems, data transformation to standardize data representations, and data deduplication to eliminate duplicate entities. Outlier detection is an important aspect of data profiling, aiming to identify data errors based on the assumption of specific "normal" data values. For instance, it is highly unlikely that an individual born in the mid-19th century would still be alive in the year 2020.

Rule-based methods are classic techniques used for data cleaning. These methods handle errors that violate integrity constraint rules, such as functional dependencies

(FDs) [46,161–164], conditional functional dependencies (CFDs) [163,165–167], and denial constraints (DCs) [163,167–171]. While rule-based methods can handle data that violate predefined rules, their effectiveness is limited by the challenge of obtaining sufficient and correct rules. Statistical cleaning methods repair errors based on probabilistic distributions within the data [46,169,171–174].

User interaction cleaning methods involve human knowledge to enhance the quality of cleaning results while minimizing the effort required [172,173,175–180]. The use of machine learning for data cleaning has gained prominence in recent years, as it simplifies the configuration of various subtasks. For example, HoloClean [171] employs observed data to build a probabilistic model for predicting unknown data values. Sudowoodo [181] uses contrastive learning to train a representation model that matches cells with possible corrections, enabling data cleansing and error correction. Other applications of machine learning for data cleaning are covered in [158,182].

If there is already a KG version to be extended, the KG information can be leveraged to identify and handle data errors. For instance, KATARA [176] employs crowdsourcing to verify whether values that do not match the KG are correct or not. Hao et al. [183] introduce detective rules (DRs) that can make actionable decisions on relational data by building connections between a relation and a KG. KGClean [184] is an initial attempt at a KG-driven cleaning framework utilizing Knowledge Graph embeddings.

LLMs can be effective for data imputation and error detection, utilizing prompts to infer missing data points or identify errors in attribute–value pairs [92].

Techniques for ensuring KG quality are discussed in Section 4.7. Approaches for identifying duplicates across data sources (entity matching) are outlined in Section 4.5. It is advantageous to remove duplicates within a source early on to simplify the deduplication process across sources.

4.3. Ontology Management

A KG's ontology defines concepts, relationships, and rules for the semantic structure within domains, using relationships like *is-a* and *has-a* to represent hierarchies and relations, enabling inference of new knowledge. RDF uses vocabularies like RDFS and OWL 2 to define classes, properties, hierarchies, and rules for semantic expression. In the Property Graph Model (PGM), ontologies are implemented as an overlay with nodes, relationships, labels, properties, and rules, utilizing tools like APOC in Neo4J for advanced functionalities

Ontology Management is the incremental process of creating, extending, and maintaining an ontological knowledge base [185]. KG construction requires developing an ontology for the initial KG and incrementally updating it to incorporate new kinds of information. As of today, ontology development and curation are still broadly manual or crowdsourced, although some semi-automatic approaches are also proposed. Semi-automatic ontology development tasks share a great overlap with methods from Knowledge Extraction, entity resolution, quality assurance, and knowledge completion. Creating the initial ontology can be derived from a single source that ideally provides already some useful ontology to build on. Public web wikis, catalogs, APIs, or crowdsourced databases are valuable starting sources as they may already contain a large amount of (semi-)structured data on general or domain-specific topics. However, cleaning and enrichment processes are required to ensure sufficient domain coverage and quality to build an initial Knowledge Graph structure from these existing data. For example, if Wikipedia is used as a primary source, its category system can be a good start to derive the most relevant classes for the KG by some NLP-based "category cleaning" [15]. Semi-automatic approaches mostly focus on learning an ontology from single sources, i.e., transforming a source into an ontology or KG. These individual ontologies or KGs can then be integrated into a previous version of the overall KG. A key prerequisite for this kind of ontology integration is the step of ontology and schema matching to determine respective ontology and schema elements (classes, properties). After discussing semi-automatic approaches for ontology learning, we discuss ontology/schema matching and conclude with approaches for ontology integration.

4.3.1. Ontology Learning

Ontology learning has two main subfields: extracting ontologies from unstructured text and from structured data like relational databases. For unstructured text, linguistic and machine learning approaches aid in semi-automatic ontology construction. In relational databases, reverse engineering and mapping techniques are used for transformation. Although fully automatic construction remains challenging [186–188], recent advancements, including the use of Large Language Models (LLMs), have improved semi-automatic methods in both areas.

Al-Aswadi et al. [186] give a state-of-the-art overview of ontology learning from **unstructured text**, where the goal is to identify the main concepts and their relations for the entities in a document collection. The approaches [186–188] can be grouped into linguistic approaches (using NLP techniques such as part-of-speech tagging, sentence parsing, syntactic structure analysis, and dependency analysis methods) and machine learning approaches. The latter include statistic-based methods (e.g., utilizing co-occurrence analysis, association rules, and clustering) and logic-based approaches using either inductive logic programming or logical inference. Al-Aswadi et al. [186] argue that there is a need to move from shallow to deep learning approaches for deeper sentence analysis and improved learning of concepts and relations.

With advancements in the field of Large Language Models (LLMs), there has been increasing interest in utilizing these models for ontology learning. Giglou et al. [189] investigate, among other tasks, the capability of several families of LLMs to deduce taxonomical information between types. Their results show that with an increasing number of parameters, the models perform better in the zero-shot paradigm (which means performing this task without having seen annotated examples). With finetuning, they could, in some cases, improve the results considerably. Funk et al. [190] have shown that OpenAI's GPT 3.5 [191] can be used to construct concept hierarchies for a given domain. While the resulting hierarchies are not perfect, the authors argue that this study demonstrates the possibility of utilizing LLMs in ontology construction. Recent developments present LLM-based user-guiding agents to create ontologies from competency questions. For instance, a semi-automatic pipeline employing LLMs generates and populates ontologies from competency questions extracted from scientific publications, highlighting the necessity of human validation and evaluation [192]. The OntoChat framework leverages LLMs to facilitate collaborative ontology engineering by guiding users in creating user stories, extracting and analyzing competency questions, and testing previous ontology versions via prompting [193]. Additionally, advanced LLMs with various prompting techniques are used to fully automate the generation of capability ontologies from natural language descriptions, focusing on developing robust quality assessment methods [194].

Ma et al. [195] give a survey of methods for learning ontologies from **relational databases** with a focus on methods for reverse engineering or the use of mappings to transform a relational database (schema) into an ontology or Knowledge Graph. Reverse engineering allows one to derive an entity–relationship diagram or conceptual model from the relational schema. Here, additional considerations are needed to deal with trigger and constraint definitions to avoid a semantic loss in the transformation. For the mapping techniques to transform RDBs to KGs, the authors differentiate between template-based, pattern-based, assertion-based, graph-based, and rule-based mapping approaches. The resulting mappings should be executable on instance data in order to generate a graph structure from a relational database [196,197].

An existing ontology can also be extended by inferring and making implicit knowledge explicit (as exemplified in Figure 1). Reasoners like Pellet [51] can be used to address this task in an efficient manner. Furthermore, an ontology can also be used to learn rules from it. For example, a learning algorithm can be used to refine or generalize the description of a class [198]. A comprehensive system that incorporates reasoners and learning algorithms is DL-Learner [199].

4.3.2. Ontology/Schema Matching

Consolidating and integrating information from multiple heterogeneous sources requires harmonizing the ontologies and/or schemas of the sources. The main step for such a data integration is ontology and schema matching, which is the task of identifying corresponding ontology and schema elements, i.e., matching ontology concepts and matching properties of concepts and entities. For example, to integrate a new source into a KG it is necessary to perform a matching of the source ontology/schema with the KG ontology to identify which source elements are already existing in the KG ontology and which ones should be added. Property matching is also important for entity resolution and entity fusion in order to determine matching entities based on the similarity of equivalent properties and to combine equivalent properties to avoid redundant information. In some cases, known entity matches can be used to aid in the ontology matching step [200]. Some tools also perform entity resolution and ontology matching in combination [201].

There is a huge amount of previous research on schema and ontology matching, although mostly outside the context of Knowledge Graphs, and there are numerous surveys and books about the topic [202–206]. The matching approaches typically rely on determining the similarity of elements using different strategies, such as the similarity of concept/property names or the similarity of instance values. Structural information can also be beneficial in the matching process, e.g., by looking at the concepts in the graph neighborhood. Matching systems commonly rely on a combination of different match strategies in order to achieve high-level match quality [207].

While string similarity can be a strong signal for a match decision, semantically similar words are often used that are dissimilar on the character level. Dictionaries or pre-trained word embeddings are, therefore, helpful for capturing this semantic similarity. Zhang et al. [208] investigated how word embeddings (using Word2Vec [209]) can be used for the task of ontology matching. They found a hybrid approach that performed the best. This approach takes the maximum of either edit-distance-based or word embedding similarity for each entity pair. Another approach that relies on word embeddings but also on meta-information about properties' names and their values as input for a dense neural network is LEAPME [210]. This system trains a classifier based on already labeled property pairs. This trained model can then decide whether unlabeled property pairs and their similarity scores constitute a match. Graph embeddings have also seen some attention in ontology matching, as they can capture structural information of an ontology. For example, Portisch et al. [211] use a variation of RDF2Vec [212], which is a walk-based embedding technique similar to Word2Vec, to encode both ontologies and then use a rotation matrix to align the embeddings. Qiang et al. [213] investigate how LLMs can be incorporated into the ontology matching task. They present an LLM-based agent system, wherein the LLM functions as a control center that utilizes different modules via prompt engineering. Falsely generated content of the LLM is mitigated by employing in-context learning and storing useful information in a searchable database, acting as short- and long-term memory for the inherently stateless LLMs. Their system performs comparably to state-of-the-art matchers. Hertling and Paulheim [214] study what design choices improve the results when utilizing LLMs in the matching process. Among other results, they find that presenting these models with a binary decision (match/no-match) leads to better results than a multiple-choice scenario. The few-shot paradigm leads to competitive results and is preferable to the zero-shot scenario.

There are some tailored ontology matching approaches for KGs, such as for mapping categories derived from Wikipedia to the Wordnet taxonomy with the goal of achieving an enriched KG ontology [15]. Other specific approaches have been developed to address the integration of RDBs with ontologies that go beyond the mapping languages described in Section 4.2.3. KARMA [148] provides a semi-automatic approach to link a structured source, such as an RDB, with an existing ontology. The process consists of assigning semantic types to each column, constructing a graph of all possible mappings between the source

and the ontology, refining the model based on user input, and finally generating a formal specification of the source model.

4.3.3. Ontology Integration

Merging new ontology or schema data into the existing KG ontology is a subtask of ontology or schema integration. This topic has achieved some attention where recent approaches utilize the mapping result of a match (alignments) to combine multiple ontologies/schemas [215–217]. If the match mapping is automatically determined, it must first be manually validated and possibly corrected to provide a valid basis for the merge step. Osman et al. [217] give a comprehensive and recent summary of ontology integration techniques, which can handle the merging of ontology and entity data using respective alignments. The authors distinguish the following merging strategies:

- **Simple Merge.** Imports all input ontologies into a new ontology and adds bridging constructs between equivalent entities, like defining OWL *equivalentClass* or *equivalentProperty* relations.
- **Full Merge.** Imports all source ontologies into a new ontology and merges each cluster of equivalent entities into a new unique entity with a union of all their relations, leaving equivalent classes untouched.
- **Asymmetric Merge.** These approaches import source ontologies into a preferred target ontology, preserving all its concepts, relations, and rules by merging matching entities into existing target entities or else by creating new ones.

Figure 3 visualizes each of the three strategies, where (a) also shows the source and target data that are merged. From the three merging strategies, the authors favor the last and mention it as a good solution for incremental ontology integration. The reason for this preference is that the asymmetric merge strategy preserves the target ontology during the integration and only adds new elements from a source ontology if necessary. This can also be seen in Figure 3c, where the target data are left unchanged, and only a new entity `E6` is added, which is connected to `E3` from the target data. An example approach for asymmetric ontology merging focusing on is-a relations is proposed in [216].

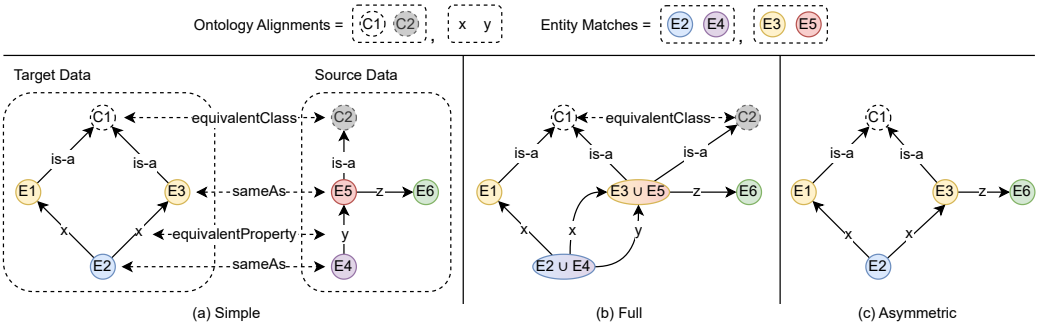

Figure 3. Ontology and entity merging strategies.

4.4. Knowledge Extraction

Knowledge extraction is a process to obtain structured, more computer-readable data from unstructured data such as texts or semi-structured data, like web pages and other markup formats. The extraction methods of semi-structured data often use a combination of data cleaning (Section 4.2.4) and rule-based mappings (Section 4.2.3) to transform the input data into the final KG, targeting already defined classes and relations of the existing ontology. Most of the work focuses on Knowledge Extraction from text, sometimes additionally considering images and figures within the text. Recently, there has been increased interest in creating multi-modal Knowledge Graphs (i.e., KGs with not only text but also other modes of data, such as images), necessitating appropriate methods of Knowledge Extraction. The detailed discussion of such methods lies outside the scope of this paper, but we refer the interested reader to the following survey by Zhu et al. [16]. The main

steps of text-based knowledge representation are named-entity recognition, entity linking, and relation extraction. These steps are discussed in the following and allow the extraction of entities and relations from text for inclusion into a KG. An example of this process is shown in Figure 4.

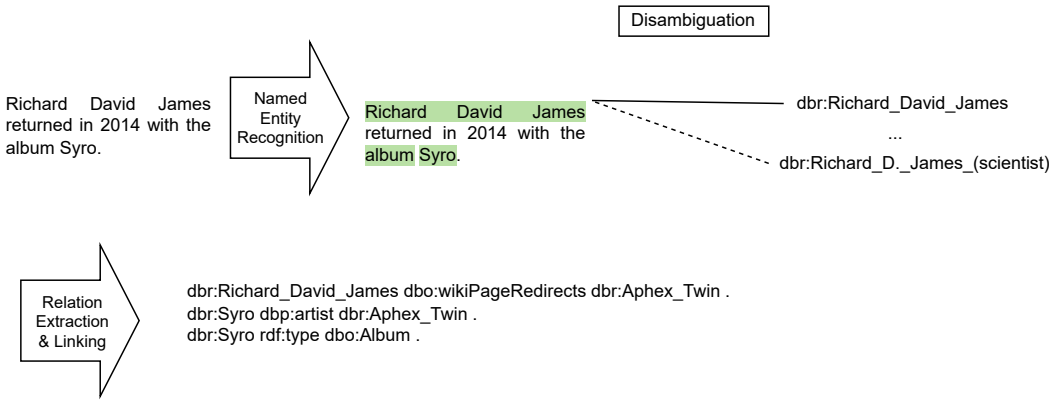

Figure 4. Knowledge extraction steps for an example sentence linking entities and relations to the DBpedia KG. Recognized named entities are highlighted in green.

4.4.1. Named Entity Recognition

Named entity recognition (NER) refers to demarcating the locations of entity mentions in an input text. In the most widely used scenarios, mentions of only a handful of types (persons, places, locations, etc.) are determined. However, KGs usually contain hundreds or thousands of types. Furthermore, off-the-shelf NER tools do not provide canonicalized identifiers for the extracted mentions. A second step is therefore necessary to link entity mentions either to existing entities in a KG or with new identifiers.

A relatively reliable and simple way to detect entity mentions in a text is the use of a *dictionary* (also referred to as *lexicon* or *gazetter*), which maps labels of desired entities to identifiers in the KG. In addition to its simplicity, such an approach already provides recognized entities in a text with the right link to the KG (i.e., solving the tasks of named-entity recognition and entity linking in one step). However, these dictionaries are usually incomplete. A simple way to increase the coverage of such dictionaries is to utilize disambiguated aliases in high-quality sources [15]. Wikipedia redirects or DBpedia's `dbo:alias` property would be a simple way to enhance an entity dictionary. For example, https://en.wikipedia.org/wiki/Richard_D_James from the example redirects to https://en.wikipedia.org/wiki/Aphex_Twin. To make dictionary lookups efficient, different data structures have been proposed. For example, prefix tries or inverted indexing have shown to be a scalable solution for large Web search engines and are used in the NER approaches AGDISTIS [218], TagME [219], and WAT [220].

Machine learning methods have become increasingly popular for tackling NER. These methods are especially useful for finding "emerging entities", i.e., entities that are unknown to the knowledge base. The machine learning models for entity recognition generally fall into the task of *sequence labeling*. A widely successful method for this task is known as conditional random fields (CRFs), which uses an undirected graph connecting input and output variables and models the conditional probability of output given the input. Generally, these graphs form a linear chain (e.g., in the Stanford CoreNLP package [221]), which means that for a prediction, only the immediate neighbors are relevant in a sequence. While CRFs require extensive feature engineering, Deep Neural Networks have become highly popular in recent years for the task of NER since they do not necessitate this amount of human interaction. For example *LSTM* networks (long short-term memory), which are a specific case of *recurrent neural networks* (RNNs), have become a prevalent choice for NER tasks [222]. The memory cells contained in this architecture are able to deal with long-term dependencies, which was previously a major pain point for RNNs. Deep learning-based approaches for NER are surveyed in [223]. BERT [224] (Bidirectional

Encoder Representations from Transformers) is another machine learning model for natural language processing, leveraging a bidirectional Transformer to contextualize words. When integrated with LSTM and CRF layers, it becomes tailored for tasks like NER and relation extraction. It has specialized adaptations like BioBert [41,225] for specific domains such as biomedicine. Given a specified schema, SPIRES [226] utilizes LLMs to perform NER and relation extraction. For NER, this approach relies on the LLM's zero-shot learning capabilities and interrogates them with specific prompt engineering. Using the specific context in the prompt greatly improves the LLM's capabilities to perform this task compared with simply asking the LLM for a specific identifier for a term.

As multi-modal data become increasingly popular, e.g., on social media platforms, several recent studies focus on multi-modal NER (MNER) [227,228], where the goal is to leverage the associated images to better identify the named entities contained in the text. Furthermore, there are first approaches [229–231] that address MNER for KGs. They aim to correlate visual content with textual facts. One typical solution parses images and texts to structured representations first and grounds events/entities across modalities. However, intra-modal relation extraction and cross-modal entity linking are still largely unresolved problems.

4.4.2. Linking

If named entities are recognized in a text, they need to be linked to the knowledge base or KG. This is called *entity linking* (EL) or *named entity disambiguation* (NED). Given a set of candidates from the knowledge base, an EL algorithm needs to decide which entity a mention belongs to. In Figure 4, this can be seen, where `Richard David James` is linked to the DBpedia entity `dbr:Richard_David_James`.

EL algorithms can rely on a variety of features. Based on the mention itself, the *confidence* of the used NER tool can be used, how *similar* the mention and the entity are, or how much *overlap* exists across mentions [232]. The context of the extracted mentions can be valuable. Keyword-based similarity can be used by relying on TF-IDF scores, where rare keywords used in the mention's context, which connect to a candidate entity, can give hints for linkage [233,234]. Words that occur frequently in the same context can also help in the disambiguation process. Here, pre-trained word embeddings can prove especially useful since they encode semantic similarity in a latent space. Furthermore, already disambiguated mentions can be used to aid in the linking of entities that occur in the same paragraph.

Holistic entity linking [232,235] approaches leverage background information in the decision process that exceeds merely using the similarity between mention and entity. Popularly, the graph structure of Wikipedia links can be used to determine *commonness* and *relatedness*. Commonness refers to the probability that an entity mentions links to the respective Wikipedia article of the given candidate entity. Relatedness measures how many articles in Wikipedia link to the articles on both candidates. Using such background knowledge, unambiguous mentions can aid in the correct linkage of ambiguous mentions [236].

Entity linking approaches furthermore need to address specific challenges such as *coreference resolution*, where entities are not consistently referred to by their names, but with indirect references such as pronouns [232], and deal with *emerging entities*, i.e., entities that are recognized but not yet existing in the target KG. For example, Hoffart et al. [237] keep a contextual profile of emerging entities, and when this profile contains enough information to infer the semantic type of the mention, it can be added to the KG with its type.

Generally, entity linking and the later-discussed entity resolution (Section 4.5) share similarities in aiming to connect the same entities in and across data sources. entity linking and entity resolution are sometimes jointly discussed under the term *entity canonicalization* [15]. While entity resolution typically deals with at least semi-structured data sources, there have been some efforts to address cases with unstructured sources, where deep-learning-based approaches are advantageous [238]. However, there are some key differences not only in the characteristic modality of the data sources but also in the signals that lead to a linking decision. For example, in the entity-linking scenario, if one has already

linked the mention "Richard James" to the entity `dbr:Richard_David_James`, seeing the mention "James" in close context makes it likely that it also refers to the same entity. By contrast, in the entity resolution scenario, if one has already confidently matched two entities, it is unlikely that a similar unmatched entity from one data source will also be matched with the already matched entity from the other data source. This is because matching can often focus on $1-1$ correspondence between two data sources under the assumption of deduplicated or clean data sources. However, it could be worthwhile to investigate how well entity-resolution approaches for dirty sources (where multiple entities may match with the same KG entity) can be utilized for entity linking and vice versa.

4.4.3. Relation Extraction

Given the identified entities in a text, relation extraction aims to determine the relationship among those entities. In Figure 4 we see this, for example, when the text snippet `album Syro` becomes the triple `dbr:Syro rdf:type dbo:Album`, i.e., when the type relation for entity `dbr:Syro` is determined.

The first techniques use rule-based approaches to extract relations, e.g., by relying on Hearst patterns to find hyponym (*is-a*) relations [239] or involving regex expressions [240,241]. In order to improve coverage, different ways to enhance such patterns were devised. The human involvement in these techniques, however, is a limiting factor. To address these shortcomings, statistical relation-extraction models were devised. Feature-based methods rely on lexical, syntactic, and semantic features to use as input for relation classifiers. Similarly, kernel-based methods [242] rely on specifically designed kernel functions for SVMs to measure the similarity between relation candidates and text fragments. Graph-based methods further integrate known relations between entities and text in order to correctly identify relations [15].

While such methods can be incredibly useful to obtain relatively simple relations with high accuracy, they are limited in terms of their recall or at least require a high degree of additional human involvement for feature engineering, the design of kernel functions, or the discovery of relational patterns [243,244]. Neural relation-extraction methods aim to close this gap. The input text is transformed via (pre-trained) word embeddings and position embeddings into a format that is suitable for the neural networks that are trained for relation extraction. Instead of devising hand-crafted features, this area focuses on investigating various neural network architectures, such as recurrent neural networks, convolutional neural networks, and LSTMs. The bottleneck for these approaches lies in the availability of training data. A common approach to address this is via distant supervision. Statements from a given data source (for example, Wikipedia) are used to train the given model. In particular, the use of pre-trained language models has pushed the state of the art to new heights [245,246]. Han et al. [247] provide a more in-depth overview of these methods and identify the main challenges in the ability to utilize more data, creating more efficient learning schemes, handling more complex contexts (e.g., relational information *across* sentences), and detecting undefined relations in new domains. With the rising popularity of LLMs, Giglou et al. [189] benchmark several model families on the task of relation extraction (among other tasks). They achieve the best results with the open-source model Flan-T5-XL [248] and can improve the performance of this model even further with finetuning. During the fine-tuning process, the prompts for the LLMs can be optimized by injecting semantic information into the prompts, leveraging already existing information [249].

Neural approaches can suffer from *catastrophic forgetting* [250], where previously learned information is lost in favor of recently introduced knowledge. The specific field of Continual Relation Extraction aims to mitigate this problem by employing techniques from Continual Learning. For example, Hu et al. [251] jointly train a classification and contrastive network, where, in the latter, contrastive learning is used with prototypical samples. Similarly, Zhao et al. [252] replay representative samples from previous iterations while also minimizing distributional shifts in the already-learned embeddings.

A special case of relation extraction aims to extract relations freely without a predefined set of relations. This is known as Open Information Extraction (OpenIE). While this can be a good way to increase the variety of information contained in the KG, a secondary step is necessary to *canonicalize* the extracted relations in order to deduplicate and possibly even link them to already contained synonymous relations in the KG [253].

Several tools exist for the entire process of Knowledge Extraction, with some tools focusing on specific aspects. For example, DBpedia Spotlight [254] mainly aims at performing named entity extraction and links those mentions to the DBpedia KG. The dstlr [255] tool extracts mentions and relations from text, links those to Wikidata, and furthermore populates the resulting KG with more facts from Wikidata. OpenNRE [256] provides an extensible framework for neural relation extraction, with trainable models; however, this approach would necessitate an independent linking step afterward.

Analogously to NER, there are also efforts to use images as information sources for relation extraction. These can range from rule-based approaches [257], which, for example, verbalize detected spatial relations of recognized entities in an image, to learning-based techniques, which encode visual features of detected objects as well as textual features into distributed vectors used to predict relations between given objects. For example, MEGA [258] aligns information contained in the syntax tree and word embeddings of the textual data and the scene graph obtained from the image. A scene graph connects detected objects in an image via their visual relations. After the alignment process, the respective representations are concatenated and sent to a Multilayer Perceptron (which is a fully connected feed-forward neural network) to predict the relation.

4.5. Entity Resolution

Entity resolution (ER), also called entity matching, deduplication, or link discovery, is a key step in data integration and for good data quality. It refers to the task of identifying entities either in one source or different sources that represent the same real-world object, e.g., a certain customer or product. An enormous amount of research has dealt with the topic as evidenced by numerous surveys and books [259–264]. In addition to several research prototypes, there are also many commercial solutions such as IBM's InfoSphere Identity Insight (https://www.ibm.com/products/infosphere-identity-insight (accessed on 15 August 2024)) or SAP's Master Data Governance Platform (https://www.sap.com/products/technology-platform/master-data-governance.html (accessed on 15 August 2024)). Most known approaches tackle static or batch-like entity resolution, where matches are determined within or between datasets of a fixed size. The more recent of these approaches deal with multi-source big data entity resolution [265,266], relying on Deep Learning [238,267] or KG embeddings [268,269], with the neural methods having seen more scrutiny recently after an era of relative hype [270]. Further, there are data- and domain-specific entity-resolution algorithms, like ORCHID, for matching geospatial entities [44].

For KG construction, however, we need incremental approaches that build on previous match decisions and determine whether new entities are already represented in the KG or whether they should be added as new entities. Furthermore, for streaming-like data ingestion into a KG, a dynamic (real-time) matching of new entities with the existing KG entities should be supported. Entity resolution results are fed to the step of *entity fusion*, which fuses matching entities to combine and thus enrich the information about an entity in a uniform way.

In the following section, we discuss proposed approaches first for incremental ER and then for entity fusion.

4.5.1. Incremental Entity Resolution

Entity resolution is challenging due to the often limited quality and high heterogeneity of different entity descriptions. It is also computationally expensive because the number of comparisons between entities typically grows quadratically with the total number of

entities. The standard approach for entity resolution uses a pipeline of three succeeding phases called blocking, linking/matching, and clustering [119,271]. While there is growing interest in utilizing KG embeddings for the incremental setting [272] this research is still in its infancy, and we will therefore focus on the more prevalent approaches. The main step is to determine the similarity between pairs of entities to determine candidates for matching. This matching step often results in a similarity graph where nodes represent entities and edges link similar pairs of entities. The preceding blocking phase aims at drastically reducing the number of entity pairs to evaluate, e.g., based on some partitioning so that only entities of the same partition need to be compared with each other (e.g., persons with the same birth year or products of the same manufacturer). After the match phase, there is an optional clustering phase that uses the similarity graph to group together all matches. This phase can typically improve the quality of entity resolution by relying on a more holistic perspective on entity similarities when compared to myopic pairwise matching. The clustering step also assists the succeeding step of entity fusion to fuse the matching entities into one representative entity for the KG.

For incremental ER, the task is to match sets of new entities from one or several sources with the current version of the KG, which is typically very large and contains entities of different types. It is thus beneficial to know the type of new entities from previous steps in the KG construction pipeline so that only KG entities of the same or related types need to be considered. Figure 5 illustrates a high-level workflow for incremental ER. The input is the current version of the KG with the already integrated entities (previous clusters in Figure 5), as well as the set of new entities to be integrated. This requires the development of incremental versions for blocking, matching, and clustering phases that focus on the new entities. To allow better match decisions for incremental ER, it is generally advantageous to retain the entities of the previously determined clusters (and their match similarities) and not only the fused cluster representatives. Some incremental clustering schemes can also use this to identify previous match mistakes and to repair existing clusters for new entities [119,273]. Some approaches [274] and tools [275] also support executing incremental ER in parallel on multiple machines to improve execution times and scalability to deal with large KGs.

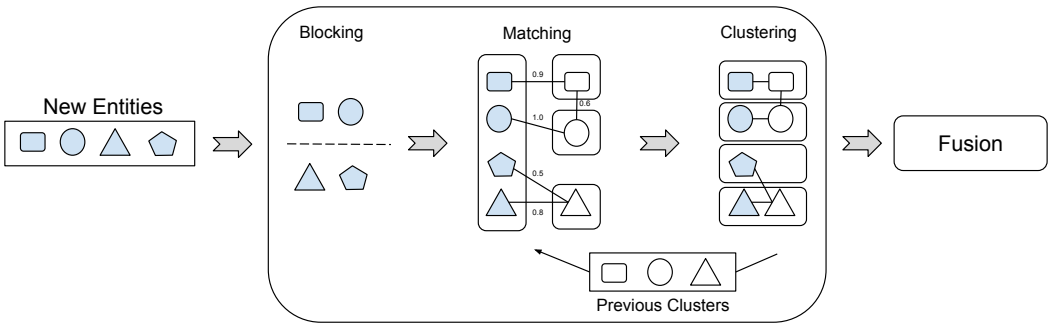

Figure 5. Incremental entity-resolution workflow.

Blocking for incremental or streaming ER requires identifying for the new entities all other entities in the KG that need to be considered for matching. Given the typically high and growing size of the KG, it is important to limit the matching to as few candidates as possible, and the determination of the candidates should also be fast. As mentioned above, blocking and entity resolution should be limited to entities of the same (or most similar) entity type, and one can apply the same blocking approach for the new entities as for the previously integrated entities, e.g., by using some attribute-based blocking key such as the birth year for persons or the manufacturer for products. Several works [263,264] have proposed further improvements over such a base approach for specific cases and blocking approaches. One approach [276–278] is to keep the blocking keys in a data structure with efficient data access to make comparisons among entities faster. Another method to speed up the computation is the use of so-called summarization techniques [278]. One such

approach summarizes (divides) larger blocks into multiple sub-blocks with a representative and directs a new record (query) to the sub-block with the most similar representative. This enables a constant number of comparisons for each new record, which is valuable for both incremental and streaming ERs. While most blocking approaches rely on domain or schema knowledge, there are also so-called schema-agnostic blocking schemes for highly heterogeneous data where entities of a certain type can have different sets of attributes. Hence, schema-agnostic approaches consider most or all attribute values and their components (e.g., words or tokens), regardless of the associated attribute names. While there are many schema-agnostic blocking approaches for non-incremental ER [263,264], schema-agnostic blocking approaches for incremental or streaming ER have only recently been proposed [274,279,280]. The use of deep learning for blocking has gained significant attention in recent years. One of the pioneering frameworks in this domain is DeepBlock [281], which introduced the use of embeddings for blocking. Building on this concept, AutoBlock [282] utilizes labeled data to position record embeddings within the embedding space, employing locality-sensitive hashing to identify the nearest neighbors for candidate set construction. Further advancements were made by DeepBlocker [283], which delved into deep self-supervised learning for blocking. This work presented two state-of-the-art methods: one based on auto-encoding (Auto) and the other on cross-tuple training (CTT). More recently, Sudowoodo [181] has been proposed, leveraging self-supervised learning alongside a transformer model to enhance blocking performance.

The **matching** step of incremental ER is limited to the new entities and involves a pair-wise comparison with the existing KG entities determined by the preceding incremental blocking step. The main goal is to determine all similar entities as potential match candidates as input for the final clustering step, where it is decided whether the new entity is added to an existing cluster or whether it should form a new cluster. Pairwise matching can be performed for batch-like ER and is based on the combined similarity between two entities derived from property values or related entities. The matching approach can be configured manually, e.g., together with some similarity threshold that should be exceeded for match candidates or by applying a supervised machine learning model [263]. If the pairwise match relationships between previously integrated KG entities are maintained in a similarity graph spawning the previous clusters, this graph can be extended by the new entities and links to the newly determined match candidates as input for incremental clustering [275].

While there are many approaches for batch-like entity **clustering** [284,285], the incremental maintenance of entity clusters for new entities has received comparatively little attention. A straightforward approach is to either simply add a new entity to the most similar existing cluster or to create a new cluster if there is no previous cluster with a high enough similarity exceeding some predefined similarity threshold [286]. However, this approach typically suffers from a strong dependency on the order in which new entities are added. In particular, wrong cluster decisions, e.g., due to data quality problems, will not be corrected and can lead to further errors when new entities are added. A more sophisticated incremental approach based on correlation clustering is proposed in [273], which maintains previous clusters within a similarity graph. The updated similarity graph is used not only to determine the clusters for new entities but also to repair previous clusters, e.g., by splitting and merging clusters or by moving entities among clusters. The incremental approaches in [118,119] support optimized clustering decisions for duplicate-free (sometimes called clean) data sources from which at most one entity can participate per cluster of matching entities. In this case, an effective clustering strategy is the so-called "max-both" approach, where an entity s from a set of new entities is only then added to the most similar cluster c when there is no other new entity that is more similar to c than to s. The approach of [119] also supports a lightweight cluster repair called n-depth reclustering, where only entities close to new entities in the updated similarity graph are considered for changing clusters.

Recent advancements in entity resolution leverage transformers and Large Language Models (LLMs) to achieve state-of-the-art performance [287–289]. Notably, Refs. [181,289] employ contrastive learning to enhance accuracy, distinguishing similar and dissimilar entity pairs and optimizing entity representations. Other studies explore graph-based methods [290,291] and the application of domain adaptation techniques for entity matching [42,292].

Several works utilize LLMs' natural language capabilities to improve entity resolution on tabular data [92], while another study [293] explores foundational models for generic entity resolution by evaluating various prompt types on domain-specific datasets. AutoAlign [294] adapts zero-shot approaches for Knowledge Graph entity alignments by creating similarity graphs for matching properties and entities.

Novel methods further enhance entity alignment in Knowledge Graphs. One approach [295] integrates visual and textual information, another [296] uses BERT's multilingual capabilities for temporal Knowledge Graphs, and a hybrid attention model [297] addresses structural bias. Furthermore, a geospatial entity-resolution method specifically designed for handling spatial data has been introduced, providing a tailored approach for resolving entities in geospatial Knowledge Graphs [43].

4.5.2. Fusion

Merging multiple records of the same real-world entity into a single, consistent, and clean representation is referred to as data fusion [298]. This is an important step in data integration as it combines information from several entities into one enriched entity. Data fusion still entails resolving inconsistencies in the data. First, the records may disagree on the names of matching attributes so that one preferred name has to be chosen that should be consistent with the attribute names of other entities of the same type to facilitate querying. Furthermore, the matching records can disagree on the values of an attribute. There are three main strategies to handle such attribute-level inconsistencies or conflicts [298]:

- Conflict Ignorance: The conflict is not handled, but the different attribute values may be retained, or the problem can be delegated to the user application.
- Conflict Avoidance: It applies a unique strategy for all data. For example, it prioritizes data from trusted sources over others.
- Conflict Resolution: It considers all data and metadata before making a decision to apply a specified strategy, such as taking the most frequent, the most recent, or a randomly selected value.

Such techniques were first applied for relational data but also found use for Linked Data fusion [299]. A valuable strategy is to combine multiple value-scoring functions. Mendes et al. [300] combine two methods named *TrustYourFriends* (prioritizing data from the trusted source) and *KeepUpToDate* (using the latest value) for conflict avoidance and resolution. Moreover, they apply input quality assessment metrics to filter out the values below a threshold or keep the values with the highest quality assessment. Other techniques, such as computing the average, minimum, and maximum or taking the most frequent values, are provided by their data-integration framework. Dong et al. [301] combine *TrustYourFriends* with a *Weighted Voting* (most frequent or similar values) approach, whereas the former source's ranking score is calculated based on a statistical approach. Similarly, Frey et al. [103] applied a median-based approach for Linked Data fusion. They distinguish functional properties (a functional property is specified to have a maximum cardinality of one; e.g., a person entity should only have one value for the date of birth) and assign a single value to them. Non-functional properties can be assigned multiple different values.

4.6. Completion

Knowledge Graph completion involves adding new entries (nodes, relations, properties) to the graph using existing relations. Paulheim [22] surveys KG completion approaches as well as evaluation methods. He also distinguishes internal from external methods, espe-

cially for determining missing entity type information and relations. Internal approaches solely rely on the KG as input, whereas external methods incorporate additional data like text corpora and, in a broader context, human knowledge sources like crowdsourcing. The survey concludes that current approaches for KG completion typically limit themselves to a single task, such as determining missing type information, missing relations (link prediction), or missing attribute values (literals). Holistic solutions to improve the quality of KGs simultaneously in several areas are thus currently missing.

4.6.1. Type Completion

Type completion refers to the task of assigning types to nodes without type information. In the case of PGM, it is in most cases not allowed to have nodes without type information [62]. Since there is limited standardization in the realm of PGM [302], and the possibility to label nodes with unknown type with a dummy label, type completion can still be seen as a relevant KG completion task. In this case, node classification approaches can be used to predict classes of unlabeled nodes. For example, Neo4j provides a specific node classification pipeline (https://neo4j.com/docs/graph-data-science/current/machine-learning/node-property-prediction/nodeclassification-pipelines/node-classification/) (accessed on 15 August 2024), although the resulting predictions are added as node properties necessitating a post-processing step to redefine the label.

The traditional way of determining missing type information in RDF datasets involves the use of logical reasoning. However, this approach is limited since it relies on already consistent facts and existing `rdf:type` information in the knowledge base [303]. To address this shortcoming, statistical approaches use the distribution of relations between entities to predict missing type information. For example, SDType [304] uses a weighted voting approach based on the statistical distribution of the subject and object types of properties. Similarly, StaTIX [305] relies on weighted statistics of multiple properties of entities as input for their clustering approach.

Recently, there has been some attention on leveraging KG embeddings to infer type information. For example, ConnectE [306] incorporates two mechanisms, with one relying on *local typing knowledge* and the other on *global triple knowledge*. The first relies on the fact that entities close in the embedding probably share the same type. Relying on relationship information, the second mechanism learns entity type embeddings by replacing the subject and object entity in a triple for their corresponding type. Finally, for entity type prediction, a composite score of the two mechanisms is used.

4.6.2. Link Prediction

The task of link prediction aims to find missing relations in a KG. In the example presented in Figure 1, we see that while there is a *writtenBy* relation between the song *Xtal* and the artist *Aphex Twin*, there is no relation between the song *Ageispolis* and *Aphex Twin*. Predicting this missing *writtenBy* relation would be a goal of link prediction.

Based on [22], a common method for the prediction of a relation between two entities is distant supervision [307–310] using external resources. This method starts with linking entities of the Knowledge Graph to the text corpus using NLP approaches and then tries finding patterns in the text between entities. Another approach [311] uses the same methodology but considers the whole Web as the corpus. Lange et al. [312] learn patterns on Wikipedia abstracts using Conditional Random Fields [313]. Blevins et al. [314] propose a similar approach but on entire Wikipedia articles. Another line of research uses semi-structured data such as tables [315,316] or list pages [317] in Wikidata for predicting missing relations.

In recent years, considerable research has been devoted to investigating KG embeddings for link prediction. These methods encode entities and relations of a KG as low-dimensional vectors in an embedding space. The existing triples in a Knowledge Graph can be used to train such models, and their performance can be evaluated on a held-out set of triples. For example, TransE [318] encodes relations as translations from subject to

object entity of a triple $(subject, predicate, object)$. (In the link-prediction literature, triples are usually signified as $(head, relation, tail)$. In favor of consistent nomenclature, we use the triple signifiers commonly used for RDF.) This is achieved by minimizing the distance between $\mathtt{s} + \mathtt{p}$ and $\mathtt{o}$, where $\mathtt{s}$, $\mathtt{p}$ and $\mathtt{o}$ are the embeddings of *subject*, *predicate* and *object*, respectively. A variety of approaches have been devised to address the problems of TransE to model $1-n$ or $n-n$ relations by, e.g., encoding relations in a separate hyperplane [319] or operating in the hyperbolic space [320]. For a broad overview and benchmark study, we refer to [321].

The described embedding-based link prediction methods rely on *shallow embeddings*, which means all embeddings are stored in an entity/relation matrix, and obtaining the respective embedding for an entity or relation is completed by using a lookup table. These approaches are unable to deal with unseen entities. The study of *inductive* link prediction aims to address this shortcoming. GraIL [322] relies on Graph Neural Networks (GNNs) to achieve this. This approach samples the subgraph enclosing the link to be predicted and labels the nodes in this subgraph based on their distance to the target nodes (i.e., the nodes to which the link would connect). The labeled subgraph is then used in a GNN to score the likelihood of a triple. NodePiece [323] can perform inductive link prediction via a compositional representation for entities. Relations around a node are sampled in order to create a node hash, which is passed through an encoder to obtain the final entity embedding. Being able to create entity representation for unseen entities but known relations permits NodePiece to then use any scoring function (e.g., TransE) for the link prediction task.

A special type of link prediction aims to discover identity links (e.g., `owl:sameAs` relations) [8], which connect nodes that refer to the same entity. This task serves the same goal as entity resolution (discussed in Section 4.5).

4.6.3. Data Enrichment

Concerning aspects of domain coverage and succinctness, additional processes are applicable to increasing the final quality of the KG. In addition to type and relation prediction, domain knowledge could possibly be extended by loading completing entity information from external (open accessible) knowledge bases. This approach is different from the process of integrating an entire external data collection but only focuses on loading necessary domain information that relates to the already integrated entities. For enhancing KG data with additional relevant domain entities, information from external knowledge bases can be requested based on extracted (global) persistent identifiers (PIDs). For example, extracted ISBN numbers, DOIs, or ORCIDs allow one to request additional external information from Wikidata, or Gene and Protein data are accessible based on their symbols in public biochemical databases, like the National Library of Medicine (https://www.ncbi.nlm.nih.gov/ (accessed on 15 August 2024)). Paulheim surveys approaches that exploit links to other KGs in order to not only verify information but also to find additional information to fill existing gaps [22].

Rule mining is used to discover rules that generate new information. When dealing with Knowledge Graphs, it is important to consider that missing facts are not necessarily false but rather indicate an incomplete graph (Open World Assumption). Approaches have to be, therefore, specifically tailored towards the Open World Assumption, which is, for example, achieved by the rule mining approach AMIE [324]. Neuro-symbolic techniques are also used to learn logical rules, enriching a KG by combining neural methods' ability to learn complex patterns with the interpretability of logical rules. Cheng et al. [325] propose a neural compositional rule-learning model that samples paths from KGs as compositional rules, breaking them into atomic components. A recurrent attention unit merges these components to infer the rule head, achieving state-of-the-art results and enabling inductive relational reasoning, including inferring missing relations with more hops than seen during training. Another important task accomplished with reasoning is entity classification, which involves automatically determining the most specific classes for instances within an ontology using defined classes, properties, and logical rules [326,327].

Research on LLMs and KGs has largely focused on knowledge completion. KG-BERT [328] treated triples as textual sequences, taking entity and relation descriptions to classify triples and predict links and relations. Using LLMs to fill in missing information in knowledge bases by prompting them with partially masked inputs of partial statements in the KG can lead to high accuracy in certain areas, significantly cutting down the need for costly manual data verification. [329,330]. Omeliyanenko et al. [331] propose a novel approach for automated Knowledge Graph completion by integrating trainable adapters with capsules from the field of continual learning, showing significant advantages in low-resource situations, particularly in the task of link prediction. NLP saw a paradigm shift from "pre-train, fine-tune", where a pre-trained language model is adapted to downstream tasks, to "pre-train, prompt, predict" [332], which reformulates downstream tasks to look more like the tasks encountered during pre-training. Inspired by this new learning pattern, research attention has been devoted to adapting this to the graph neural network domain. Here, the pre-training task is usually link prediction, with, e.g., node classification being a downstream task. Node classification is then reformulated as a link prediction task (which is carried out by the graph prompting function) [333]. Newer work generalizes the pre-training task to graph-level learning objectives and is able to perform well on multiple downstream tasks, ranging from node-level and edge-level to graph-level [334].

4.7. Quality Assurance

The quality of a KG is crucial for its credibility and, therefore, its usability in applications [335]. Quality assurance is the task of maintaining a high KG quality despite the continuous evolution of the KG. It comprises *quality evaluation* to assess the quality and detect quality issues, as well as *quality improvement* to fix or mitigate quality issues by refining, repairing, and completing the KG. We already discussed *knowledge completion* separately in the previous Section 4.6, due to its unique nature of adding data to the KG rather than improving or removing existing information. Quality assurance is important not only for the resulting KG as an outcome of the KG construction process but also within the different construction tasks, such as selecting good-quality sources (Section 4.2.2), data cleaning for acquired data, Knowledge Extraction, ontology evolution, or entity fusion. The data cleaning approaches mentioned in Section 4.2.4 can also be applied to the KG, e.g., to identify outliers or contradicting information. Metadata such as provenance information is also important for quality assurance, for example, to explain and maintain KG data concerning the context and validity of conflicting values [15].

Assessing the quality of a KG is extremely challenging since there are many valid ways to structure and populate KGs, and even subproblems such as evaluating the quality of a KG ontology are already difficult [336,337]. In general, evaluating the quality depends on the scope of a KG and should be easier for domain-specific KGs than for very large KGs covering many domains for which completeness may not be possible. Moreover, the intended KG use cases influence the quality needs of the KG and should thus be considered for KG construction and KG evaluation. For example, e-commerce KGs such as the Amazon Product Graph [40] should provide up-to-date and reliable product information, demanding the enforcement of quality criteria such as accuracy and timeliness. Applications such as financial risk analysis also need accurate and trustworthy information from KGs such as the Bloomberg Knowledge Graph [38]. On the other hand, there may also be use cases for which approximate answers—and thus reduced KG quality—may be sufficient, e.g., for obtaining recommendations (similar products, related literature) or to receive aggregated information (e.g., about the relative average income in different countries).

We begin our overview of quality assurance by identifying and describing important quality dimensions. Then, we explore various evaluation methods to measure these dimensions. Next, we investigate correction methods to improve and rectify quality issues. Finally, we present quality evaluation frameworks and benchmark datasets that facilitate quality assessment.

4.7.1. Quality Dimensions

KG evaluation typically involves analyzing various quality dimensions, and the relevance of the dimensions typically depends on the intended kinds of KG usage. Quality dimensions can be correlated and possibly impact each other positively or negatively; for example, completeness can negatively affect accuracy. Wang et al. identified and surveyed in [91] six main quality dimensions (accuracy, consistency, timeliness, completeness, trustworthiness, and availability) for use in KG evaluation:

- Accuracy indicates the correctness of facts in a KG, including type, value, and relation correctness. It can be separated into syntactic accuracy, assessing wrong value datatype/format, and semantic accuracy, assessing wrong information.
- Consistency ensures coherency and uniformity of the data within the graph. A consistent KG follows logical rules, avoids contradictions, and maintains coherence among entities, relationships, and attributes. Inconsistencies arise from conflicting information, duplicates, or rule violations.
- Timeliness in the context of KGs refers to the currency and freshness of the information present in the graph. KG timeliness can be influenced by the chosen integration approach, which may involve batch processing at specific intervals or real-time updates.
- Completeness captures and reflects knowledge coverage within a specific domain. Completeness is also a goal for KG completion as it involves generating new values or data to augment the current KG.
- Trustworthiness indicates the confidence and reliability of the KG and depends on source selection and the applied construction methods. It is strongly related to the quality dimensions of completeness, accuracy, and timeliness.
- Availability is the extent to which knowledge is convenient to use. In other words, it refers to how easily and quickly the knowledge of KGs can be retrieved concerning query complexity and data representation.

4.7.2. Evaluation Methods

A common approach involves crowdsourcing techniques or expert knowledge in evaluating Knowledge Graphs. During the validation phase, curators can spot errors or verify facts. Using an iterative human-in-the-loop process allows for continuous improvement and refinement, enhancing the overall reliability and trustworthiness of the graph's data. One conventional approach is to evaluate the accuracy of the KG against a manually labeled subset of entities and relations [22]. However, this is costly, so those manually labeled gold standards are usually small. Other approaches use statistical methods such as distance-based, deviation-based, and distribution-based methods [338]. Acosta et al. [339] leverage the wisdom of the crowds in two ways. They launched a contest targeting an expert crowd in order to find and classify erroneous RDF triples and then published the outcome of this contest as paid microtasks on Amazon Mechanical Turk (MTurk) in order to verify the issues spotted by the experts. Their empirical evaluation on DBpedia shows that the two styles of crowdsourcing are complementary and that crowdsourcing-enabled quality assessment is a promising and affordable way to enhance data quality. Paulheim et al. [22] define *retrospective evaluation* as a method in which the human judges the correctness of the KG. The reported quality metric is accuracy or precision. Since KGs are often voluminous, the retrospective approach is typically restricted to a KG sample.

Another method to evaluate quality is statistical analysis, identifying outliers, inconsistencies, or abnormal data distributions based on patterns and the data structure, including clustering, correlation analysis, or anomaly detection techniques [340].

Further, semantic reasoning and inference allow for the validation of the KG's consistency based on the given ontology or individual structural constraints. One method is to calculate disjoint axioms by identifying wrong types of statements based on existing relations (e.g., domain and range checks or disjoint classes) [341]. Earlier reasoners optimized methods [27,51], while recent approaches use distributed frameworks for scalability [52–54].

Data profiling and cleaning techniques can also be applied to find erroneous values based on their distribution. Duplicate detection, schema matching, or entity resolution can be used to identify and resolve inconsistencies, redundancies, or errors (format errors).

Another method of quality evaluation relies on aligning and comparing the KG's entities with external knowledge and reference sources. Li et al. [342] investigate the correctness of a fact by searching for pieces of evidence in other knowledge bases, web data, and searching logs. Similarly, ref. [343] suggests the individual checking of a single fact in different datasets in order to detect inaccurate facts. This is also useful for yielding larger-scale gold standards, but it has two sources of error: errors in the target Knowledge Graph and errors in the linkage between the two. Tuefek et al. [344] present a recent method using Large Language Models to translate requirements texts into SPARQL queries for KG validation.

Finally, rule-based analysis is a common solution for detecting quality issues based on manually generated constraints, such as value restrictions or allowed/wanted properties for a specifically typed entity. For RDF, besides syntax validation (triples/quads, URIs, datatypes), RDF triple stores do not provide a standard method to define and validate graph data integrity or shape constraints (similar to relation database schemata). Therefore, overlaid solutions such as SHACL (Shape Constraint Language [56]) or ShEx (Shape Expressions [57]) are developed, which can be used to validate the semantic correctness of the graphs structure, node or property constraints, cardinalities, and other constructs. Similar to RDF stores, the data integrity of PGM databases is generally limited to syntax or basic value constraints. A first effort about the aspects of property graph *key constraints* is proposed by Angles et al. [78] by identifying four natural key types: identifier, exclusive mandatory, exclusive singleton, or exclusive. Additionally, PG-Schema offers a robust formalism for specifying property graph schemas [79].

4.7.3. Quality Improvement

Quality improvement aims to optimize the KG, making it more reliable, useful, and valuable for its intended purpose and domain. This includes and combines several task areas discussed in the former sections. Data cleaning (Section 4.2.4) addresses errors, inconsistencies, and redundancies in the graph. Error-correction techniques eliminate incorrect or outdated information and adjust inconsistent data entries. Outlier detection identifies and handles data points deviating significantly from the norm. Entity resolution (Section 4.5) methods merge or link entities referring to the same real-world entity. Data fusion (Section 4.5.2) integrates information from multiple sources to enhance overall data quality. Continuous ontology development (Section 4.3) refines and expands the graph's underlying ontology to accommodate new knowledge and evolving requirements.

Instead of filling in missing data, it may be preferable to remove irrelevant entities that do not pertain to the intended domain. This will prevent the KG from being unnecessarily bloated. Applying automatic approaches can cause irrelevant information to be extracted and requires manual techniques or techniques leveraging known information from external, already structured databases.

In KG, quality assurance, versioning, and rollback mechanisms are crucial for managing errors and maintaining data integrity. By implementing version control mechanisms, changes in the KG can be tracked, allowing for easy rollback in the event of errors or quality issues. This ensures that previous versions of the KG can be restored, providing a safety net for data consistency. Furthermore, maintaining an audit trail of changes and ensuring traceability supports data governance and reproducibility. Section 4.1.3 discusses various approaches to versioning that can be applied in this context.

4.7.4. Frameworks and Benchmarks

The importance and complexity of KG quality assessment and improvement demand powerful frameworks and tools to support these tasks. A quality evaluation framework incorporates metrics and processes to evaluate quality dimensions, ensuring a clear under-

standing of the graph's quality aligned with specific applications or use cases. Further, such a framework can already support mechanisms and techniques for quality improvement, either requiring a human-in-the-loop approach or applying automatic error correction.

Chen et al. [345] give an overview of the requirements of KG evaluation frameworks, focusing on specific domains. A special requirement is the scalability of such a framework to be applicable to a huge amount of data. Considering the degree of automation, using human-in-the-loop approaches might require KG sampling to only evaluate sub-graphs of the entire KG.

Several frameworks and tools for KG quality evaluation and benchmarking already exist. TripleCheckMate [346] is a crowdsourcing tool that allows users to evaluate single resources in an RDF KG by annotating found errors with 1 of 17 error classes. Another human-in-the-loop approach was proposed by NELL, where a user validated learned extraction patterns after a certain number of iterations [347]. RDFUnit [348] is an evaluation tool for validating and testing RDF graphs against predefined quality constraints and patterns. It can assess the quality and compliance of RDF datasets concerning schema definitions, vocabulary usage, and data integrity (supporting SHACL). The tool enables the automatic generation of tests by analyzing the structure of schemata, like ontologies or vocabularies, and generating test cases based on defined rules or patterns. Additionally, the tool allows users to define custom validation rules or include existing vocabularies and ontologies for validation purposes.

Hobbit [349] (Holistic Benchmarking of Big Linked Data) is a platform that facilitates the benchmarking of Linked Data systems and components. It provides a standardized framework for evaluating and comparing algorithms and approaches used in processing linked datasets. Key features include configuring benchmarking workflows, evaluating performance metrics, visualizing results, and supporting reproducibility and the sharing of benchmarks.

Benchmark datasets exist for specific subtasks of KG construction, such as entity resolution (e.g., Gollum [350]) and knowledge completion (e.g., CoDEx [351]). While there are several benchmark datasets available that focus on specific subtasks of the construction process, there is a lack of widely used end-to-end benchmark datasets, and researchers often create custom datasets or use subsets of existing datasets to evaluate their construction pipeline. Recent works utilize LLMs to generate synthetic datasets for benchmarking a wide range of data integration tasks [352]. Text2KGBench [353] is a benchmark encompassing two datasets to evaluate LLM-driven KG generation from text and ontologies. LLMKG-Bench [354] is an extensible framework to benchmark LLM-assisted knowledge engineering for syntax and error correction, fact extraction, and KG generation. More LLM-related quality assurance approaches are discussed in Section 4.6 about knowledge completion.

5. KG Systems

We now investigate and compare construction pipelines for existing KGs and for KG construction toolsets with respect to the KG requirements and construction steps introduced in the previous sections. The KG-specific approaches focus on integrating data from a rather fixed set of data sources for a single KG, while the toolsets (or strategies) are more generic and can be applied to different sources and KGs. Overall, we consider 16 KG-specific approaches (with a focus on ten semi-automatic and open implementations) and eleven toolsets. In the first subsection, we give an overview of the different approaches, including data statistics for the respective KGs and characteristics of the data sources and their construction pipelines. This overview aims at providing a good assessment of the current state of the art.

Given the enormous and growing number of KGs, we had to restrict ourselves to a small amount of effort. In our selection, we try to cover popular KGs such as DBpedia and Yago, as well as more current approaches for either a single domain or several domains (cross-domain). Most importantly, we focus on KG projects described in peer-reviewed articles and discuss **closed KGs** only briefly as their data are not publicly accessible, and the

techniques used are not verifiable. Such closed KGs are typically developed and used in companies such as company-specific enterprise KGs [355] and the KGs of big web and IT companies such as Google [31], Amazon [40], Facebook, Microsoft [356], Tencent, or IBM. However, open and easy-to-use KG toolsets are still in their infancy. Here, we tried to include recently described approaches that have already been applied to create several KGs, including those for a specific domain or a single data type.

In order to obtain a representative sample of the state-of-the-art, we employed a keyword-based search on the DBLP dataset in academic search online engines (e.g., SemanticScholar) and Github to gather all papers and approaches (systems and toolsets) that might fit our criteria. We also relied on existing surveys to gather potentially missed approaches. After a manual selection process via the paper titles and abstracts and comparison with our requirements, we created a list of candidate systems. This methodology is in line with other surveys [20]. After closer inspection with respect to the coverage of our requirements and filtering the approaches by age and availability of documentation or publication, we were left with the 21 works that are described in detail here. We expect that our comparison criteria and methodology will also be useful in evaluating KG-specific and more generic construction approaches not covered in this paper.

5.1. Specific KGs

Table 3 summarizes general characteristics of the selected KGs, which are grouped into *closed* and *open access* KGs and in each group ordered by their year of announcement or first publication. The table also displays the KG's targeted domain, number of data sources processed, underlying data model, graph size (number of entities, relations, entity types, and relation types), number of versions, and year of last update. Table 3 excludes the toolset projects as these are not restricted to a single KG. The values in this table were obtained from the most recent available version, either from the publication or directly from the dataset. The date of this version is denoted in the "Update" column in the table.

Table 3. Overview of selected KGs. "*" in the first column indicates manually curated (crowd-sourced) KGs. "?" means unknown/undisclosed values. The statistics include the KG's year of announcement, targeted domain, processed number of data sources, KG data model, graph size, number of versions, and year of last update. Domain abbreviations: Cross = cross-domain, MLang = multi-lingual data.

	Year	Domain	Srcs.	Model	Entities	Relations	Types	R-Types	Vers.	Update
Closed KG										
Google KG [31]	2012	Cross, MLang	>>>1	Custom, RDF	1B	>100B	?	?	?	?
Diffbot.com	2019	Cross	>>>1	RDF	5.9B	>1T	?	?	?	?
Amazon PG [40]	2020	Products	>1	Custom	30M	1B	19K	1K	?	?
Open Access KG										
* Freebase [357]	2007	Cross	>>1	RDF	22M	3.2B	53K	70K	>1	2016
DBpedia [108]	2007	Cross, MLang	140	RDF	50M	21B	1.3K	55K	>20	2023
YAGO [358,359]	2007	Cross	2–3	RDF(-Star)	67M	2B	10K	157	5	2020
NELL [347]	2010	Cross	≥1	Custom, RDF	2M	2.8M	1.2K	834	>1100	2018
* Wikidata [360]	2012	Cross, MLang	>>>1	Custom, RDF	100M	14B	300K	10.3K	>100	2023
DBpedia-EN Live [361]	2012	Cross	1	RDF	7.6M	1.1B	800	1.3K	>>>1	2023
Artist-KG [362]	2016	Artists	4	Custom	161K	15M	>1	18	1	2016
* ORKG [363]	2019	Research	>>1	RDF	130K	870K	1.3K	6.3K	>1	2023
AI-KG [364]	2020	AI Science	3	RDF	820K	1.2M	5	27	2	2020
CovidGraph [35]	2020	COVID-19	17	PGM	36M	59M	128	171	>1	2020
DRKG [34]	2020	BioMedicine	>7	CSV	97K	5.8M	17	107	1	2020
VisualSem [365]	2020	Cross, MLang	2	Custom	90K	1.5M	(49K)	13	2	2020
WorldKG [366]	2021	Geographic	1	RDF	113M	829M	1176	1820	1	2021

The table includes three manually curated projects based on crowdsourcing, namely the well-known Freebase and Wikidata approaches, as well as the newer Open Research Knowledge Graph (ORKG). Freebase [357,367] was one of the first collaboratively built

and versioned KGs and, after its shutdown in 2016, became a popular source for creating several other KGs, like Wikidata [360]. Wikidata supports entity annotation with key–value pairs, including validity time, provenance, and other metadata, such as references [368]. It facilitates semi-automatic curation involving both bots and human curators. As a project of the Wikimedia Foundation, full data dump snapshots are released twice a month. The ORKG [363] focuses on publications where manually uploaded papers are automatically enriched with metadata. The platform provides tools to extract information such as tables and figures from publications and to help find and compare similar publications of interest.

Most of the considered KGs are based on RDF, while some use a property graph or custom graph data model (fifth column in Table 3). Regarding the covered domains, the selected KGs either integrate sources from different domains (cross-domain) or focus on a single domain, such as research, biomedicine, or COVID-19. A possible limitation of cross-domain KGs, especially for smaller-sized ones, is that they can miss domain-specific details or expert knowledge. Some KGs contain and connect multilingual information (MLang) by providing descriptive entity values in different languages. These translations are mostly taken directly from one of the sources (e.g., Wikipedia or BableNet), instead of generating their own translations during the construction process. There are large differences among the KGs regarding the number of integrated source datasets (from 1 to 140) and the size of the KGs in terms of number of entity and relation types and the number of entities and relations. With the highest number of sources, DBpedia independently extracts 140 sources (Wikipedias), with equivalent entities being interlinked by the extracted `sameAs` connections contained in the page articles. The closed KGs are by far the largest, with up to almost 6 billion entities and more than a trillion relations (Diffbot.com). Wikidata is the largest open-source KG with about 100 million entities of 300K entity types and 14 billion relations of 300K relation types. The smallest KGs have less than 1 million entities or relations. In general, open KGs are rather limited in the number and diversity of the data sources, while closed approaches such as Google KG aim to integrate information at the web scale. Only a few of the KG projects continuously release updated versions of their KG, while most projects only release data once or irregularly every few years. This underlines that the continuous maintenance of KGs is not yet commonplace. With over 1100 dumps (http://rtw.ml.cmu.edu/rtw/resources (accessed on 15 August 2024)) NELL features the highest number of continuously and incrementally generated KG versions.

5.2. KG Frameworks

We now turn to a closer inspection of the KG construction processes of the individual KGs and toolsets. Table 4 summarizes the corresponding information for the 10 open-access KGs with semi-automatic construction, as well as for 11 toolsets/strategies for KG construction. We derived the concrete set of comparison criteria from our previously specified KG requirements in Section 3, such as support for incremental updates and different input data. Other criteria relate to the individual construction tasks from Section 4 that are necessary to meet the requirements, e.g., to support certain kinds of input data (e.g., Knowledge Extraction or entity resolution tasks) or to meet the requirement of quality assurance (tasks of input cleaning, quality assurance, and knowledge completion).

In Table 4, we weighted the criteria with regards to their fulfillment/presence in a specific solution by indicating strong approaches (considering automation, quality, and flexibility), with a full circle, and weaker approaches, with an open circle symbol, compared to the other approaches. We also provide information on the year of the considered version (publication) and indicate whether the approach offers an open implementation. We see that the pipeline/toolset implementations for two of the open-access KGs (NELL, AI-KG) and even five of the eleven toolsets are closed-source, including the approaches from Amazon (AutoKnow) and Apple (SAGA).

Table 4. Comparison of KG construction approaches with respect to *Consumed Data*, generated *Metadata*, and *Performed Construction Tasks*. The construction tasks are rated as *simple/manual* ○ or *sophisticated/semi-automatic* ● . '?' indicates *mentioned but unclear* implementation. Each criterion can cover multiple subtasks and implementations. However, due to the importance of fusion, we display it separately from entity resolution, indicated by the asterisk '*'.

		Consumed Data						(Meta) Data			Performed Construction Tasks								
Name of System	**System Version/Year**	**Open Implementation**	**Unstructured Data**	**Semi-Structured Data**	**Structured Data**	**(Event-)Stream Data**	**Supplementary Data**	**(Deep) Provenance**	**Version/Temporal Data**	**Additional Metadata**	**KG Initialization**	**Data Preprocessing**	**Ontology Management**	**Knowledge Extraction**	**Entity Resolution**	*** Entity/Value Fusion**	**Quality Assurnace**	**Knowledge Completion**	**Incremental Integration**
Dataset Specific																			
DBpedia	2019	✓		✓			✓	✓	✓	✓	○	●	○				●	○	
YAGO4	2020	✓		✓	✓		✓	✓	✓		○	○	●				●		
DBpedia-Live	2012	✓		✓		✓	✓	✓	✓		○	●	○						○
NELL	2011		✓	✓			✓	✓	✓		○		●	●			○		●
Artist-KG	2016	✓		✓	✓						○	○	●		●				●
AI-KG	2020		✓				✓	✓	✓		○			●			○		?
CovidGraph	2020	✓	✓	✓	✓		✓	✓	?		○	?	○	●	○				○
DRKG	2020	✓		✓	✓			✓			?		○	○		○		●	
VisualSem	2020	✓	✓	✓	✓		✓				○	●		●					
WorldKG	2021	✓		✓			✓				●	●	●				○		
Toolset/Strategy																			
FlexiFusion [103]	2019			✓	✓		✓	✓		✓	○	○				●			
dstlr [255]	2019	✓	✓					✓			○			●			○	○	?
XI [88]	2020		✓	✓				?	?	?	○			●			?		?
AutoKnow [40]	2020		✓	✓			✓				○	●	●	●				●	
HKGB [369]	2020		✓	✓			✓		✓		●		●	●	?		○	●	○
SLOGERT [47]	2021	✓		✓				✓	✓		○			●	?			○	?
SAGA [84]	2022		✓	✓	✓	✓	✓	✓	✓		?	●	○	●	●	●	●	●	●
Plumber [370]	2023	✓	✓				✓				○			●					
Image2Triplets [371]	2023	✓	✓				✓				○		○	●					
SAKA [109]	2023		✓	✓					✓		○	○	?	○					
Helio [372]	2024	✓		✓	✓		✓		?		?	○		○	○		?	?	?

5.3. Comparison

In the following, we describe and discuss the further criteria considered that fall into three groups regarding the consumed input data, the generated metadata, and the construction tasks performed.

Consumed Data. For the input data, we differentiate the supported kind of data (*unstructured*, *semi-structured*, *structured*) and consider support for stream input data and supplementary input data for further processing steps (e.g., metadata, mappings, but excluding tool configurations).

As Table 4 shows, populating KGs from semi-structured data is most common, while only about half of the considered solutions or toolsets support the import from unstructured or structured data. Scientific publications are a common unstructured data source (AI-KG, CovidGraph, DRKG, XI) as well as other forms of text (dstlr, SAGA, AutoKnow, Plumber). Several popular KGs (DBpedia, YAGO, NELL) integrate information from Wikipedia and use it as a premium source of valuable knowledge. Open accessible databases such as WordNet, ImageNet, or BabelNet are also frequent starting points for KG construction (e.g., YAGO or VisualSem). Only two of the projects (DBpedia Live and SAGA) support the continuous consumption of event streams. NELL continuously crawls the web for new data but updates the KG in a batch-like manner. Several approaches can ingest data from multiple modes, such as images (VisualSem, Image2Triplets) or audio (SAKA). Most approaches integrate supplementary data, especially mapping rules (DBpedia, DBpedia-

Live, YAGO, VisualSem, FlexiFusion, SAGA, Helio), training data (NELL), or quality constraints like SHACL shapes (YAGO).

Collected Metadata. We consider whether deep or fact-level provenance, temporal information (e.g., validity time), and additional metadata such as aggregated statistics, process reports, or descriptive and administrative information are collected and added to the KG or a separate repository.

The acquisition of provenance data is the most common kind of metadata support and ranges from simple source identifiers and confidence scores up to the inclusion of the original values. Several systems maintain temporal metadata, while further metadata are hardly supported or at least not described. In the case of the toolsets, the generation of additional metadata is possible in XI but depends on the use case and resulting pipeline. In general, support for metadata is thus limited and has room for improvement.

DBpedia, YAGO, AI-KG, and NELL store versions as single downloadable dumps. SLOGERT delegates and assigns the log timestamps through the pipeline for each constructed entity, and SAGA keeps timestamps for added, deleted, and updated entities. For SAKA, the system uses Neo4j to store and dynamically update Knowledge Graphs (KGs) while managing metadata in a relational database (SQLite). The relational database schema includes tables for KGs, entity types, and relations, ensuring each KG is uniquely identified and linked to its metadata. This combination allows for flexible updates and comprehensive management, including viewing, updating, and deleting KGs along with their associated data. In the DRKG, the final entity IDs contain the originating source, thereby supporting backtracking the origin of facts in the KG. CovidGraph contains provenance information about the originating paper, and nodes have modification timestamps.

Both the DBpedia release and the FlexiFusion Framework utilize the DBpedia Databus platform to maintain data artifacts using the DataID vocabulary. In addition to version information, they include information about the author and dataset descriptions. While the former only deploys intermediate or final results, FlexiFusion also accesses the Databus to consume the source data. For most of the other Frameworks or Datasets, it was unclear how intermediate data artifacts are managed. However, for the final KG, common Graph or RDF databases were utilized with some custom extensions at times.

Construction Tasks. In this group, we consider to what degree the construction tasks introduced in the previous section are supported.

- **KG Initialization.** Here, a common strategy is to manually create the initial KG either by developing it from scratch or by reusing existing KGs (ontologies). There may also be a complex pipeline to construct the initial KG by processing semi-structured data from catalogs, wikis, or category systems. All projects start with building or using some initial KG data. Most of the approaches reuse or sample existing KGs and ontologies as the initial target KG. WorldKG and HKGB semi-automatically build an initial ontology and are, therefore, more advanced than manual ontology construction. For DRKG, SAGA, and SAKA, it is unclear how the initial KG (ontology) was derived.
- **Data Preprocessing.** These are access methods and support the filtering, normalization, or correction of noisy input data. We exclude here NLP/text pre-processing as this is normally part of Knowledge Extraction. We tried to highlight the incorporation of multiple of these steps with the filled circle.
  Some approaches apply a filtering step to integrate only entities of relevant types into the KG. This functionality is not always provided (or documented) and is often based on manually defined rules and filter definitions, e.g., to select properties and relationships for certain entity types. Artist-KG links entities in the data source to the current Knowledge Graph to identify entities of relevant target types. VisualSem filters out nodes of images that do not meet the inclusion criteria, like valid images, near duplicates, and non-photographic images. YAGO filters low-coverage entities and accepts entities and their types that are transitively connected to one of the initial classes via sub-class relations, resulting in the final taxonomy of 10k classes (taxonomy enrichment).

Many of the investigated tools use a custom mapping approach to convert semi-structured data into a KG representation (DBpedia, YAGO, etc.) The flexible framework Helio requires providing an RDF mapping framework implementation. World KG maps and constructs RDF data based on the key–value pair-based tag system of Open Street Map. The FlexiFusion uses externally calculated entity and relation alignments and transforms the sources into an intermediate KG meta representation with IDs in the same namespace, keeping the initial source IDs as provenance. SAGA transforms input into a format in which the source, trust score, and one-hop relation information are extended to a triple. SAKA employs a generic mapping approach to convert key–value pairs from the source data into RDF and then allows the assignment of entity and relation types.

Lastly, some solutions also apply normalization steps during preparation, e.g., to unify date or number representations. In the case of DBpedia(-Live), the implementation recognizes value types and employs data parsers to normalize them into the same units or representations.

- **Ontology Management.** Most approaches have at least some basic (manual) support for evolving the KG ontology and schema data for newly structured input data. In DBpedia, the KG ontology (and data mappings) can be changed manually and need to be loaded before running a new batch update. The more freshness-oriented approach of DBpedia Live continuously watches ontology changes and immediately schedules affected entities for re-extraction. More advanced approaches rely on semi-automatic ontology evolution or enrichment. In particular, some systems can identify new entities and relation types in the input data to add to the ontology after manual confirmation (NELL, HKGB). Image2Triplets can fully automatically add newly recognized entities or relations to the KG but reserve human intervention for specific edge cases where the system alone cannot decide the manner of integration. ArtistKG uses Karma for ontology matching. While merging is mentioned, it is unclear what procedure is used. While WorldKG, for example, relies on an unsupervised ML approach for ontology alignment, most approaches still perform ontology alignment and merging manually. SAGA's ingestion component requires mappings from new data to the internal KG ontology. This step only requires predicate mappings, as the subject and object fields can remain in their original namespace and are linked later in the process. CovidGraph performs the mapping of biomedical ontology terms based on their IDs.
- **Knowledge Extraction.** Many solutions use rule-based methods to extract entities and relations from semi-structured sources (DRKG, VisualSem). Some tools use machine learning approaches for extraction (AI-KG, AutoKnow, CovidGraph, dstlr, SLOGERT, NELL). For entity linking, different approaches are used, such as dictionary-based approaches relying on gathered synonyms (e.g., AI-KG), the use of human interaction (XI), or the application of entity resolution (e.g., HKGB). Plumber selects approaches from a combination of 33 different methods for NER, RE, and EL using an ML approach on dataset samples and provided metadata. A few approaches have a multi-modal domain of extraction. Image2Triplets and VisualSem extract information from images. Image2Triplets uses computer vision techniques to extract visual relationships from images. They also determine human–object interaction in images, detecting novel objects and actions through zero-shot learning. VisualSem, on the other hand, only allows pre-defined relations. SAKA first segments audio files based on speakers and removes non-speech segments. They then transform this audio into text and then perform Knowledge Extraction on it. Given the focus on semi-structured data sources, Knowledge Extraction techniques are generally relatively advanced compared to other steps in KG construction. This has also been made possible by the frequent use of existing Knowledge Extraction tools, such as Stanford CoreNLP, as will be seen in the discussion of the approaches in the next subsections.

- **Entity Resolution.** This task is supported by only a few approaches, and the pipelines that do employ ER tend to use sophisticated methods like blocking to address scalability issues (ArtistKG, SAGA), and machine-learning-based matchers (SAGA). HKGB's description of their ER solution is too vague to make a definite statement, and for SLOGERT, it is mentioned that in some cases, ER might be necessary but should be performed with an external tool. Similarly, Helios could enable any ER method, but they only briefly mention this possibility in their requirements through the underlying plugin architecture. CovidGraph relies on string similarities and global identifiers to identify matches. For textual data, the identification and matching of entities to KG elements are already covered by entity linking in the Knowledge Extraction step (Section 4.4). Only SAGA and ArtistKG use blocking methods to scale the matching process.
- **Entity Fusion.** This is the least supported task among the considered solutions. None of the dataset-specific KGs perform classical (sophisticated) entity fusion, consolidating possible value candidates and selecting final entity IDs or values. Instead, the final KG often contains a union of all extracted values, either with or without provenance, leaving final consolidation/selection to the targeted applications. The DRKG project uses a simple form of entity fusion to normalize entity identifiers. Even for the discussed toolsets, this task's coverage is relatively low. FlexiFusion allows the application of specific fusion functions, leverages provenance information, and performs a stable ID assignment for entity and property clusters. SAGA refers to the usage of truth-discovery- and source-reliability-based fusion methods.
- **Quality Assurance.** Human-in-the-loop strategies have been applied to varying degrees, with some solutions, such as HKGB or XI, relying heavily on user interaction. In contrast, others require only final user approval of the correctness of extracted values or patterns, like NELL. In the World, the KG approach manually verifies all class and predicate matches to the external ontologies.
  Further, SAGA tries to detect potential errors or results of vandalism automatically. It quarantines them for human curation, where changes are treated directly in the live graph and later applied to the stable graph. AI-KG bases the validity of a triple on the trustworthiness of the extraction tool, the frequency of that triple being extracted reaching a certain threshold, or a specifically trained classifier deciding that it is valid. DBpedia and YAGO perform an automatic consistency check. The Helio paper mentions that in a specific use case, their approach was extended to use a validation mechanism, which they do not specify in more detail. Additionally, YAGO guarantees ontological consistency by applying a logical reasoner, and DBpedia checks for dataset completeness and measures quality against the former version.
  In our study, only dstlr supports validating extracted facts against an external knowledge base.
- **Knowledge Completion.** DBpedia attaches additional entity-type information based on current ontology and relation data. Three approaches (DRKG, HKGB, SAGA) presented ML-based link prediction on graph embeddings to find further knowledge. In the case of the DRKG and HKGB approaches, it is unclear if the newly predicted information flows back into the KG or is stored separately. AutoKnow uses a learning-based approach to categorize product types.
  Regarding enrichment with external knowledge, dstlr links entities to Wikidata and fetches stored properties from this external source. However, SLOGERT only adds links to external information based on previously extracted identifiers (PIDs).

Incremental Solutions. Some approaches mention incremental KG generation but do not describe it in detail. AI-KG mentions the generation of updated versions, but the steps are unclear. CovidGraphs' dockerized approach allows for ingesting new data sources into the KG; however, a direct solution for this has not been explained. For dstlr, the authors mention the ability to track document changes via Apache Solr, but they do not specify any ways to deal with such changes. The same is true for SLOGERT, as the authors only mention

that update mechanisms will be addressed in future work. In the case of the XI pipeline, support for this feature was considered in the final implementations. Helio's versioning concept allows for comparing different KG versions and building new increments, but its pipeline implementation is not entirely clear as it depends on task-specific implementations such as mapping, matching, and fusion.

Few approaches describe manual or straightforward update strategies for changes in sources or ontology. For instance, DBpedia-Live continuously integrates Wikipedia changes into a live graph, but it only retains the latest version and lacks quality control. Extending the HKGB with new data and sources about additional diseases involves several manual steps, including collecting synonyms, checking candidates, annotating concepts, annotating disease-related concepts and relations from unstructured data, and creating mapping and extraction rules.

Only three of the frameworks evaluated have a good approach with respect to implementing a maintainable incremental process. NELL is truly focused on providing an incremental solution for KG construction from unstructured web text. It provides a continuous learning method and a human-in-the-loop approach for the quality assurance of learned extraction patterns. ArtistKG allows for the integration of new sources iteratively, adding one source at a time. SAGA maintains a continuous live graph and a batch KG, with delta computation and source provenance. It combines automatic and manual quality assurance but is closed-source, making its approach not fully transparent or verifiable. Overall, from these pipelines, SAGA provides the most advanced incremental solution. It offers an extendable multi-source approach capable of handling various heterogeneous data updates. This includes combining Knowledge Extraction and entity resolution with fusion capabilities while still supporting continuous quality assurance.

6. Discussion and Open Challenges

Our study of existing solutions for KG construction showed that there are many different approaches not only for building specific KGs but also in the current toolsets. This underlines the inherent complexity of the problem and the dependency on different criteria, such as the major kinds of input data and the intended use cases. The requirements we posed in Section 3 are also not yet met to a larger degree, indicating the need for more research and development efforts. This is also because there are inherent tradeoffs between the goals of high data quality, scalability, and automation [15] that ask for compromise solutions. So while it is possible to have a large degree of automation for individual construction tasks, human interaction generally tends to improve the quality significantly. On the other hand, such human interaction can become a limiting factor toward scalability to many sources and high data volume.

Below, we discuss open challenges and areas for future work on KG construction. The focus is on broader issues rather than specific limitations in individual steps.

Incremental KG Construction. We observed that most of the construction pipelines for specific KGs and in toolsets do not yet support incremental KG updates but are limited to a batch-like re-creation of the entire KG. As already discussed in Section 3, this approach has significant limitations and prevents scalability to many data sources and high data volume. We, therefore, need better support for incremental KG updates, especially in toolsets. Such a capability has to provide solutions to a variety of issues. As already discussed in Section 4.2, it must be detected *if* there are changes in the input data, and if so, it must be determined *what* has changed. The changes to be dealt with are not limited to adding new information; deletions and updates in the sources have to be propagated to the KG as well. Changes that impact the underlying ontology or the pipeline's configuration also have to be managed and may require manual interaction/confirmation. Inferred knowledge may become inconsistent with newly introduced data. While there are incremental reasoning approaches, that aim to address this problem, they need to be integrated with the KG construction pipeline holistically [373,374].

Support for a streaming-like propagation of source changes should also increase so that KGs can provide the most recent information.

Lack of open tools. As we have seen in Table 4, most KG construction toolsets, especially the more advanced ones, are closed-source and thus cannot be used by others for creating a KG or for evaluating their functionality. Hence, there is a strong need for more open-source toolsets to help improve the development of KGs and to advance the state of the art in KG construction. Researchers and developers providing such an implementation and associated publications could have a high impact [375].

Improved Extensibility and Modularization, Ease of Use. A toolset for KG construction should be able to define and execute different pipelines depending on the data sources to be integrated and specific requirements, e.g., for incremental updates. Hence, an extensible and modular KG construction approach should be provided with alternate implementations for the different KG construction tasks to choose from. This can be facilitated by using existing implementations, as has been carried out already for NLP tasks (e.g., Stanford CoreNLP) but not yet for other tasks such as entity resolution. From the projects compared in Section 4, only a few addressed this problem, so more solutions are needed.

The definition of a KG construction pipeline should be relatively easy, supported by a user-friendly GUI, and have a low effort for configuring the pipeline and its individual tasks. The configuration can be simplified by providing default settings for most parameters or even automatic approaches based on machine learning [370,376]. On the other hand, a manual configuration should also be possible to achieve customizability and support for special cases (e.g., a new entity type or input format). The extensibility and modularization of a tool should not lead to a higher configuration effort for users. For example, Helio [372] provides a flexible system but lacks a default configuration, still requiring extensive manual configuration.

Data and Metadata Management. Good data and metadata management is vital in an open and incremental KG process. Only a few solutions even mention an underlying management architecture supporting the construction processes. Having uniform access or interfaces to data and relevant metadata can drastically improve the quality of the former [94] and increase the workflow's replicability and possibilities for debugging. A dedicated metadata repository can store used mappings, schemata, and quality reports, improving the transparency of the entire pipeline process.

Metadata support is limited in current solutions, and only some pipeline approaches acknowledge the importance of provenance tracking and debugging possibilities. We found that the term *provenance* is rather vaguely used, mostly referring to tracking the source of facts and the processes involved in their generation. Only a few approaches, such as SAGA [84], also try to maintain the trustworthiness of facts. Metadata, such as fact-level provenance, should be used more to support construction tasks. For example, in data fusion, it can be used to determine final entity values. In general, there is a need for maintaining more metadata, especially temporal information, that is also essential for studying the evolution of KG information. Support for developing temporal KGs maintaining historical and current data, compared to the common sequences of static KG snapshot versions, is also a promising direction. Temporal KGs would then also enable the possibility of temporal reasoning [377].

Data Quality. One of the main goals of KG construction is to achieve and maintain a high-quality KG. The difficulty of this task grows with the rising number and heterogeneity of data sources, especially if one relies on automatic data acquisition and data integration. High-quality sources can provide a clean hierarchy of types and can serve as training data to alleviate some data-quality issues that would be more difficult to address by treating low-quality sources in isolation [15]. Lower-quality data sources often contain a high degree of long-tail entities (which is the reason these data sources are valuable). Nevertheless, corroborating the information from these sources with evidence from higher-quality sources remains difficult and can reduce data quality. For example, the automatic

fusion of conflicting entity values can easily introduce wrong information into a KG, and even a restricted degree of human intervention is problematic on a large scale [217,378].

To achieve the best possible data quality, data cleaning should be part of all major steps in the construction pipeline so that the degree of dirty or wrong information that is entering the KG is limited. Moreover, the identification and repair of errors should be a continuous task, especially in projects with large KGs [379]. To better address these problems, more comprehensive data quality measures and repair strategies are needed that minimize human intervention to retain high scalability for KG construction.

While one of KGs' main strengths is its reasoning capabilities, it is intuitive that these capabilities are strongly correlated with data quality. Heavily updated KGs could require automatic consistency checks and mechanisms to resolve inconsistencies. The introduced overhead of such data quality assurances would need to be balanced with the desired timeliness of the KG.

Evaluation. The evaluation of complete KG construction pipelines is an open but important problem to measure the performance and quality of current approaches and to improve on them. So far, there are benchmarks for individual tasks such as Knowledge Extraction [380–382], ontology matching [383], entity resolution [268,269,350,384] and KG completion [351,385,386]. While these benchmarks, in some cases, still leave gaps, e.g., regarding scalability [263] or domain diversity [387], they are already quite complex and indicate the great difficulty in defining a benchmark for the entire KG construction pipelines.

A benchmark could be based on similar settings to those used for the creation of specific KGs discussed in Section 4, aiming at the initial construction and incremental update of either a domain-specific or cross-domain KG from a defined set of data sources of different kinds. The KG ontology and the KG data model (RDF or property graph) could be predefined to facilitate the evaluation of the resulting KG. The size of the resulting KG should be relatively large, and the construction should be challenging with the need for Knowledge Extraction, entity linking/resolution, and entity fusion. Determining the quality of the constructed KG is difficult, as it would ideally be based on a near-perfect result (gold standard) for the initial KG and for its updated version(s). For all entity and relation types in the given KG ontology, it has then to be determined to what degree they could correctly be populated compared to the gold standard, which requires an extension to known metrics such as precision and recall. Further evaluation criteria include the runtimes for the initial KG construction, the incremental updates, and perhaps the manual effort to set up the construction pipelines. Ideally, an evaluation platform could be established—similar to other task-specific platforms like Hobbit [349]—for a comparative evaluation of different pipelines with different implementations of the individual construction steps.

The whole is more than the sum of its parts. While the individual parts of KG construction pipelines are well-established research problems with sometimes decades of previous research, the complex interaction of the pipeline tasks is not well researched yet. For example, the disambiguation strategies of the Knowledge Extraction task, especially entity linking, are very similar to entity resolution. The use of background knowledge and various inter-dependencies between different information is commonly summarized as *holistic entity linking*. This approach has seen some research attention, and a survey with future research directions was published by Oliveira et al. in a 2021 paper [388]. While such approaches go in the right direction, our pipeline scenario would invite an even more holistic case, where named-entity linking and entity resolution approaches aid each other in boosting their performance. Furthermore, data cleaning can be performed independently in several tasks, but it would be beneficial to have a coordinated approach to avoid duplicate efforts.

LLM for KG Construction. LLMs offer significant advancements in automating and enhancing natural language understanding, which is helpful and promising for bridging the gap between users and KG construction tasks to reduce the amount of manual involvement. Initial approaches showed that LLMs could support several KG construction tasks, e.g., data transformation and cleaning, ontology development, entity resolution,

and quality assurance and completion. The proposed approaches are mostly investigated in isolation and not for KG construction specifically, so there is a need for solutions tailored to KG construction [330]. LLMs also introduce complexities [389] such as their inherent bias and limited domain-specific knowledge, which can hinder their effectiveness in certain areas and cause phenomena such as *hallucination*. Moreover, continuous operation and interaction with LLMs is often cost- and time-intensive, potentially affecting overall scalability. A promising approach is using LLMs as a decision support agent for pipeline orchestration by choosing the correct function or tool for the next processing step based on a given sample of data and source metadata [370]. Another solution would be to utilize LLMs for generating configurations for tools and a given data source [390]. Furthermore, the different LLM-enhanced construction tasks must be effectively combined within a complete construction pipeline, requiring further research and development [12]. A more general objective is the comprehensive combination of LLMs and KGs not only where LLMs help in KG construction but also where KGs help to improve LLMs, e.g., to provide up-to-date and verifiable information instead of hallucinated answers. Research in this direction has already begun (for example, [391]). There are indications that the widely praised performance of LLMs on various tasks can largely be attributed to memorization rather than reasoning [392]. Here, the reasoning capabilities of KGs could be a valuable complement to the generative abilities of LLMs.

7. Related Work

The construction of KGs uses technologies from different areas, and we have discussed the tasks and surveys in these areas already in Section 4. Here, we therefore focus on related surveys on the construction of KGs in general.

In almost 300 pages, Weikum et al. [15] give an extensive tour of the automatic creation and curation of RDF-based knowledge bases or KGs, specifically from semi- and unstructured sources. Their discussion of requirements is also concerned with the KG itself, whereas our requirements are more focused on the KG construction process. We also cover structured input data for KG construction, e.g., in the requirements on *Input Data* and tasks such as entity resolution. Their article provides overviews about the open Knowledge Graphs YAGO, DBpedia, NELL, and Wikidata, which are also discussed in our work, as well as industrial Knowledge Graphs, which we only mention briefly due to the limited amount of publicly available information. By contrast, we systematically compare many further approaches with respect to our derived requirements, including general KG construction toolsets. Furthermore, we have identified several new challenges for future work, e.g., regarding incremental approaches, open toolsets, and benchmarks.

Hogan et al. [8] give a comprehensive introduction to KGs. Similar to ours, their discussion includes multiple graph data models, and they present methods for dealing with unstructured, semi-structured, and structured data. Serving as an introductory text to KGs in general, their work provides a broad view on KGs, including tasks like learning or publishing them. We are more focused on KG construction and cover many additional aspects such as requirements for KG construction and maintenance, a more detailed discussion of construction tasks, a systematic comparison of state-of-the-art approaches, as well as open challenges for KG construction.

Ryen et al. [17] provide a systematic literature review on KG creation approaches based on Semantic Web technologies. They survey and compare 36 approaches with respect to their identified construction steps: ontology development, data preprocessing, data integration, quality and refinement, and data publication. One of their findings was that data quality appears to be a major blind spot. We more comprehensively investigate the requirements, approaches, and open challenges of KG construction and maintenance and also include other KG data models, such as the PGM.

Tamašauskaitė et al. [393] propose a KG development lifecycle consisting of six main steps with several possible subtasks. Our work covers their construction tasks and feasible solutions in more detail and complements them with other relevant main tasks, such as

metadata management and the discussion of temporal aspects and versioning. Furthermore, our survey and evaluation of KG construction approaches are based on a set of requirements that led to the tasks not covered in their approach. We also provide a comparison of many construction approaches and toolsets and identify open challenges not covered in their work.

There are several papers on the construction of KGs for specific cases. Zhu et al. [16] focus on the creation of multi-modal KGs, specifically combining the symbolic knowledge in a KG with corresponding images. They comprehensively present the two directions in which this task can be performed: visual Knowledge Extraction in order to label images with information from the KG and discovering images that describe entities and relations from the KG. In our work, we aim to discuss the KG construction process more broadly and in a complementary manner, only briefly discussing multi-modal (image-related) techniques. Xiaogang Ma reviews applications and construction approaches for KGs in the geoscience domain [18]. The discussed KG creation methods range from mostly manual approaches to processes relying on the data mining of crowdsourced data. Furthermore, they discuss how KGs are used in geoscience data analysis, e.g., to enhance information extraction for public health hazards. Şimşek et al. [394] give a high-level overview of the KG construction process in the general context of a KG's lifecycle. Their discussion of these general steps is found alongside an in-use case study, where they provide the challenges they encountered. While they give valuable insights into KG construction in the real world, they do not include a systematic comparison of the state-of-the-art approaches regarding the requirements of KG construction.

Abu Salih [13] conducted an extensive survey on over 140 papers focused on domain-specific KGs. The work proposes a refined definition of domain-specific Knowledge Graphs, categorizes the approaches into one of seven domains, and highlights the diversity of methods used in constructing domain-specific KGs. As a result, the work identifies limitations in current approaches and proposes future research to address research gaps in this area.

In summary, our work focuses more on KG construction than previous KG surveys and provides additional information in several areas related to KG construction. We are not limiting ourselves to RDF-based KGs but also consider alternate graph data models such as the PGM. Our study considers not only the acquisition and integration of unstructured and semi-structured data but also of structured data, and we not only investigate the one-time construction of KGs but also their incremental maintenance. In contrast to most previous surveys, we explicitly specify the main requirements for KG construction and use these as a guideline for evaluating and comparing many KG-specific construction approaches and toolsets and identifying new open challenges.

8. Conclusions

This work presented the current state of Knowledge Graph construction, giving an overview of the requirements and defining this area's central concepts and tasks. We gave a synopsis of techniques used to address individual steps of such a pipeline with a perspective on how well the state-of-the-art solutions for these specific tasks can be integrated into an incremental KG construction approach. We comparatively analyzed a selection of current KG-specific pipelines and toolsets for KG construction based on a list of criteria derived from our initial requirements. We found vast differences across these pipelines concerning the number and structure of the input data, applied construction methods, Ontology Management, the ability to continuously integrate new information, and the tracking of provenance throughout the pipeline. The open KG-specific approaches are currently rather limited in their scalability to many sources, support for incremental updates, and several steps regarding metadata, Ontology Management, entity resolution/fusion, and quality assurance. The considered toolsets are generally better in terms of their functionality, but they are mostly closed-source and thus not usable for new KG projects or research investigations.

We identified several challenges to address for improved incremental KG construction. These problems range from engineering questions, like the need for a flexible software architecture, through numerous task-specific problems and the support for incremental construction, to hurdles that must be addressed collectively by the research community, like the development of open-source and modular toolsets for KG construction and benchmarking and evaluation processes. Concerning the exploitation of new data sources, integrating more multimodal data is of great potential but also requires more research to achieve effective solutions. Moreover, the use of Large Language Models for KG construction should be investigated more and also optimized. Addressing the derived challenges promises significant advances for future KG construction pipelines and a considerable reduction in effort for creating and maintaining high-quality KGs.

Author Contributions: Conceptualization, M.H, D.O., A.S. and E.R.; Methodology, M.H., D.O., A.S. and E.R.; Investigation, M.H. and D.O.; Data Curation, M.H. and D.O.; Writing—original draft preparation, M.H, D.O., H.K. and A.S.; Writing—review and editing, M.H, D.O., H.K., A.S. and E.R.; Visualization, M.H., D.O. and A.S.; Supervision, E.R.; Project Administration, E.R.; Funding Acquisition, E.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Federal Ministry of Education and Research of Germany and by the Sächsische Staatsministerium für Wissenschaft Kultur und Tourismus in the program Center of Excellence for AI-research "Center for Scalable Data Analytics and Artificial Intelligence Dresden/Leipzig", project grant identification number: ScaDS.AI.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Huang, X.; Zhang, J.; Li, D.; Li, P. Knowledge Graph Embedding Based Question Answering. In Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, 11–15 February 2019; ACM: New York, NY, USA; pp. 105–113. [CrossRef]
2. Wang, X.; He, X.; Cao, Y.; Liu, M.; Chua, T. KGAT: Knowledge Graph Attention Network for Recommendation. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, 4–8 August 2019; ACM: New York, NY, USA; pp. 950–958. [CrossRef]
3. Mohamed, S.K.; Nováček, V.; Nounu, A. Discovering protein drug targets using Knowledge Graph embeddings. *Bioinformatics* **2019**, *36*, 603–610. [CrossRef]
4. Oberkampf, H.; Zillner, S.; Bauer, B. Interpreting Patient Data using Medical Background Knowledge. In Proceedings of the 3rd International Conference on Biomedical Ontology (ICBO 2012), KR-MED Series, Graz, Austria, 21–25 July 2012; Volume 897.
5. Sonntag, D.; Tresp, V.; Zillner, S.; Cavallaro, A.; Hammon, M.; Reis, A.; Fasching, P.A.; Sedlmayr, M.; Ganslandt, T.; Prokosch, H.; et al. The Clinical Data Intelligence Project—A smart data initiative. *Inform. Spektrum* **2016**, *39*, 290–300. [CrossRef]
6. Fan, R.; Wang, L.; Yan, J.; Song, W.; Zhu, Y.; Chen, X. Deep Learning-Based Named Entity Recognition and Knowledge Graph Construction for Geological Hazards. *ISPRS Int. J. Geo-Inf.* **2020**, *9*, 15. [CrossRef]
7. Nickel, M.; Murphy, K.; Tresp, V.; Gabrilovich, E. A review of relational machine learning for Knowledge Graphs. *Proc. IEEE* **2015**, *104*, 11–33. [CrossRef]
8. Hogan, A.; Blomqvist, E.; Cochez, M.; d'Amato, C.; de Melo, G.; Gutiérrez, C.; Kirrane, S.; Labra Gayo, J.E.; Navigli, R.; Neumaier, S.; et al. *Knowledge Graphs*; Synthesis Lectures on Data, Semantics, and Knowledge (SLDSK); Springer: Cham, Switzerland, 2022; ISBN 978-3-031-00790-3. [CrossRef]
9. Ji, S.; Pan, S.; Cambria, E.; Marttinen, P.; Philip, S.Y. A survey on Knowledge Graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *33*, 494–514. [CrossRef]
10. Pan, S.; Luo, L.; Wang, Y.; Chen, C.; Wang, J.; Wu, X. Unifying Large Language Models and Knowledge Graphs: A Roadmap. *IEEE Trans. Knowl. Data Eng.* **2024**, *36*, 3580–3599. [CrossRef]
11. Yang, L.; Chen, H.; Li, Z.; Ding, X.; Wu, X. Give Us the Facts: Enhancing Large Language Models with Knowledge Graphs for Fact-aware Language Modeling. *arXiv* **2023**, arXiv:2306.11489. [CrossRef]
12. Allen, B.P.; Stork, L.; Groth, P. Knowledge Engineering Using Large Language Models. *arXiv* **2023**, arXiv:2310.00637. [CrossRef]
13. Abu-Salih, B. Domain-specific Knowledge Graphs: A survey. *J. Netw. Comput. Appl.* **2021**, *185*, 103076. [CrossRef]
14. Zou, X. A Survey on Application of Knowledge Graph. *J. Phys. Conf. Ser.* **2020**, *1487*, 012016. [CrossRef]
15. Weikum, G.; Dong, L.; Razniewski, S.; Suchanek, F.M. Machine Knowledge: Creation and Curation of Comprehensive Knowledge Bases. *Found. Trends Databases* **2021**, *10*, 108–490. [CrossRef]
16. Zhu, X.; Li, Z.; Wang, X.; Jiang, X.; Sun, P.; Wang, X.; Xiao, Y.; Yuan, N.J. Multi-Modal Knowledge Graph Construction and Application: A Survey. *IEEE Trans. Knowl. Data Eng.* **2024**, *36*, 715–735. [CrossRef]

17. Ryen, V.; Soylu, A.; Roman, D. Building Semantic Knowledge Graphs from (Semi-) Structured Data: A Review. *Future Internet* **2022**, *14*, 129. [CrossRef]
18. Ma, X. Knowledge graph construction and application in geosciences: A review. *Comput. Geosci.* **2021**, *161*, 105082. [CrossRef]
19. Xiao, G.; Ding, L.; Cogrel, B.; Calvanese, D. Virtual Knowledge Graphs: An Overview of Systems and Use Cases. *Data Intell.* **2019**, *1*, 201–223. [CrossRef]
20. Assche, D.V.; Delva, T.; Haesendonck, G.; Heyvaert, P.; Meester, B.D.; Dimou, A. Declarative RDF graph generation from heterogeneous (semi-)structured data: A systematic literature review. *J. Web Semant.* **2023**, *75*, 100753. [CrossRef]
21. Schneider, E.W. *Course Modularization Applied: The Interface System and Its Implications For Sequence Control and Data Analysis*; Report HumBRO-PP-10-73; Human Resources Research Organization: Alexandria, VA, USA, 1973.
22. Paulheim, H. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semant. Web* **2017**, *8*, 489–508. [CrossRef]
23. Ehrlinger, L.; Wöß, W. Towards a Definition of Knowledge Graphs. In Proceedings of the Joint Proceedings of the Posters and Demos Track of the 12th International Conference on Semantic Systems—SEMANTiCS 2016 and the 1st International Workshop on Semantic Change & Evolving Semantics (SuCCESS'16) Co-Located with the 12th International Conference on Semantic Systems (SEMANTiCS 2016), Leipzig, Germany, 13–14 September 2016.
24. Lissandrini, M.; Mottin, D.; Hose, K.; Pedersen, T.B. Knowledge Graph Exploration Systems: Are we lost? In Proceedings of the 12th Conference on Innovative Data Systems Research, CIDR, Chaminade, CA, USA, 9–12 January 2022.
25. Hogan, A.; Brickley, D.; Gutierrez, C.; Polleres, A.; Zimmerman, A. (Re)Defining Knowledge Graphs. In Proceedings of the Knowledge Graphs: New Directions for Knowledge Representation on the Semantic Web (Dagstuhl Seminar 18371), Wadern, Germany, 9–14 September 2018; Volume 8, pp. 74–79. [CrossRef]
26. Feilmayr, C.; Wöß, W. An analysis of ontologies and their success factors for application to business. *Data Knowl. Eng.* **2016**, *101*, 1–23. [CrossRef]
27. Dentler, K.; Cornet, R.; ten Teije, A.; de Keizer, N. Comparison of reasoners for large ontologies in the OWL 2 EL profile. *Semant. Web* **2011**, *2*, 71–87. [CrossRef]
28. Abburu, S. A survey on ontology reasoners and comparison. *Int. J. Comput. Appl.* **2012**, *57*, 33–39.
29. Chen, X.; Jia, S.; Xiang, Y. A review: Knowledge reasoning over Knowledge Graph. *Expert Syst. Appl.* **2020**, *141*, 112948. [CrossRef]
30. Kejriwal, M. *Domain-Specific Knowledge Graph Construction*; Springer Briefs in Computer Science (BRIEFSCOMPUTER); Springer: Cham, Switzerland, 2019; ISBN 978-3-030-12374-1. [CrossRef]
31. Noy, N.; Gao, Y.; Jain, A.; Narayanan, A.; Patterson, A.; Taylor, J. Industry-scale Knowledge Graphs: Lessons and Challenges: Five diverse technology companies show how it's done. *Queue* **2019**, *17*, 48–75. [CrossRef]
32. Song, Y.; Li, W.; Dai, G.; Shang, X. Advancements in Complex Knowledge Graph Question Answering: A Survey. *Electronics* **2023**, *12*, 4395. [CrossRef]
33. Liu, J.; Huang, W.; Li, T.; Ji, S.; Zhang, J. Cross-Domain Knowledge Graph Chiasmal Embedding for Multi-Domain Item-Item Recommendation. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 4621–4633. [CrossRef]
34. Ioannidis, V.N.; Song, X.; Manchanda, S.; Li, M.; Pan, X.; Zheng, D.; Ning, X.; Zeng, X.; Karypis, G. DRKG—Drug Repurposing Knowledge Graph for COVID-19. 2020. Available online: https://github.com/gnn4dr/DRKG/blob/1a3141e71fbbd2ffa97d91a91ad4d12754dc7bd6/DRKG.
35. Preusse, M.; Jarasch, A.; Bleimehl, T.; Muller, S.; Munro, J.; Gutebier, L.; Henkel, R.; Waltemath, D. COVIDGraph: Connecting Biomedical COVID-19 Resources and Computational Biology Models. In Proceedings of the 2nd Workshop on Search, Exploration, and Analysis in Heterogeneous Datastores (SEA-Data 2021) Co-Located with 47th International Conference on Very Large Data Bases (VLDB 2021), Copenhagen, Denmark, 20 August 2021; Volume 2929, pp. 34–37.
36. Su, X.; You, Z.; Huang, D.; Wang, L.; Wong, L.; Ji, B.; Zhao, B. Biomedical Knowledge Graph Embedding With Capsule Network for Multi-Label Drug-Drug Interaction Prediction. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 5640–5651. [CrossRef]
37. Kertkeidkachorn, N.; Nararatwong, R.; Xu, Z.; Ichise, R. FinKG: A Core Financial Knowledge Graph for Financial Analysis. In Proceedings of the 17th IEEE International Conference on Semantic Computing, ICSC 2023, Laguna Hills, CA, USA, 1–3 February 2023; pp. 90–93. [CrossRef]
38. Reinanda, R. Financial Knowledge Graph at Bloomberg: Applications and Challenges. In Proceedings of the Knowledge Graph Conference (KGC) 2021— KGC, Virtual, 3–6 May 2021. [CrossRef]
39. Abu-Salih, B.; Alotaibi, S. Knowledge Graph Construction for Social Customer Advocacy in Online Customer Engagement. *Technologies* **2023**, *11*, 123. [CrossRef]
40. Dong, X.; He, X.; Kan, A.; Li, X.; Liang, Y.; Ma, J.; Xu, Y.; Zhang, C.; Zhao, T.; Saldana, G.B.; et al. AutoKnow: Self-Driving Knowledge Collection for Products of Thousands of Types. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD), Virtual, 26 July 2020.
41. Lee, J.; Yoon, W.; Kim, S.; Kim, D.; Kim, S.; So, C.H.; Kang, J. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* **2019**, *36*, 1234–1240. [CrossRef]
42. Trabelsi, M.; Heflin, J.; Cao, J. DAME: Domain Adaptation for Matching Entities. In Proceedings of the WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Tempe, AZ, USA, 21–25 February 2022; pp. 1016–1024. [CrossRef]

43. Balsebre, P.; Yao, D.; Cong, G.; Hai, Z. Geospatial Entity Resolution. In Proceedings of the ACM Web Conference 2022, Lyon, France, 25–29 April 2022; pp. 3061–3070. [CrossRef]
44. Ngomo, A.N. ORCHID—Reduction-Ratio-Optimal Computation of Geo-spatial Distances for Link Discovery. In Proceedings of the Semantic Web—ISWC 2013—12th International Semantic Web Conference, Sydney, Australia, 21–25 October 2013; Volume 8218, pp. 395–410. [CrossRef]
45. Cui, Z.; Sun, X.; Pan, L.; Liu, S.; Xu, G. Event-Based Incremental Recommendation via Factors Mixed Hawkes Process. *Inf. Sci.* **2023**, *639*, 119007. [CrossRef]
46. Wang, P.; He, Y. Uni-Detect: A Unified Approach to Automated Error Detection in Tables. In Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, 30 June–5 July 2019; pp. 811–828. [CrossRef]
47. Ekelhart, A.; Ekaputra, F.J.; Kiesling, E. The SLOGERT Framework for Automated Log Knowledge Graph Construction. In Proceedings of the ESWC, 2021, Virtual, 6–10 June 2021.
48. Sakr, S.; Bonifati, A.; Voigt, H.; Iosup, A.; Ammar, K.; Angles, R.; Aref, W.G.; Arenas, M.; Besta, M.; Boncz, P.A.; et al. The future is big graphs: A community view on graph processing systems. *Commun. ACM* **2021**, *64*, 62–71. [CrossRef]
49. Lassila, O. Resource Description Framework (RDF) Model and Syntax Specification, W3C Recommendation. 1999. Available online: http://www.w3.org/TR/PR-rdf-syntax (accessed on 18 August 2024).
50. Horrocks, I.; Patel-Schneider, P.F.; Boley, H.; Tabet, S.; Grosof, B.; Dean, M. SWRL: A semantic web rule language combining OWL and RuleML. *W3C Memb. Submiss.* **2004**, *21*, 1–31.
51. Sirin, E.; Parsia, B.; Grau, B.C.; Kalyanpur, A.; Katz, Y. Pellet: A practical OWL-DL reasoner. *J. Web Semant.* **2007**, *5*, 51–53. [CrossRef]
52. Urbani, J.; Margara, A.; Jacobs, C.J.H.; van Harmelen, F.; Bal, H.E. DynamiTE: Parallel Materialization of Dynamic RDF Data. In Proceedings of the 12th International Semantic Web Conference (ISWC) 2013, Sydney, Australia, 21–25 October 2013; Volume 8218, pp. 657–672. [CrossRef]
53. Mohamed, H.; Fathalla, S.; Lehmann, J.; Jabeen, H. A Scalable Approach for Distributed Reasoning over Large-scale OWL Datasets. In Proceedings of the 13th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, IC3K 2021, Volume 2: KEOD, Virtual, 25–27 October 2021; pp. 51–60. [CrossRef]
54. Benítez-Hidalgo, A.; Navas-Delgado, I.; del Mar Roldán García, M. NORA: Scalable OWL reasoner based on NoSQL databases and Apache Spark. *Softw. Pract. Exp.* **2023**, *53*, 2377–2392. [CrossRef]
55. Hu, P.; Urbani, J.; Motik, B.; Horrocks, I. Datalog Reasoning over Compressed RDF Knowledge Bases. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, 3–7 November 2019; pp. 2065–2068. [CrossRef]
56. Knublauch, H.; Kontokostas, D. Shapes constraint language (SHACL). *W3C Candidate Recomm.* **2017**, *11*. Available online: https://www.w3.org/TR/shacl/ (accessed on 18 August 2024).
57. Prud'hommeaux, E.; Gayo, J.E.L.; Solbrig, H.R. Shape expressions: An RDF validation and transformation language. In Proceedings of the Joint Conference on Lexical and Computational Semantics, Dublin, Ireland, 23–24 August 2014.
58. Frey, J.; Müller, K.; Hellmann, S.; Rahm, E.; Vidal, M.E. Evaluation of metadata representations in RDF stores. *Semant. Web* **2019**, *10*, 205–229. [CrossRef]
59. Sikos, L.F.; Philp, D. Provenance-aware knowledge representation: A survey of data models and contextualized Knowledge Graphs. *Data Sci. Eng.* **2020**, *5*, 293–316. [CrossRef]
60. Zhang, F.; Li, Z.; Peng, D.; Cheng, J. RDF for temporal data management—A survey. *Earth Sci. Inform.* **2021**, *14*, 563–599. [CrossRef]
61. Lehmann, J.; Sejdiu, G.; Bühmann, L.; Westphal, P.; Stadler, C.; Ermilov, I.; Bin, S.; Chakraborty, N.; Saleem, M.; Ngomo, A.C.N.; et al. Distributed Semantic Analytics Using the SANSA Stack. In Proceedings of the International Workshop on the Semantic Web (ISWC) 2017, Vienna, Austria, 21–25 October 2017.
62. Angles, R. The Property Graph Database Model. In Proceedings of the AMW, 2018, Cali, Colombia, 21–25 May 2018.
63. Lbath, H.; Bonifati, A.; Harmer, R. Schema inference for property graphs. In Proceedings of the EDBT 2021-24th International Conference on Extending Database Technology, Nicosia, Cyprus, 23–26 March 2021; pp. 499–504.
64. Neo4j Inc. Neo4j Graph Database. Available online: https://neo4j.com/ (accessed on 18 August 2024).
65. The Linux Foundation. JanusGraph: An Open Source, Distributed Graph Database. Available online: https://janusgraph.org (accessed on 18 August 2024).
66. TigerGraph, Inc. TigerGraph Graph Database. Available online: https://www.tigergraph.com (accessed on 18 August 2024).
67. Hong, S.; Depner, S.; Manhardt, T.; Van Der Lugt, J.; Verstraaten, M.; Chafi, H. PGX.D: A Fast Distributed Graph Processing Engine. In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC '15, New York, NY, USA, 12–17 November 2015. [CrossRef]
68. Rost, C.; Gómez, K.; Täschner, M.; Fritzsche, P.; Schons, L.; Christ, L.; Adameit, T.; Junghanns, M.; Rahm, E. Distributed temporal graph analytics with GRADOOP. *VLDB J.* **2022**, *31*, 375–401. [CrossRef]
69. Wood, P.T. Query languages for graph databases. *SIGMOD Rec.* **2012**, *41*, 50–60. [CrossRef]

70. Angles, R.; Arenas, M.; Barceló, P.; Boncz, P.A.; Fletcher, G.H.L.; Gutierrez, C.; Lindaaker, T.; Paradies, M.; Plantikow, S.; Sequeda, J.F.; et al. G-CORE: A Core for Future Graph Query Languages. In Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, 10–15 June 2018; pp. 1421–1432. [CrossRef]
71. Rodriguez, M.A. The Gremlin graph traversal machine and language (invited talk). In Proceedings of the 15th Symposium on Database Programming Languages (SPLASH), Pittsburgh, PA, USA, 25–30 October 2015. [CrossRef]
72. van Rest, O.; Hong, S.; Kim, J.; Meng, X.; Chafi, H. PGQL: A property graph query language. In Proceedings of the Fourth International Workshop on Graph Data Management Experiences and Systems, Redwood Shores, CA, USA, 24 June 2016; p. 7. [CrossRef]
73. Francis, N.; Green, A.; Guagliardo, P.; Libkin, L.; Lindaaker, T.; Marsault, V.; Plantikow, S.; Rydberg, M.; Selmer, P.; Taylor, A. Cypher: An Evolving Query Language for Property Graphs. In Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, 10–15 June 2018; pp. 1433–1445. [CrossRef]
74. Deutsch, A.; Francis, N.; Green, A.; Hare, K.; Li, B.; Libkin, L.; Lindaaker, T.; Marsault, V.; Martens, W.; Michels, J.; et al. Graph Pattern Matching in GQL and SQL/PGQ. In Proceedings of the SIGMOD'22: International Conference on Management of Data, Philadelphia, PA, USA, 12–17 June 2022; pp. 2246–2258. [CrossRef]
75. Chiba, H.; Yamanaka, R.; Matsumoto, S. Property Graph Exchange Format. *arXiv* **2019**, arXiv:1907.03936. [CrossRef]
76. Tomaszuk, D.; Angles, R.; Szeremeta, L.; Litman, K.; Cisterna, D. Serialization for Property Graphs. In Proceedings of the Beyond Databases, Architectures and Structures. Paving the Road to Smart Data Processing and Analysis—15th International Conference, BDAS 2019, Ustroń, Poland, 28–31 May 2019; pp. 57–69. [CrossRef]
77. Neelam, S.; Sharma, U.; Bhatia, S.; Karanam, H.; Likhyani, A.; Abdelaziz, I.; Fokoue, A.; Subramaniam, L.V. Expressive Reasoning Graph Store: A Unified Framework for Managing RDF and Property Graph Databases. *arXiv* **2022**, arXiv:2209.05828. [CrossRef]
78. Angles, R.; Bonifati, A.; Dumbrava, S.; Fletcher, G.; Hare, K.; Hidders, J.; Lee, V.E.; Li, B.; Libkin, L.; Martens, W.; et al. PG-Keys: Keys for Property Graphs. In Proceedings of the 2021 International Conference on Management of Data, Shanxi, China, 3–5 June 2021.
79. Bonifati, A.; Dumbrava, S.; Fletcher, G.; Hidders, J.; Li, B.; Libkin, L.; Martens, W.; Murlak, F.; Plantikow, S.; Savkovi'c, O.; et al. PG-Schema: Schemas for Property Graphs. *Proc. ACM Manag. Data* **2022**, *1*, 1–25.
80. Rost, C.; Fritzsche, P.; Schons, L.; Zimmer, M.; Gawlick, D.; Rahm, E. Bitemporal Property Graphs to Organize Evolving Systems. *arXiv* **2021**, arXiv:2111.13499. [CrossRef]
81. Besta, M.; Fischer, M.; Kalavri, V.; Kapralov, M.; Hoefler, T. Practice of Streaming and Dynamic Graphs: Concepts, Models, Systems, and Parallelism. *arXiv* **2019**, arXiv:1912.12740. [CrossRef]
82. Lassila, O.; Schmidt, M.; Hartig, O.; Bebee, B.; Bechberger, D.; Broekema, W. The OneGraph Vision: Challenges of Breaking the Graph Model Lock-In. *Semant. Web* **2022**. [CrossRef]
83. Tian, Y. The World of Graph Databases from An Industry Perspective. *SIGMOD Rec.* **2022**, *51*, 60–67. [CrossRef]
84. Ilyas, I.F.; Rekatsinas, T.; Konda, V.; Pound, J.; Qi, X.; Soliman, M. Saga: A Platform for Continuous Construction and Serving of Knowledge at Scale. In Proceedings of the 2022 International Conference on Management of Data, SIGMOD '22, New York, NY, USA, 12–17 June 2022; pp. 2259–2272. [CrossRef]
85. Hartig, O. Reconciliation of RDF* and Property Graphs. *arXiv* **2014**, arXiv:1409.3288. [CrossRef]
86. Abuoda, G.; Dell'Aglio, D.; Keen, A.; Hose, K. Transforming RDF-star to Property Graphs: A Preliminary Analysis of Transformation Approaches. In Proceedings of the QuWeDa 2022: 6th Workshop on Storing, Querying and Benchmarking Knowledge Graphs at ISWC, Online, 23 October 2022; Volume 3279, pp. 17–32.
87. Taelman, R.; Sande, M.V.; Verborgh, R. GraphQL-LD: Linked Data Querying with GraphQL. In Proceedings of the ISWC 2018 Posters & Demonstrations, Industry and Blue Sky Ideas Tracks Co-Located with 17th International Semantic Web Conference (ISWC 2018), Monterey, CA, USA, 8–12 October 2018; Volume 2180.
88. Cudré-Mauroux, P. Leveraging Knowledge Graphs for big data integration: The XI pipeline. *Semant. Web* **2020**, *11*, 13–17. [CrossRef]
89. Madnick, S.E.; Wang, R.Y.; Lee, Y.W.; Zhu, H. Overview and Framework for Data and Information Quality Research. *ACM J. Data Inf. Qual.* **2009**, *1*, 1–22. [CrossRef]
90. Zaveri, A.; Rula, A.; Maurino, A.; Pietrobon, R.; Lehmann, J.; Auer, S. Quality assessment for linked data: A survey. *Semant. Web* **2016**, *7*, 63–93. [CrossRef]
91. Wang, X.; Chen, L.; Ban, T.; Usman, M.; Guan, Y.; Liu, S.; Wu, T.; Chen, H. Knowledge Graph Quality Control: A Survey. *Fundam. Res.* **2021**, *1*, 607–626. [CrossRef]
92. Narayan, A.; Chami, I.; Orr, L.J.; Ré, C. Can Foundation Models Wrangle Your Data? *Proc. VLDB Endow.* **2022**, *16*, 738–746. [CrossRef]
93. Trummer, I. From BERT to GPT-3 Codex: Harnessing the Potential of Very Large Language Models for Data Management. *Proc. VLDB Endow.* **2022**, *15*, 3770–3773. [CrossRef]
94. Wilkinson, M.D.; Dumontier, M.; Aalbersberg, I.J.; Appleton, G.; Axton, M.; Baak, A.; Blomberg, N.; Boiten, J.W.; da Silva Santos, L.B.; Bourne, P.E.; et al. The FAIR Guiding Principles for scientific data management and stewardship. *Sci. Data* **2016**, *3*, 1–9. [CrossRef] [PubMed]
95. Kricke, M.; Grimmer, M.; Schmeißer, M. Preserving Recomputability of Results from Big Data Transformation Workflows. *Datenbank-Spektrum* **2017**, *17*, 245–253. [CrossRef]

96. Greenberg, J. Understanding metadata and metadata schemes. *Cat. Classif. Q.* **2005**, *40*, 17–36. [CrossRef]
97. Neto, C.B.; Kontokostas, D.; Kirschenbaum, A.; Publio, G.C.; Esteves, D.; Hellmann, S. IDOL: Comprehensive & complete LOD insights. In Proceedings of the 13th International Conference on Semantic Systems (SEMANTiCS), Amsterdam, The Netherlands, 11–14 September 2017; pp. 49–56.
98. Duval, E.; Hodgins, W.; Sutton, S.; Weibel, S.L. Metadata principles and practicalities. *D-Lib Mag.* **2002**, *8*, 1–10. [CrossRef]
99. Arora, S.; Yang, B.; Eyuboglu, S.; Narayan, A.; Hojel, A.; Trummer, I.; Ré, C. Language Models Enable Simple Systems for Generating Structured Views of Heterogeneous Data Lakes. *Proc. VLDB Endow.* **2023**, *17*, 92–105. [CrossRef]
100. Chen, Z.; Cao, L.; Madden, S.; Kraska, T.; Shang, Z.; Fan, J.; Tang, N.; Gu, Z.; Liu, C.; Cafarella, M. SEED: Domain-Specific Data Curation With Large Language Models. *arXiv* **2023**, arXiv:2310.00749. [CrossRef]
101. Kadioglu, D.; Breil, B.; Knell, C.; Lablans, M.; Mate, S.; Schlue, D.; Serve, H.; Storf, H.; Ückert, F.; Wagner, T.O.; et al. Samply. MDR-A Metadata Repository and Its Application in Various Research Networks. In Proceedings of the GMDS, Osnabrück, Germany, 2–6 September 2018; pp. 50–54.
102. Frey, J.; Götz, F.; Hofer, M.; Hellmann, S. Managing and Compiling Data Dependencies for Semantic Applications Using Databus Client. In Proceedings of the Research Conference on Metadata and Semantics Research, London, UK, 29 November–3 December 2022; pp. 114–125.
103. Frey, J.; Hofer, M.; Obraczka, D.; Lehmann, J.; Hellmann, S. DBpedia FlexiFusion the best of Wikipedia> Wikidata> your data. In Proceedings of the 18th International Semantic Web Conference, Auckland, New Zealand, 26–30 October 2019; pp. 96–112.
104. Meester, B.D.; Dimou, A.; Verborgh, R.; Mannens, E. Detailed Provenance Capture of Data Processing. In Proceedings of the SemSci@ISWC, Vienna, Austria, 21 October 2017.
105. Meester, B.D.; Seymoens, T.; Dimou, A.; Verborgh, R. Implementation-independent function reuse. *Future Gener. Comput. Syst.* **2020**, *110*, 946–959. [CrossRef]
106. Fernández, J.D.; Polleres, A.; Umbrich, J. Towards Efficient Archiving of Dynamic Linked Open Data. In Proceedings of the First DIACHRON Workshop on Managing the Evolution and Preservation of the Data Web Co-Located with 12th European Semantic Web Conference (ESWC 2015), Portorož, Slovenia, 31 May 2015; Volume 1377, pp. 34–49.
107. Taelman, R.; Mahieu, T.; Vanbrabant, M.; Verborgh, R. Optimizing storage of RDF archives using bidirectional delta chains. *Semant. Web* **2022**, *13*, 705–734. [CrossRef]
108. Hofer, M.; Hellmann, S.; Dojchinovski, M.; Frey, J. The new dbpedia release cycle: Increasing agility and efficiency in Knowledge Extraction workflows. In Proceedings of the International Conference on Semantic Systems, Amsterdam, The Netherlands, 7–10 September 2020; pp. 1–18.
109. Zhang, H.; Wang, X.; Pan, J.; Wang, H. SAKA: An intelligent platform for semi-automated Knowledge Graph construction and application. *Serv. Oriented Comput. Appl.* **2023**, *17*, 201–212. [CrossRef]
110. Graube, M.; Hensel, S.; Urbas, L. R43ples: Revisions for Triples—An Approach for Version Control in the Semantic Web. In Proceedings of the 1st Workshop on Linked Data Quality Co-Located with 10th International Conference on Semantic Systems, LDQ@SEMANTiCS 2014, Leipzig, Germany, 2 September 2014; Volume 1215.
111. Arndt, N.; Naumann, P.; Radtke, N.; Martin, M.; Marx, E. Decentralized Collaborative Knowledge Management Using Git. *J. Web Semant.* **2019**, *54*, 29–47. [CrossRef]
112. Anderson, J.; Bendiken, A. Transaction-Time Queries in Dydra. In Proceedings of the Joint Proceedings of the 2nd Workshop on Managing the Evolution and Preservation of the Data Web (MEPDaW 2016) and the 3rd Workshop on Linked Data Quality (LDQ 2016) Co-Located with 13th European Semantic Web Conference (ESWC 2016), Heraklion, Greece, 30 May 2016; Volume 1585, pp. 11–19.
113. Debrouvier, A.; Parodi, E.; Perazzo, M.; Soliani, V.; Vaisman, A.A. A model and query language for temporal graph databases. *VLDB J.* **2021**, *30*, 825–858. [CrossRef]
114. Dong, X.L.; Gabrilovich, E.; Murphy, K.; Dang, V.; Horn, W.; Lugaresi, C.; Sun, S.; Zhang, W. Knowledge-Based Trust: Estimating the Trustworthiness of Web Sources. *Proc. VLDB Endow.* **2015**, *8*, 938–949. [CrossRef]
115. Amsterdamer, Y.; Cohen, M. Automated Selection of Multiple Datasets for Extension by Integration. In Proceedings of the 30th ACM International Conference on Information & Knowledge Management, Queensland, Australia, 1–5 November 2021; pp. 27–36.
116. Fetahu, B.; Dietze, S.; Pereira Nunes, B.; Antonio Casanova, M.; Taibi, D.; Nejdl, W. A scalable approach for efficiently generating structured dataset topic profiles. In Proceedings of the European Semantic Web Conference (ESWC), Crete, Greece, 25–29 May 2014; pp. 519–534.
117. Blei, D.M.; Lafferty, J.D. A correlated topic model of science. *Ann. Appl. Stat.* **2007**, *1*, 17–35. [CrossRef]
118. Nentwig, M.; Rahm, E. Incremental clustering on linked data. In Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW), Singapore, 17–20 November 2018; pp. 531–538.
119. Saeedi, A.; Peukert, E.; Rahm, E. Incremental Multi-source Entity Resolution for Knowledge Graph Completion. In Proceedings of the European Semantic Web Conference (ESWC), Athens, Greece, 2–6 November 2020; pp. 393–408.
120. Hertling, S.; Paulheim, H. Order Matters: Matching Multiple Knowledge Graphs. In Proceedings of the K-CAP '21: Knowledge Capture Conference, Virtual, 2–3 December 2021; pp. 113–120. [CrossRef]
121. Giese, M.; Soylu, A.; Vega-Gorgojo, G.; Waaler, A.; Haase, P.; Jiménez-Ruiz, E.; Lanti, D.; Rezk, M.; Xiao, G.; Özçep, Ö.L.; et al. Optique: Zooming in on Big Data. *Computer* **2015**, *48*, 60–67. [CrossRef]

122. Civili, C.; Console, M.; Giacomo, G.D.; Lembo, D.; Lenzerini, M.; Lepore, L.; Mancini, R.; Poggi, A.; Rosati, R.; Ruzzi, M.; et al. MASTRO STUDIO: Managing Ontology-Based Data Access applications. *Proc. VLDB Endow.* **2013**, *6*, 1314–1317. [CrossRef]
123. Mami, M.N.; Graux, D.; Scerri, S.; Jabeen, H.; Auer, S.; Lehmann, J. Squerall: Virtual Ontology-Based Access to Heterogeneous and Large Data Sources. In Proceedings of the 18th International Semantic Web Conference (ISWC), Auckland, New Zealand, 26–30 October 2019; Volume 11779, pp. 229–245. [CrossRef]
124. Banavar, G.; Chandra, T.; Mukherjee, B.; Nagarajarao, J.; Strom, R.E.; Sturman, D.C. An efficient multicast protocol for content-based publish-subscribe systems. In Proceedings of the 19th IEEE International Conference on Distributed Computing Systems (Cat. No. 99CB37003), Austin, TX, USA, 5 June 1999; pp. 262–272.
125. Völkel, M.; Groza, T. SemVersion: An RDF-based Ontology Versioning System. In Proceedings of the IADIS International Conference on WWW/Internet, IADIS, Murcia, Spain, 5–8 October 2006.
126. Im, D.H.; Lee, S.W.; Kim, H.J. A Version Management Framework for RDF Triple Stores. *Int. J. Softw. Eng. Knowl. Eng.* **2012**, *22*, 85–106. [CrossRef]
127. Sande, M.V.; Colpaert, P.; Verborgh, R.; Coppens, S.; Mannens, E.; de Walle, R.V. R&Wbase: Git for triples. In Proceedings of the LDOW, Rio de Janeiro, Brazil, 14 May 2013.
128. Neumann, T.; Weikum, G. X-RDF-3X: Fast Querying, High Update Rates, and Consistency for RDF Databases. *Proc. VLDB Endow.* **2010**, *3*, 256–263. [CrossRef]
129. Stefanidis, K.; Chrysakis, I.; Flouris, G. On Designing Archiving Policies for Evolving RDF Datasets on the Web. In Proceedings of the Conceptual Modeling: 33rd International Conference, ER 2014, Atlanta, GA, USA, 27–29 October 2014; Volume 8824, pp. 43–56.
130. Taelman, R.; Colpaert, P.; Mannens, E.; Verborgh, R. Generating public transport data based on population distributions for RDF benchmarking. *Semant. Web* **2019**, *10*, 305–328. [CrossRef]
131. Lancker, D.V.; Colpaert, P.; Delva, H.; de Vyvere, B.V.; Meléndez, J.A.R.; Dedecker, R.; Michiels, P.; Buyle, R.; Craene, A.D.; Verborgh, R. Publishing Base Registries as Linked Data Event Streams. In Proceedings of the Web Engineering—21st International Conference, ICWE 2021, Biarritz, France, 18–21 May 2021; pp. 28–36. [CrossRef]
132. Assche, D.V.; Oo, S.M.; Rojas, J.A.; Colpaert, P. Continuous generation of versioned collections' members with RML and LDES. In Proceedings of the 3rd International Workshop on Knowledge Graph Construction (KGCW 2022) Co-Located with 19th Extended Semantic Web Conference (ESWC 2022), Hersonissos, Greek, 30 May 2022; Volume 3141.
133. Aebeloe, C.; Keles, I.; Montoya, G.; Hose, K. Star Pattern Fragments: Accessing Knowledge Graphs through Star Patterns. *arXiv* **2020**, arXiv:2002.09172. [CrossRef]
134. Polleres, A.; Kamdar, M.R.; Fernández, J.D.; Tudorache, T.; Musen, M.A. A More Decentralized Vision for Linked Data. In Proceedings of the 2nd Workshop on Decentralizing the Semantic Web Co-Located with the 17th International Semantic Web Conference, DeSemWeb@ISWC 2018, Monterey, CA, USA, 8 October 2018; Volume 2165.
135. Verborgh, R.; Sande, M.V.; Hartig, O.; Herwegen, J.V.; Vocht, L.D.; Meester, B.D.; Haesendonck, G.; Colpaert, P. Triple Pattern Fragments: A low-cost Knowledge Graph interface for the Web. *J. Web Semant.* **2016**, *37–38*, 184–206. [CrossRef]
136. Aebeloe, C.; Montoya, G.; Hose, K. A Decentralized Architecture for Sharing and Querying Semantic Data. In Proceedings of the Semantic Web—16th International Conference, ESWC 2019, Portorož, Slovenia, 2–6 June 2019; Volume 11503, pp. 3–18. [CrossRef]
137. Aebeloe, C.; Montoya, G.; Hose, K. Decentralized Indexing over a Network of RDF Peers. In Proceedings of the Semantic Web—ISWC 2019—18th International Semantic Web Conference, Auckland, New Zealand, 26–30 October 2019; Volume 11778, pp. 3–20. [CrossRef]
138. Cai, M.; Frank, M.R. RDFPeers: A scalable distributed RDF repository based on a structured peer-to-peer network. In Proceedings of the 13th International Conference on World Wide Web, WWW 2004, New York, NY, USA, 17–20 May 2004; pp. 650–657. [CrossRef]
139. Azzam, A.; Fernández, J.D.; Acosta, M.; Beno, M.; Polleres, A. SMART-KG: Hybrid Shipping for SPARQL Querying on the Web. In Proceedings of the WWW '20: The Web Conference 2020, Taipei, Taiwan, 20–24 April 2020; pp. 984–994. [CrossRef]
140. Hartig, O.; Aranda, C.B. Bindings-Restricted Triple Pattern Fragments. In Proceedings of the on the Move to Meaningful Internet Systems: OTM 2016 Conferences—Confederated International Conferences: CoopIS, C&TC, and ODBASE 2016, Rhodes, Greece, 24–28 October 2016; Volume 10033, pp. 762–779. [CrossRef]
141. Minier, T.; Skaf-Molli, H.; Molli, P. SaGe: Préemption Web pour les services publics d'évaluation de requêtes SPARQL. In Proceedings of the IC 2019: 30es Journées Francophones d'Ingénierie des Connaissances (Proceedings of the 30th French Knowledge Engineering Conference), Toulouse, France, 2–4 July 2019; p. 141.
142. Montoya, G.; Aebeloe, C.; Hose, K. Towards Efficient Query Processing over Heterogeneous RDF Interfaces. In Proceedings of the Emerging Topics in Semantic Technologies—ISWC 2018 Satellite Events [Best Papers from 13 of the Workshops Co-Located with the ISWC 2018 Conference], Monterey, CA, USA, October 2018; Demidova, E., Zaveri, A., Simperl, E., Eds.; IOS Press: Amsterdam, The Netherlands, 2018; Volume 36, pp. 39–53. [CrossRef]
143. Azzam, A.; Aebeloe, C.; Montoya, G.; Keles, I.; Polleres, A.; Hose, K. WiseKG: Balanced Access to Web Knowledge Graphs. In Proceedings of the WWW '21: The Web Conference 2021, Ljubljana, Slovenia, 19–23 April 2021; Leskovec, J., Grobelnik, M., Najork, M., Tang, J., Zia, L., Eds.; ACM/IW3C2, 2021; pp. 1422–1434. [CrossRef]

144. Junior, A.C.; Debruyne, C.; Brennan, R.; O'Sullivan, D. FunUL: A method to incorporate functions into uplift mapping languages. In Proceedings of the 18th International Conference on Information Integration and Web-Based Applications and Services, Singapore, 28–30 November 2016.
145. Dimou, A. R2RML and RML Comparison for RDF Generation, their Rules Validation and Inconsistency Resolution. *arXiv* **2020**, arXiv:2005.06293. [CrossRef]
146. Dimou, A.; Vander Sande, M.; Colpaert, P.; Verborgh, R.; Mannens, E.; Van de Walle, R. RDF mapping language (RML). *Specif. Propos. Draft.* **2014**. Available online: https://rml.io/specs/rml/ (accessed on 18 August 2024).
147. Iglesias, E.; Jozashoori, S.; Chaves-Fraga, D.; Collarana, D.; Vidal, M.E. SDM-RDFizer: An RML Interpreter for the Efficient Creation of RDF Knowledge Graphs. In Proceedings of the 29th ACM International Conference on Information & Knowledge Management, Virtual Event, 19–23 October 2020.
148. Knoblock, C.A.; Szekely, P.A.; Ambite, J.L.; Goel, A.; Gupta, S.; Lerman, K.; Muslea, M.; Taheriyan, M.; Mallick, P. Semi-automatically Mapping Structured Sources into the Semantic Web. In Proceedings of the Semantic Web: Research and Applications—9th Extended Semantic Web Conference, ESWC 2012, Heraklion, Greece, 27–31 May 2012; Simperl, E., Cimiano, P., Polleres, A., Corcho, Ó., Presutti, V., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2012; Volume 7295, pp. 375–390. [CrossRef]
149. Jain, N.; Liao, G.; Willke, T.L. Graphbuilder: Scalable graph ETL framework. In Proceedings of the First International Workshop on Graph Data Management Experiences and Systems (GRADES), New York, NY, USA, 23 June 2013.
150. Kricke, M.; Peukert, E.; Rahm, E. Graph data transformations in Gradoop. In Proceedings of the Conference on Database Systems for Business, Technology and Web (BTW), Rostock, Germany, 4–8 March 2019. [CrossRef]
151. Angles, R.; Thakkar, H.; Tomaszuk, D. Mapping RDF databases to property graph databases. *IEEE Access* **2020**, *8*, 86091–86110. [CrossRef]
152. Lefrançois, M.; Zimmermann, A.; Bakerally, N. A SPARQL Extension for Generating RDF from Heterogeneous Formats. In Proceedings of the Extended Semantic Web Conference (ESWC), Portoroz, Slovenia, 28 May–1 June 2017.
153. de Medeiros, L.F.; Priyatna, F.; Corcho, Ó. MIRROR: Automatic R2RML Mapping Generation from Relational Databases. In Proceedings of the International Conference on Web Engineering (ICWE), Rotterdam, The Netherlands, 23–26 June 2015.
154. Sicilia, Á.; Nemirovski, G. AutoMap4OBDA: Automated Generation of R2RML Mappings for OBDA. In Proceedings of the International Conference Knowledge Engineering and Knowledge Management (EKAW), Bologna, Italy, 19–23 November 2016.
155. Jiménez-Ruiz, E.; Kharlamov, E.; Zheleznyakov, D.; Horrocks, I.; Pinkel, C.; Skjæveland, M.G.; Thorstensen, E.; Mora, J. BootOX: Practical Mapping of RDBs to OWL 2. In Proceedings of the International Workshop on the Semantic Web (ISWC), Bethlehem, PA, USA, 11–15 October 2015.
156. Rahm, E.; Do, H.H. Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.* **2000**, *23*, 3–13.
157. Abedjan, Z.; Chu, X.; Deng, D.; Fernandez, R.C.; Ilyas, I.F.; Ouzzani, M.; Papotti, P.; Stonebraker, M.; Tang, N. Detecting Data Errors: Where are we and what needs to be done? *Proc. VLDB Endow.* **2016**, *9*, 993–1004. [CrossRef]
158. Ilyas, I.F.; Chu, X. *Data Cleaning*; Morgan & Claypool: San Rafael, CA, USA, 2019; ISBN 978-1-4503-7152-0.
159. Fiorelli, M.; Stellato, A. Lifting Tabular Data to RDF: A Survey. In Proceedings of the Metadata and Semantic Research (MTSR), Virtual, 2–4 December 2020; Garoufallou, E., Ovalle-Perandones, M.A., Eds.; Springer: Cham, Switzerland, 2020; pp. 85–96. [CrossRef]
160. Abedjan, Z.; Golab, L.; Naumann, F.; Papenbrock, T. Data profiling. *Synth. Lect. Data Manag.* **2018**, *10*, 1–154.
161. Beskales, G.; Ilyas, I.F.; Golab, L. Sampling the Repairs of Functional Dependency Violations under Hard Constraints. *Proc. VLDB Endow.* **2010**, *3*, 197–207. [CrossRef]
162. Beskales, G.; Ilyas, I.F.; Golab, L.; Galiullin, A. On the relative trust between inconsistent data and inaccurate constraints. In Proceedings of the 29th IEEE International Conference on Data Engineering, ICDE 2013, Brisbane, Australia, 8–12 April 2013; Jensen, C.S., Jermaine, C.M., Zhou, X., Eds.; IEEE Computer Society: Washington, DC, USA, 2013; pp. 541–552. [CrossRef]
163. Khayyat, Z.; Ilyas, I.F.; Jindal, A.; Madden, S.; Ouzzani, M.; Papotti, P.; Quiané-Ruiz, J.; Tang, N.; Yin, S. BigDansing: A System for Big Data Cleansing. In Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Australia, 31 May–4 June 2015; Sellis, T.K., Davidson, S.B., Ives, Z.G., Eds.; ACM: New York, NY, USA, 2015; pp. 1215–1230. [CrossRef]
164. Kolahi, S.; Lakshmanan, L.V.S. On approximating optimum repairs for functional dependency violations. In Proceedings of the Database Theory—ICDT 2009, 12th International Conference, St. Petersburg, Russia, 23–25 March 2009; Fagin, R., Ed.; ACM: New York, NY, USA, 2009; Volume 361, pp. 53–62. [CrossRef]
165. Bohannon, P.; Fan, W.; Geerts, F.; Jia, X.; Kementsietsidis, A. Conditional Functional Dependencies for Data Cleaning. In Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, The Marmara Hotel, Istanbul, Turkey, 15–20 April 2007; pp. 746–755. [CrossRef]
166. Fan, W.; Geerts, F.; Jia, X.; Kementsietsidis, A. Conditional functional dependencies for capturing data inconsistencies. *ACM Trans. Database Syst.* **2008**, *33*, 48. [CrossRef]
167. Geerts, F.; Mecca, G.; Papotti, P.; Santoro, D. The LLUNATIC Data-Cleaning Framework. *Proc. VLDB Endow.* **2013**, *6*, 625–636. [CrossRef]
168. Chu, X.; Ilyas, I.F.; Papotti, P. Holistic data cleaning: Putting violations into context. In Proceedings of the 29th IEEE International Conference on Data Engineering, ICDE 2013, Brisbane, Australia, 8–12 April 2013; pp. 458–469. [CrossRef]

169. Heidari, A.; McGrath, J.; Ilyas, I.F.; Rekatsinas, T. HoloDetect: Few-Shot Learning for Error Detection. In Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, 30 June–5 July 2019; pp. 829–846. [CrossRef]
170. Lopatenko, A.; Bravo, L. Efficient Approximation Algorithms for Repairing Inconsistent Databases. In Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, The Marmara Hotel, Istanbul, Turkey, 15–20 April 2007; pp. 216–225. [CrossRef]
171. Rekatsinas, T.; Chu, X.; Ilyas, I.F.; Ré, C. HoloClean: Holistic Data Repairs with Probabilistic Inference. *Proc. VLDB Endow.* **2017**, *10*, 1190–1201. [CrossRef]
172. Krishnan, S.; Wang, J.; Wu, E.; Franklin, M.J.; Goldberg, K. ActiveClean: Interactive Data Cleaning For Statistical Modeling. *Proc. VLDB Endow.* **2016**, *9*, 948–959. [CrossRef]
173. Mahdavi, M.; Abedjan, Z.; Fernandez, R.C.; Madden, S.; Ouzzani, M.; Stonebraker, M.; Tang, N. Raha: A Configuration-Free Error Detection System. In Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, 30 June–5 July 2019; pp. 865–882. [CrossRef]
174. Milani, M.; Zheng, Z.; Chiang, F. CurrentClean: Spatio-Temporal Cleaning of Stale Data. In Proceedings of the 35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, 8–11 April 2019; pp. 172–183. [CrossRef]
175. Assadi, A.; Milo, T.; Novgorodov, S. DANCE: Data Cleaning with Constraints and Experts. In Proceedings of the 33rd IEEE International Conference on Data Engineering, ICDE 2017, San Diego, CA, USA, 19–22 April 2017; pp. 1409–1410. [CrossRef]
176. Chu, X.; Ouzzani, M.; Morcos, J.; Ilyas, I.F.; Papotti, P.; Tang, N.; Ye, Y. KATARA: Reliable Data Cleaning with Knowledge Bases and Crowdsourcing. *Proc. VLDB Endow.* **2015**, *8*, 1952–1955. [CrossRef]
177. He, J.; Veltri, E.; Santoro, D.; Li, G.; Mecca, G.; Papotti, P.; Tang, N. Interactive and Deterministic Data Cleaning. In Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, 26 June–1 July 2016; pp. 893–907. [CrossRef]
178. Thirumuruganathan, S.; Berti-Équille, L.; Ouzzani, M.; Quiané-Ruiz, J.; Tang, N. UGuide: User-Guided Discovery of FD-Detectable Errors. In Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference 2017, Chicago, IL, USA, 14–19 May 2017; pp. 1385–1397. [CrossRef]
179. Tong, Y.; Cao, C.C.; Zhang, C.J.; Li, Y.; Chen, L. CrowdCleaner: Data cleaning for multi-version data on the web via crowdsourcing. In Proceedings of the IEEE 30th International Conference on Data Engineering, ICDE 2014, Chicago, IL, USA, 31 March–4 April 2014; pp. 1182–1185. [CrossRef]
180. Yakout, M.; Elmagarmid, A.K.; Neville, J.; Ouzzani, M.; Ilyas, I.F. Guided data repair. *Proc. VLDB Endow.* **2011**, *4*, 279–289. [CrossRef]
181. Wang, R.; Li, Y.; Wang, J. Sudowoodo: Contrastive Self-supervised Learning for Multi-purpose Data Integration and Preparation. In Proceedings of the 2023 IEEE 39th International Conference on Data Engineering (ICDE), Los Alamitos, CA, USA, 3–7 April 2023; pp. 1502–1515. [CrossRef]
182. Neutatz, F.; Chen, B.; Abedjan, Z.; Wu, E. From Cleaning before ML to Cleaning for ML. *IEEE Data Eng. Bull.* **2021**, *44*, 24–41.
183. Hao, S.; Tang, N.; Li, G.; Li, J.; Feng, J. Distilling relations using knowledge bases. *VLDB J.* **2018**, *27*, 497–519. [CrossRef]
184. Ge, C.; Gao, Y.; Weng, H.; Zhang, C.; Miao, X.; Zheng, B. KGClean: An Embedding Powered Knowledge Graph Cleaning Framework. *arXiv* **2020**, arXiv:2004.14478. [CrossRef]
185. Noy, N.F.; McGuinness, D.L. *Ontology Development 101: A Guide to Creating Your First Ontology*; Stanford Knowledge Systems Laboratory Technical Report KSL-01-05; Stanford Knowledge Systems Laboratory: Stanford, CA, USA, 2001.
186. Al-Aswadi, F.N.; Chan, H.Y.; Gan, K.H. Automatic ontology construction from text: A review from shallow to deep learning trend. *Artif. Intell. Rev.* **2020**, *53*, 3901–3928. [CrossRef]
187. Browarnik, A.; Maimon, O. Ontology learning from text: Why the ontology learning layer cake is not viable. *Int. J. Signs Semiot. Syst. (IJSSS)* **2015**, *4*, 1–14. [CrossRef]
188. Wong, W.; Liu, W.; Bennamoun, M. Ontology learning from text: A look back and into the future. *ACM Comput. Surv. (CSUR)* **2012**, *44*, 1–36. [CrossRef]
189. Giglou, H.B.; D'Souza, J.; Auer, S. LLMs4OL: Large Language Models for Ontology Learning. In Proceedings of the Semantic Web-ISWC 2023—22nd International Semantic Web Conference, Athens, Greece, 6–10 November 2023; Volume 14265, pp. 408–427. [CrossRef]
190. Funk, M.; Hosemann, S.; Jung, J.C.; Lutz, C. Towards Ontology Construction with Language Models. In Proceedings of the Joint proceedings of the 1st workshop on Knowledge Base Construction from Pre-Trained Language Models (KBC-LM) and the 2nd challenge on Language Models for Knowledge Base Construction (LM-KBC) Co-Located with the 22nd International Semantic Web Conference (ISWC 2023), Athens, Greece, 6 November 2023; Volume 3577.
191. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models are Few-Shot Learners. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Vancouver, BC, Canada, 6–12 December 2020; Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2020; Volume 33, pp. 1877–1901.
192. Kommineni, V.K.; König-Ries, B.; Samuel, S. From human experts to machines: An LLM supported approach to ontology and Knowledge Graph construction. *arXiv* **2024**, arXiv:2403.08345. [CrossRef]

193. Zhang, B.; Carriero, V.A.; Schreiberhuber, K.; Tsaneva, S.; González, L.S.; Kim, J.; de Berardinis, J. OntoChat: A Framework for Conversational Ontology Engineering using Language Models. *arXiv* **2024**, arXiv:2403.05921. [CrossRef]
194. da Silva, L.M.V.; Köcher, A.; Gehlhoff, F.; Fay, A. On the Use of Large Language Models to Generate Capability Ontologies. *arXiv* **2024**, arXiv:2404.17524. [CrossRef]
195. Ma, C.; Molnár, B. Ontology learning from relational database: Opportunities for semantic information integration. *Vietnam J. Comput. Sci.* **2022**, *9*, 31–57. [CrossRef]
196. De Virgilio, R.; Maccioni, A.; Torlone, R. R2G: A Tool for Migrating Relations to Graphs. In Proceedings of the International Conference on Extending Database Technology (EDBT), Athens, Greece, 24–28 March 2014; pp. 640–643.
197. Petermann, A.; Junghanns, M.; Müller, R.; Rahm, E. BIIIG: Enabling business intelligence with integrated instance graphs. In Proceedings of the 2014 IEEE 30th International Conference on Data Engineering Workshops, Chicago, IL, USA, 31 March–4 April 2014; pp. 4–11.
198. Lehmann, J.; Auer, S.; Bühmann, L.; Tramp, S. Class expression learning for ontology engineering. *J. Web Semant.* **2011**, *9*, 71–81. [CrossRef]
199. Bühmann, L.; Lehmann, J.; Westphal, P. DL-Learner—A framework for inductive learning on the Semantic Web. *J. Web Semant.* **2016**, *39*, 15–24. [CrossRef]
200. Obraczka, D.; Saeedi, A.; Rahm, E. Knowledge Graph Completion with FAMER (DI2KG Challenge Winner). In Proceedings of the 1st International Workshop on Challenges and Experiences from Data Integration to Knowledge Graphs Co-Located with the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD 2019), Anchorage, AK, USA, 5 August 2019.
201. Suchanek, F.M.; Abiteboul, S.; Senellart, P. PARIS: Probabilistic Alignment of Relations, Instances, and Schema. *Proc. VLDB Endow.* **2011**, *5*, 157–168. [CrossRef]
202. Rahm, E.; Bernstein, P.A. A survey of approaches to automatic schema matching. *VLDB J.* **2001**, *10*, 334–350. [CrossRef]
203. Euzenat, J.; Shvaiko, P. *Ontology Matching*; Springer: Cham, Switzerland, 2007; Volume 18, ISBN 978-3-642-38721-0.
204. Bellahsene, Z.; Bonifati, A.; Rahm, E. *Schema Matching and Mapping*; Springer: Cham, Switzerland, 2011; ISBN 978-3-642-16518-4.
205. Rahm, E. Towards Large-Scale Schema and Ontology Matching. In *Schema Matching and Mapping*; Springer: Cham, Switzerland, 2011; pp. 3–27. ISBN 978-3-642-16518-4. [CrossRef]
206. Otero-Cerdeira, L.; Rodríguez-Martínez, F.J.; Gómez-Rodríguez, A. Ontology matching: A literature review. *Expert Syst. Appl.* **2015**, *42*. [CrossRef]
207. Do, H.H.; Rahm, E. COMA—A system for flexible combination of schema matching approaches. In Proceedings of the 28th International Conference on Very Large Databases (VLDB), Hong Kong, China, 20–23 August 2002; pp. 610–621.
208. Zhang, Y.; Wang, X.; Lai, S.; He, S.; Liu, K.; Zhao, J.; Lv, X. Ontology Matching with Word Embeddings. In Proceedings of the Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data—13th China National Conference, CCL 2014, and Second International Symposium, NLP-NABD 2014, Wuhan, China, 18–19 October 2014; Volume 8801, pp. 34–45. [CrossRef]
209. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed Representations of Words and Phrases and their Compositionality. In Proceedings of the Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013, Lake Tahoe, NV, USA, 5–8 December 2013; pp. 3111–3119.
210. Ayala, D.; Hernández, I.; Ruiz, D.; Rahm, E. LEAPME: Learning-based Property Matching with Embeddings. *Data Knowl. Eng.* **2022**, *137*, 101943. [CrossRef]
211. Portisch, J.; Costa, G.; Stefani, K.; Kreplin, K.; Hladik, M.; Paulheim, H. Ontology Matching Through Absolute Orientation of Embedding Spaces. In Proceedings of the Semantic Web: ESWC 2022 Satellite Events, Hersonissos, Greece, 29 May–2 June 2022; Volume 13384, pp. 153–157. [CrossRef]
212. Portisch, J.; Hladik, M.; Paulheim, H. RDF2Vec Light—A Lightweight Approach for Knowledge Graph Embeddings. *arXiv* **2020**, arXiv:2009.07659. [CrossRef]
213. Qiang, Z.; Wang, W.; Taylor, K. Agent-OM: Leveraging Large Language Models for Ontology Matching. *arXiv* **2023**, arXiv:2312.00326. [CrossRef]
214. Hertling, S.; Paulheim, H. OLaLa: Ontology Matching with Large Language Models. In Proceedings of the 12th Knowledge Capture Conference 2023, Pensacola, FL, USA, 5–7 December 2023; ACM: New York, NY, USA; pp. 5–7. [CrossRef]
215. Pottinger, R.A.; Bernstein, P.A. Merging models based on given correspondences. In Proceedings of the 2003 VLDB Conference, Berlin, Germany, 9–12 September 2003; Elsevier: Amsterdam, The Netherlands, 2003; pp. 862–873.
216. Raunich, S.; Rahm, E. Target-driven merging of taxonomies with ATOM. *Inf. Syst.* **2014**, *42*, 1–14. [CrossRef]
217. Osman, I.; Yahia, S.B.; Diallo, G. Ontology integration: Approaches and challenging issues. *Inf. Fusion* **2021**, *71*, 38–63. [CrossRef]
218. Usbeck, R.; Ngonga Ngomo, A.C.; Auer, S.; Gerber, D.; Both, A. AGDISTIS—Graph-Based Disambiguation of Named Entities using Linked Data. In Proceedings of the 13th International Semantic Web Conference, Riva del Garda, Italy, 19–23 October 2014.
219. Ferragina, P.; Scaiella, U. TAGME: On-the-fly annotation of short text fragments (by wikipedia entities). In Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, ON, Canada, 26–30 October 2010; pp. 1625–1628. [CrossRef]
220. Piccinno, F.; Ferragina, P. From TagME to WAT: A New Entity Annotator. In Proceedings of the First International Workshop on Entity Recognition & Disambiguation, New York, NY, USA, 11 July 2014; ERD '14, pp. 55–62. [CrossRef]

221. Manning, C.; Surdeanu, M.; Bauer, J.; Finkel, J.; Bethard, S.; McClosky, D. The Stanford CoreNLP Natural Language Processing Toolkit. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Baltimore, MD, USA, 23–24 June 2014; pp. 55–60. [CrossRef]
222. Goldberg, Y. *Neural Network Methods for Natural Language Processing*; Synthesis Lectures on Human Language Technologies; Morgan & Claypool Publishers: San Rafael, CA, USA, 2017; ISBN 978-3-031-01037-8. [CrossRef]
223. Li, J.; Sun, A.; Han, J.; Li, C. A Survey on Deep Learning for Named Entity Recognition. *IEEE Trans. Knowl. Data Eng.* **2022**, *34*, 50–70. [CrossRef]
224. Devlin, J.; Chang, M.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, 2–7 June 2019; Volume 1 (Long and Short Papers); Burstein, J., Doran, C., Solorio, T., Eds.; Association for Computational Linguistics: Stroudsburg, PA, USA, 2019; pp. 4171–4186. [CrossRef]
225. Harnoune, A.; Rhanoui, M.; Mikram, M.; Yousfi, S.; Elkaimbillah, Z.; El Asri, B. BERT Based Clinical Knowledge Extraction for Biomedical Knowledge Graph Construction and Analysis. *Comput. Methods Programs Biomed. Update* **2021**, *1*, 100042. [CrossRef]
226. Caufield, J.H.; Hegde, H.; Emonet, V.; Harris, N.L.; Joachimiak, M.P.; Matentzoglu, N.; Kim, H.; Moxon, S.A.T.; Reese, J.T.; Haendel, M.A.; et al. Structured Prompt Interrogation and Recursive Extraction of Semantics (SPIRES): A method for populating knowledge bases using zero-shot learning. *Bioinformatics* **2024**, *40*, btae104. [CrossRef] [PubMed]
227. Moon, S.; Neves, L.; Carvalho, V. Multimodal Named Entity Recognition for Short Social Media Posts. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), New Orleans, LA, USA, 1–6 June 2018; pp. 852–860. [CrossRef]
228. Yu, J.; Jiang, J.; Yang, L.; Xia, R. Improving Multimodal Named Entity Recognition via Entity Span Detection with Unified Multimodal Transformer. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Virtual, 5–10 July 2020; pp. 3342–3352. [CrossRef]
229. Pezeshkpour, P.; Chen, L.; Singh, S. Embedding Multimodal Relational Data for Knowledge Base Completion. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP), Brussels, Belgium, 31 October–4 November 2018; pp. 3208–3218. [CrossRef]
230. Li, M.; Zareian, A.; Lin, Y.; Pan, X.; Whitehead, S.; Chen, B.; Wu, B.; Ji, H.; Chang, S.F.; Voss, C.; et al. GAIA: A Fine-Grained Multimedia Knowledge Extraction System. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Virtual, 5–10 July 2020; pp. 77–86. [CrossRef]
231. Ding, Y.; Yu, J.; Liu, B.; Hu, Y.; Cui, M.; Wu, Q. MuKEA: Multimodal Knowledge Extraction and Accumulation for Knowledge-Based Visual Question Answering. *arXiv* **2022**, arXiv:2203.09138. [CrossRef]
232. Martinez-Rodriguez, J.L.; Hogan, A.; Lopez-Arevalo, I. Information extraction meets the Semantic Web: A survey. *Semant. Web* **2020**, *11*, 255–335. [CrossRef]
233. Kulkarni, S.; Singh, A.; Ramakrishnan, G.; Chakrabarti, S. Collective annotation of Wikipedia entities in web text. In Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining–KDD, Paris, France, 28 June–1 July 2009; ACM Press: New York, NY, USA, 2009. [CrossRef]
234. Milne, D.; Witten, I.H. Learning to link with wikipedia. In Proceedings of the 17th ACM Conference on Information and Knowledge Mining—CIKM, Napa Valley, CA, USA, 26–30 October 2008; ACM Press: New York, NY, USA, 2008. [CrossRef]
235. Han, X.; Sun, L.; Zhao, J. Collective entity linking in web text: A graph-based method. In Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, Beijing, China, 25–29 July 2011; Ma, W., Nie, J., Baeza-Yates, R., Chua, T., Croft, W.B., Eds.; ACM: New York, NY, USA, 2011; pp. 765–774. [CrossRef]
236. Medelyan, O.; Witten, I.H.; Milne, D. Topic Indexing with Wikipedia. In Proceedings of the First AAAI Workshop on Wikipedia and Artificial Intelligence (WIKIAI 2008), Washington, DC, USA, 13–14 July 2008.
237. Hoffart, J.; Milchevski, D.; Weikum, G.; Anand, A.; Singh, J. The Knowledge Awakens: Keeping Knowledge Bases Fresh with Emerging Entities. In Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, QC, Canada, 11–15 April 2016; Companion Volume; Bourdeau, J., Hendler, J., Nkambou, R., Horrocks, I., Zhao, B.Y., Eds.; ACM: New York, NY, USA, 2016; pp. 203–206. [CrossRef]
238. Mudgal, S.; Li, H.; Rekatsinas, T.; Doan, A.; Park, Y.; Krishnan, G.; Deep, R.; Arcaute, E.; Raghavendra, V. Deep Learning for Entity Matching: A Design Space Exploration. In Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, 10–15 June 2018; Das, G., Jermaine, C.M., Bernstein, P.A., Eds.; ACM: New York, NY, USA, 2018; pp. 19–34. [CrossRef]
239. Hearst, M.A. Automatic Acquisition of Hyponyms from Large Text Corpora. In Proceedings of the 14th International Conference on Computational Linguistics, COLING 1992, Nantes, France, 23–28 August 1992; pp. 539–545.
240. Agichtein, E.; Gravano, L. *Snowball*: Extracting relations from large plain-text collections. In Proceedings of the Fifth ACM Conference on Digital Libraries, San Antonio, TX, USA, 2–7 June 2000; ACM: New York, NY, USA, 2000; pp. 85–94. [CrossRef]
241. Brin, S. Extracting Patterns and Relations from the World Wide Web. In Proceedings of the World Wide Web and Databases, International Workshop WebDB'98, Valencia, Spain, 27–28 March 1998; Selected Papers; Lecture Notes in Computer Science; Atzeni, P., Mendelzon, A.O., Mecca, G., Eds.; Springer: Cham, Switzerland, 1998; Volume 1590, pp. 172–183. [CrossRef]

242. Zhou, G.; Zhang, M.; Ji, D.H.; Zhu, Q. Tree Kernel-Based Relation Extraction with Context-Sensitive Structured Parse Tree Information. In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), Prague, Czech Republic, 28–30 June 2007; pp. 728–736.
243. Nguyen, T.H.; Grishman, R. Relation Extraction: Perspective from Convolutional Neural Networks. In Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing, VS@NAACL-HLT 2015, Denver, CO, USA, 5 June 2015; pp. 39–48. [CrossRef]
244. Zeng, D.; Liu, K.; Chen, Y.; Zhao, J. Distant Supervision for Relation Extraction via Piecewise Convolutional Neural Networks. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, 17–21 September 2015; Màrquez, L., Callison-Burch, C., Su, J., Pighin, D., Marton, Y., Eds.; ACL: Stroudsburg, PA, USA, 2015; pp. 1753–1762. [CrossRef]
245. Baldini Soares, L.; FitzGerald, N.; Ling, J.; Kwiatkowski, T. Matching the Blanks: Distributional Similarity for Relation Learning. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 2895–2905. [CrossRef]
246. Wu, S.; He, Y. Enriching Pre-trained Language Model with Entity Information for Relation Classification. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, 3–7 November 2019; pp. 2361–2364. [CrossRef]
247. Han, X.; Gao, T.; Lin, Y.; Peng, H.; Yang, Y.; Xiao, C.; Liu, Z.; Li, P.; Zhou, J.; Sun, M. More Data, More Relations, More Context and More Openness: A Review and Outlook for Relation Extraction. In Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, Suzhou, China, 4–7 December 2020; pp. 745–758.
248. Chung, H.W.; Hou, L.; Longpre, S.; Zoph, B.; Tay, Y.; Fedus, W.; Li, Y.; Wang, X.; Dehghani, M.; Brahma, S.; et al. Scaling instruction-finetuned language models. *J. Mach. Learn. Res.* **2024**, *25*, 1–53.
249. Chen, X.; Zhang, N.; Xie, X.; Deng, S.; Yao, Y.; Tan, C.; Huang, F.; Si, L.; Chen, H. KnowPrompt: Knowledge-aware Prompt-tuning with Synergistic Optimization for Relation Extraction. In Proceedings of the WWW '22: The ACM Web Conference 2022, Lyon, France, 25–29 April 2022; Laforest, F., Troncy, R., Simperl, E., Agarwal, D., Gionis, A., Herman, I., Médini, L., Eds.; ACM: New York, NY, USA, 2022; pp. 2778–2788. [CrossRef]
250. Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A.A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. Overcoming catastrophic forgetting in neural networks. *Proc. Natl. Acad. Sci. USA* **2017**, *114*, 3521–3526. [CrossRef] [PubMed]
251. Hu, C.; Yang, D.; Jin, H.; Chen, Z.; Xiao, Y. Improving Continual Relation Extraction through Prototypical Contrastive Learning. In Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, 12–17 October 2022; pp. 1885–1895.
252. Zhao, K.; Xu, H.; Yang, J.; Gao, K. Consistent Representation Learning for Continual Relation Extraction. In Proceedings of the Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, 22–27 May 2022; pp. 3402–3411. [CrossRef]
253. Vashishth, S.; Jain, P.; Talukdar, P.P. CESI: Canonicalizing Open Knowledge Bases using Embeddings and Side Information. In Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, 23–27 April 2018; pp. 1317–1327. [CrossRef]
254. Daiber, J.; Jakob, M.; Hokamp, C.; Mendes, P.N. Improving Efficiency and Accuracy in Multilingual Entity Extraction. In Proceedings of the 9th International Conference on Semantic Systems (I-Semantics), Graz, Austria, 4–6 September 2013.
255. Clancy, R.; Ilyas, I.F.; Lin, J. Scalable Knowledge Graph Construction from Text Collections. In Proceedings of the Second Workshop on Fact Extraction and VERification (FEVER), Hong Kong, China, 3 November 2019; pp. 39–46. [CrossRef]
256. Han, X.; Gao, T.; Yao, Y.; Ye, D.; Liu, Z.; Sun, M. OpenNRE: An Open and Extensible Toolkit for Neural Relation Extraction. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations, Hong Kong, China, 3 November 2019; pp. 169–174. [CrossRef]
257. Elliott, D.; Keller, F. Image Description using Visual Dependency Representations. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, WA, USA, 18–21 October 2013; pp. 1292–1302.
258. Zheng, C.; Feng, J.; Fu, Z.; Cai, Y.; Li, Q.; Wang, T. Multimodal Relation Extraction with Efficient Graph Alignment. In Proceedings of the 29th ACM International Conference on Multimedia, New York, NY, USA, 20–24 October 2021; pp. 5298–5306. [CrossRef]
259. Köpcke, H.; Rahm, E. Frameworks for entity matching: A comparison. *Data Knowl. Eng.* **2010**, *69*, 197–210. [CrossRef]
260. Christen, P. The data matching process. In *Data-Centric Systems and Applications*; Springer: Cham, Switzerland, 2012; pp. 23–35. ISBN 978-3-642-31164-2.
261. Nentwig, M.; Hartung, M.; Ngomo, A.N.; Rahm, E. A survey of current Link Discovery frameworks. *Semant. Web* **2017**, *8*, 419–436. [CrossRef]
262. Barlaug, N.; Gulla, J.A. Neural networks for entity matching: A survey. *ACM Trans. Knowl. Discov. Data (TKDD)* **2021**, *15*, 1–37. [CrossRef]
263. Christophides, V.; Efthymiou, V.; Palpanas, T.; Papadakis, G.; Stefanidis, K. An Overview of End-to-End Entity Resolution for Big Data. *ACM Comput. Surv.* **2020** 53, 1–42. [CrossRef]

264. Papadakis, G.; Skoutas, D.; Thanos, E.; Palpanas, T. Blocking and filtering techniques for entity resolution: A survey. *ACM Comput. Surv. (CSUR)* **2020**, *53*, 1–42. [CrossRef]
265. Saeedi, A.; Peukert, E.; Rahm, E. Using link features for entity clustering in Knowledge Graphs. In Proceedings of the European Semantic Web Conference (EWSC) 2018, Heraklion, Crete, Greece, 3–7 June 2018; Springer: Cham, Switzerland, 2018; pp. 576–592.
266. Papadakis, G.; Tsekouras, L.; Thanos, E.; Pittaras, N.; Simonini, G.; Skoutas, D.; Isaris, P.; Giannakopoulos, G.; Palpanas, T.; Koubarakis, M. JedAI3: Beyond batch, blocking-based Entity Resolution. In Proceedings of the 23th EDBT, Copenhagen, Denmark, 30 March–2 April 2020; pp. 603–606.
267. Ebraheem, M.; Thirumuruganathan, S.; Joty, S.R.; Ouzzani, M.; Tang, N. DeepER—Deep Entity Resolution. *arXiv* **2017**, arXiv:1710.00597. [CrossRef]
268. Sun, Z.; Zhang, Q.; Hu, W.; Wang, C.; Chen, M.; Akrami, F.; Li, C. A Benchmarking Study of Embedding-based Entity Alignment for Knowledge Graphs. *Proc. VLDB Endow.* **2020**, *13*, 2326–2340. [CrossRef]
269. Obraczka, D.; Schuchart, J.; Rahm, E. Embedding-Assisted Entity Resolution for Knowledge Graphs. In Proceedings of the 2nd International Workshop on Knowledge Graph Construction Co-Located with 18th Extended Semantic Web Conference (ESWC 2021), Online, 6 June 2021; Volume 2873.
270. Leone, M.; Huber, S.; Arora, A.; García-Durán, A.; West, R. A Critical Re-Evaluation of Neural Methods for Entity Alignment. *Proc. VLDB Endow.* **2022**, *15*, 1712–1725. [CrossRef]
271. Papadakis, G.; Ioannou, E.; Thanos, E.; Palpanas, T. *The Four Generations of Entity Resolution*. Synthesis Lectures on Data Management; Springer: Cham, Switzerland, 2021; ISBN 978-3-031-00750-7. [CrossRef]
272. Wang, Y.; Cui, Y.; Liu, W.; Sun, Z.; Jiang, Y.; Han, K.; Hu, W. Facing Changes: Continual Entity Alignment for Growing Knowledge Graphs. In Proceedings of the Semantic Web-ISWC 2022—21st International Semantic Web Conference, Virtual Event, 23–27 October 2022; Volume 13489, pp. 196–213. [CrossRef]
273. Gruenheid, A.; Dong, X.L.; Srivastava, D. Incremental record linkage. *Proc. VLDB Endow.* **2014**, *7*, 697–708. [CrossRef]
274. Gazzarri, L.; Herschel, M. End-to-end Task Based Parallelization for Entity Resolution on Dynamic Data. In Proceedings of the 2021 IEEE 37th International Conference on Data Engineering (ICDE), Chania, Greece, 19–22 April 2021; pp. 1248–1259.
275. Saeedi, A.; Nentwig, M.; Peukert, E.; Rahm, E. Scalable matching and clustering of entities with FAMER. *Complex Syst. Inform. Model. Q.* **2018**, *16*, 61–83. [CrossRef]
276. Ramadan, B.; Christen, P. Forest-Based Dynamic Sorted Neighborhood Indexing for Real-Time Entity Resolution. In Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM), Shanghai, China, 3–7 November 2014; pp. 1787–1790. [CrossRef]
277. Ramadan, B.; Christen, P.; Liang, H.; Gayler, R.W. Dynamic Sorted Neighborhood Indexing for Real-Time Entity Resolution. *J. Data Inf. Qual.* **2015**, *6*. [CrossRef]
278. Karapiperis, D.; Gkoulalas-Divanis, A.; Verykios, V.S. Summarization Algorithms for Record Linkage. In Proceedings of the EDBT, Vienna, Austria, 26–29 March 2018; pp. 73–84.
279. Brasileiro Araújo, T.; Stefanidis, K.; Santos Pires, C.E.; Nummenmaa, J.; Pereira da Nóbrega, T. Incremental blocking for entity resolution over web streaming data. In Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence; Thessaloniki, Greece, 14–17 October 2019; pp. 332–336.
280. Araújo, T.B.; Stefanidis, K.; Santos Pires, C.E.; Nummenmaa, J.; Da Nóbrega, T.P. Schema-agnostic blocking for streaming data. In Proceedings of the 35th Annual ACM Symposium on Applied Computing, Brno, Czech Republic, 30 March–3 April 2020; pp. 412–419.
281. Javdani, D.; Rahmani, H.; Allahgholi, M.; Karimkhani, F. DeepBlock: A Novel Blocking Approach for Entity Resolution using Deep Learning. In Proceedings of the 2019 5th International Conference on Web Research (ICWR), Tehran, Iran, 24–25 April 2019; pp. 41–44.
282. Zhang, W.; Wei, H.; Sisman, B.; Dong, X.L.; Faloutsos, C.; Page, D. AutoBlock: A Hands-off Blocking Framework for Entity Matching. In Proceedings of the WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, 3–7 February 2020; pp. 744–752. [CrossRef]
283. Thirumuruganathan, S.; Li, H.; Tang, N.; Ouzzani, M.; Govind, Y.; Paulsen, D.; Fung, G.; Doan, A. Deep Learning for Blocking in Entity Matching: A Design Space Exploration. *Proc. VLDB Endow.* **2021**, *14*, 2459–2472. [CrossRef]
284. Hassanzadeh, O.; Chiang, F.; Lee, H.C.; Miller, R.J. Framework for evaluating clustering algorithms in duplicate detection. *Proc. VLDB Endow.* **2009**, *2*, 1282–1293. [CrossRef]
285. Saeedi, A.; Peukert, E.; Rahm, E. Comparative evaluation of distributed clustering schemes for multi-source entity resolution. In Proceedings of the European Conference on Advances in Databases and Information Systems (ADBIS), Nicosia, Cyprus, 24–27 September 2017; Springer: Cham, Switzerland, 2017; pp. 278–293.
286. Welch, J.M.; Sane, A.; Drome, C. Fast and accurate incremental entity resolution relative to an entity knowledge base. In Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIRK 2012), Maui, HI, USA, 29 October–2 November 2012; pp. 2667–2670.
287. Brunner, U.; Stockinger, K. Entity Matching with Transformer Architectures—A Step Forward in Data Integration. In Proceedings of the 23rd International Conference on Extending Database Technology, EDBT 2020, Copenhagen, Denmark, 30 March–2 April 2020; pp. 463–473. [CrossRef]

288. Li, Y.; Li, J.; Suhara, Y.; Doan, A.; Tan, W. Deep Entity Matching with Pre-Trained Language Models. *Proc. VLDB Endow.* **2020**, *14*, 50–60. [CrossRef]
289. Peeters, R.; Bizer, C. Dual-Objective Fine-Tuning of BERT for Entity Matching. *Proc. VLDB Endow.* **2021**, *14*, 1913–1921. [CrossRef]
290. Ge, C.; Wang, P.; Chen, L.; Liu, X.; Zheng, B.; Gao, Y. CollaborEM: A Self-Supervised Entity Matching Framework Using Multi-Features Collaboration. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 12139–12152. [CrossRef]
291. Yao, D.; Gu, Y.; Cong, G.; Jin, H.; Lv, X. Entity Resolution with Hierarchical Graph Attention Networks. In Proceedings of the SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, 12–17 June 2022; Ives, Z.G., Bonifati, A., Abbadi, A.E., Eds.; ACM: New York, NY, USA, 2022; pp. 429–442. [CrossRef]
292. Tu, J.; Fan, J.; Tang, N.; Wang, P.; Chai, C.; Li, G.; Fan, R.; Du, X. Domain Adaptation for Deep Entity Resolution. In Proceedings of the SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, 12–17 June 2022; Ives, Z.G., Bonifati, A., Abbadi, A.E., Eds.; ACM: New York, NY, USA, 2022; pp. 443–457. [CrossRef]
293. Tang, J.; Zuo, Y.; Cao, L.; Madden, S. Generic entity resolution models. In Proceedings of the NeurIPS 2022 First Table Representation Workshop, New Orleans, LA, USA, 2 December 2022.
294. Zhang, R.; Su, Y.; Trisedya, B.D.; Zhao, X.; Yang, M.; Cheng, H.; Qi, J. AutoAlign: Fully Automatic and Effective Knowledge Graph Alignment Enabled by Large Language Models. *IEEE Trans. Knowl. Data Eng.* **2024**, *36*, 2357–2371. [CrossRef]
295. Li, Q.; Ji, C.; Guo, S.; Liang, Z.; Wang, L.; Li, J. Multi-Modal Knowledge Graph Transformer Framework for Multi-Modal Entity Alignment. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, 6–10 December 2023; pp. 987–999. [CrossRef]
296. Bai, L.; Song, X.; Zhu, L. Joint Multi-Feature Information Entity Alignment for Cross-Lingual Temporal Knowledge Graph with BERT. *IEEE Trans. Big Data* 2024, 1–13. [CrossRef]
297. Fanourakis, N.; Lekbour, F.; Efthymiou, V.; Renton, G.; Christophides, V. HybEA: Hybrid Attention Models for Entity Alignment. *arXiv* **2024**, arXiv:2407.02862. [CrossRef]
298. Bleiholder, J.; Naumann, F. Data fusion. *ACM Comput. Surv. (CSUR)* **2009**, *41*, 1–41. [CrossRef]
299. Bizer, C.; Becker, C.; Mendes, P.N.; Isele, R.; Matteini, A.; Schultz, A. Ldif—A framework for large-scale Linked Data integration. In Proceedings of the (WWW) 2012 Developer Track, Lyon, France, 18–20 April 2012.
300. Mendes, P.N.; Mühleisen, H.; Bizer, C. Sieve: Linked data quality assessment and fusion. In Proceedings of the 2012 Joint EDBT/ICDT Workshops, Berlin, Germany, 30 March 2012; pp. 116–123.
301. Dong, X.; Berti-Équille, L.; Srivastava, D. Data Fusion: Resolving Conflicts from Multiple Sources. In Proceedings of the Interational Conference on Web-Age Information Management (WAIM 2013), Beidaihe, China, 14–16 June 2013.
302. Angles, R.; Thakkar, H.; Tomaszuk, D. RDF and Property Graphs Interoperability: Status and Issues. In Proceedings of the 13th Alberto Mendelzon International Workshop on Foundations of Data Management, Asunción, Paraguay, 3–7 June 2019; Volume 2369.
303. Paulheim, H.; Bizer, C. Type Inference on Noisy RDF Data. In Proceedings of the Semantic Web-ISWC 2013—12th International Semantic Web Conference, Sydney, Australia, 21–25 October 2013; Volume 8218, pp. 510–525. [CrossRef]
304. Paulheim, H.; Bizer, C. Improving the Quality of Linked Data Using Statistical Distributions. *Int. J. Semant. Web Inf. Syst.* **2014**, *10*, 63–86. [CrossRef]
305. Lutov, A.; Roshankish, S.; Khayati, M.; Cudré-Mauroux, P. StaTIX—Statistical Type Inference on Linked Data. In Proceedings of the IEEE International Conference on Big Data, Big Data 2018, Seattle, WA, USA, 10–13 December 2018; pp. 2253–2262. [CrossRef]
306. Zhao, Y.; Zhang, A.; Xie, R.; Liu, K.; Wang, X. Connecting Embeddings for Knowledge Graph Entity Typing. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online Event, 5–10 July 2020; pp. 6419–6428. [CrossRef]
307. Aprosio, A.P.; Giuliano, C.; Lavelli, A. Extending the Coverage of DBpedia Properties Using Distant Supervision over Wikipedia. In Proceedings of the NLP & DBpedia Workshop Co-Located with the 12th International Semantic Web Conference (ISWC 2013), Sydney, Australia, 21–25 October 2013; Springer: Cham, Switzerland, 2013.
308. Gerber, D.; Hellmann, S.; Bühmann, L.; Soru, T.; Usbeck, R.; Ngonga Ngomo, A.C. Real-time RDF extraction from unstructured data streams. In Proceedings of the International Semantic Web Conference (ISWC), Sydney, Australia, 21–25 October 2013; Springer: Cham, Switzerland, 2013; pp. 135–150.
309. Gerber, D.; Ngomo, A.C.N. Bootstrapping the Linked Data web. In Proceedings of the 1st Workshop on Web Scale Knowledge Extraction@ ISWC, Bonn, Germany, October 2011; Volume 2011, p. 61.
310. Mintz, M.; Bills, S.; Snow, R.; Jurafsky, D. Distant supervision for relation extraction without labeled data. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, Singapore, 2–7 August 2009; pp. 1003–1011.
311. West, R.; Gabrilovich, E.; Murphy, K.; Sun, S.; Gupta, R.; Lin, D. Knowledge base completion via search-based question answering. In Proceedings of the 23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, 7–11 April 2014; pp. 515–526. [CrossRef]
312. Lange, D.; Böhm, C.; Naumann, F. Extracting structured information from Wikipedia articles to populate infoboxes. In Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, ON, Canada, 26–30 October 2010; pp. 1661–1664. [CrossRef]
313. Fields, C.R. Probabilistic models for segmenting and labeling sequence data. In Proceedings of the ICML 2001, San Francisco, CA, USA, 28 June–1 July 2001.

314. Blevins, T.; Zettlemoyer, L. Moving Down the Long Tail of Word Sense Disambiguation with Gloss Informed Bi-encoders. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Virtual, 5–10 July 2020; ACM: New York, NY, USA; pp. 1006–1017. [CrossRef]
315. Munoz, E.; Hogan, A.; Mileo, A. Triplifying wikipedia's tables. In Proceedings of the First International Conference on Linked Data for Information Extraction (LD4IE), Sydney, Australia, 21 October 2013; pp. 26–37.
316. Ritze, D.; Lehmberg, O.; Bizer, C. Matching html tables to dbpedia. In Proceedings of the 5th International Conference on Web Intelligence, Mining and Semantics (WIMS), Larnaca, Cyprus, 13–15 July 2015; pp. 1–6.
317. Paulheim, H.; Ponzetto, S.P. Extending DBpedia with Wikipedia List Pages. In Proceedings of the 2013th International Conference on NLP & DBpedia, Sydney, Australia, 22 October 2013; pp. 85–90.
318. Bordes, A.; Usunier, N.; García-Durán, A.; Weston, J.; Yakhnenko, O. Translating Embeddings for Modeling Multi-relational Data. In Proceedings of the Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013, Lake Tahoe, NV, USA, 5–8 December 2013; pp. 2787–2795.
319. Wang, Z.; Zhang, J.; Feng, J.; Chen, Z. Knowledge Graph Embedding by Translating on Hyperplanes. In Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, Québec City, QC, Canada, 27 –31 July 2014; pp. 1112–1119.
320. Kolyvakis, P.; Kalousis, A.; Kiritsis, D. HyperKG: Hyperbolic Knowledge Graph Embeddings for Knowledge Base Completion. *arXiv* **2019**, arXiv:1908.04895. [CrossRef]
321. Ali, M.; Berrendorf, M.; Hoyt, C.T.; Vermue, L.; Galkin, M.; Sharifzadeh, S.; Fischer, A.; Tresp, V.; Lehmann, J. Bringing Light Into the Dark: A Large-scale Evaluation of Knowledge Graph Embedding Models under a Unified Framework. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, 44, 8825–8845. [CrossRef]
322. Teru, K.K.; Denis, E.G.; Hamilton, W.L. Inductive Relation Prediction by Subgraph Reasoning. In Proceedings of the 37th International Conference on Machine Learning, ICML 2020, Virtual, 13–18 July 2020; Volume 119, pp. 9448–9457.
323. Galkin, M.; Denis, E.; Wu, J.; Hamilton, W.L. NodePiece: Compositional and Parameter-Efficient Representations of Large Knowledge Graphs. In Proceedings of the International Conference on Learning Representations (ICLR), Online Event, 25–29 April 2022.
324. Galárraga, L.A.; Teflioudi, C.; Hose, K.; Suchanek, F. AMIE: Association rule mining under incomplete evidence in ontological knowledge bases. In Proceedings of the 22nd International Conference on World Wide Web (WWW), Rio de Janeiro, Brazil, 13–17 May 2013; pp. 413–422. [CrossRef]
325. Cheng, K.; Ahmed, N.K.; Sun, Y. Neural Compositional Rule Learning for Knowledge Graph Reasoning. In Proceedings of the Eleventh International Conference on Learning Representations, (ICLR) 2023, Kigali, Rwanda, 1–5 May 2023.
326. Romero, A.A.; Grau, B.C.; Horrocks, I. MORe: Modular Combination of OWL Reasoners for Ontology Classification. In Proceedings of the Semantic Web–ISWC 2012—11th International Semantic Web Conference, Boston, MA, USA, 11–15 November 2012; Volume 7649, pp. 1–16. [CrossRef]
327. Wang, C.; Feng, Z.; Zhang, X.; Wang, X.; Rao, G.; Fu, D. ComR: A combined OWL reasoner for ontology classification. *Front. Comput. Sci.* **2019**, *13*, 139–156. [CrossRef]
328. Yao, L.; Mao, C.; Luo, Y. KG-BERT: BERT for Knowledge Graph Completion. *arXiv* **2019**, arXiv:1909.03193. [CrossRef]
329. Choi, B.; Jang, D.; Ko, Y. MEM-KGC: Masked Entity Model for Knowledge Graph Completion with Pre-Trained Language Model. *IEEE Access* **2021**, *9*, 132025–132032. [CrossRef]
330. Veseli, B.; Singhania, S.; Razniewski, S.; Weikum, G. Evaluating Language Models for Knowledge Base Completion. In Proceedings of the Semantic Web—20th International Conference, ESWC 2023, Hersonissos, Greece, 28 May–1 June 2023; Volume 13870, pp. 227–243. [CrossRef]
331. Omeliyanenko, J.; Zehe, A.; Hotho, A.; Schlör, D. CapsKG: Enabling Continual Knowledge Integration in Language Models for Automatic Knowledge Graph Completion. In Proceedings of the Semantic Web—ISWC 2023—22nd International Semantic Web Conference, Athens, Greece, 6–10 November 2023; Volume 14265, pp. 618–636. [CrossRef]
332. Liu, P.; Yuan, W.; Fu, J.; Jiang, Z.; Hayashi, H.; Neubig, G. Pre-Train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ACM Comput. Surv.* **2023**, *55*, 195:1–195:35. [CrossRef]
333. Sun, M.; Zhou, K.; He, X.; Wang, Y.; Wang, X. GPPT: Graph Pre-training and Prompt Tuning to Generalize Graph Neural Networks. In Proceedings of the KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 14–18 August 2022; pp. 1717–1727. [CrossRef]
334. Sun, X.; Cheng, H.; Li, J.; Liu, B.; Guan, J. All in One: Multi-Task Prompting for Graph Neural Networks. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, 6–10 August 2023; pp. 2120–2131. [CrossRef]
335. Wang, R.Y.; Strong, D.M. Beyond Accuracy: What Data Quality Means to Data Consumers. *J. Manag. Inf. Syst.* **1996**, *12*, 5–33. [CrossRef]
336. Tartir, S.; Arpinar, I.B.; Moore, M.; Sheth, A.P.; Aleman-Meza, B. OntoQA: Metric-based ontology quality analysis. In Proceedings of the IEEE ICDM Workshop on Knowledge Acquisition from Distributed, Autonomous, Semantically Heterogeneous Data and Knowledge Sources, Houston, TX, USA, 27 November 2005.
337. McDaniel, M.; Storey, V.C. Evaluating domain ontologies: Clarification, classification, and challenges. *ACM Comput. Surv. (CSUR)* **2019**, *52*, 1–44. [CrossRef]
338. Bizer, C.; Cyganiak, R. Quality-driven information filtering using the WIQA policy framework. *J. Web Semant.* **2009**, *7*, 1–10. [CrossRef]

339. Acosta, M.; Zaveri, A.; Simperl, E.; Kontokostas, D.; Auer, S.; Lehmann, J. Crowdsourcing Linked Data quality assessment. In Proceedings of the International Semantic Web Conference (ISWC), Sydney, Australia, 21–25 October 2013; Springer: Cham, Switzerland, 2013; pp. 260–276.
340. Senaratne, A.; Omran, P.G.; Williams, G.J. Unsupervised Anomaly Detection in Knowledge Graphs. In Proceedings of the 10th International Joint Conference on Knowledge Graphs (IJCKG), Virtual, 6–8 December 2021.
341. Ma, Y.; Gao, H.; Wu, T.; Qi, G. Learning Disjointness Axioms With Association Rule Mining and Its Application to Inconsistency Detection of Linked Data. In Proceedings of the China Semantic Web Symposium (CSWS), Changsha, China, 5–7 September 2014.
342. Li, F.; Dong, X.L.; Langen, A.; Li, Y. Knowledge verification for long-tail verticals. *Proc. VLDB Endow.* **2017**, *10*, 1370–1381. [CrossRef]
343. Lehmann, J.; Gerber, D.; Morsey, M.; Ngonga Ngomo, A.C. Defacto-deep fact validation. In Proceedings of the International Semantic Web Conference (ISWC), Boston, MA, USA, 11–15 November 2012; Springer: Cham, Switzerland, 2012; pp. 312–327.
344. Tufek, N.; Saissre, A.; Hanbury, A. Validating Semantic Artifacts With Large Language Models. In Proceedings of the 21th European Semantic Web Conference (ESWC), Krete, Greece, 24–30 May 2024.
345. Chen, H.; Cao, G.; Chen, J.; Ding, J. A Practical Framework for Evaluating the Quality of Knowledge Graph. In Proceedings of the China Conference on Knowledge Graph and Semantic Computing (CCKS), Hangzhou, China, 24–27 August 2019.
346. Kontokostas, D.; Zaveri, A.; Auer, S.; Lehmann, J. TripleCheckMate: A Tool for Crowdsourcing the Quality Assessment of Linked Data. In Proceedings of the International Conference on Knowledge Engineering and the Semantic Web (KESW), St. Petersburg, Russia, 7–9 October 2013.
347. Carlson, A.; Betteridge, J.; Kisiel, B.; Settles, B.; Hruschka, E.; Mitchell, T. Toward an Architecture for Never-Ending Language Learning. In Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, GA, USA, 11–15 July 2010.
348. Kontokostas, D.; Westphal, P.; Auer, S.; Hellmann, S.; Lehmann, J.; Cornelissen, R.; Zaveri, A. Test-driven evaluation of Linked Data quality. In Proceedings of the 23rd international Conference on World Wide Web (WWW), Seoul, Republic of Korea, 7–11 April 2014.
349. Röder, M.; Kuchelev, D.; Ngomo, A.N. HOBBIT: A platform for benchmarking Big Linked Data. *Data Sci.* **2020**, *3*, 15–35. [CrossRef]
350. Hertling, S.; Paulheim, H. Gollum: A Gold Standard for Large Scale Multi Source Knowledge Graph Matching. In Proceedings of the 4th Conference on Automated Knowledge Base Construction, AKBC 2022, London, UK, 3–5 November 2022.
351. Safavi, T.; Koutra, D. CoDEx: A Comprehensive Knowledge Graph Completion Benchmark. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Virtual, 16–20 November 2020.
352. Li, Z.; Zhu, H.; Lu, Z.; Yin, M. Synthetic Data Generation with Large Language Models for Text Classification: Potential and Limitations. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, 6–10 December 2023; pp. 10443–10461.
353. Mihindukulasooriya, N.; Tiwari, S.; Enguix, C.F.; Lata, K. Text2KGBench: A Benchmark for Ontology-Driven Knowledge Graph Generation from Text. In Proceedings of the Semantic Web-ISWC 2023—22nd International Semantic Web Conference, Athens, Greece, 6–10 November 2023; Volume 14266, pp. 247–265. [CrossRef]
354. Meyer, L.; Frey, J.; Junghanns, K.; Brei, F.; Bulert, K.; Gründer-Fahrer, S.; Martin, M. Developing a Scalable Benchmark for Assessing Large Language Models in Knowledge Graph Engineering. In Proceedings of the Posters and Demo Track of the 19th International Conference on Semantic Systems Co-Located with 19th International Conference on Semantic Systems (SEMANTiCS 2023), Leipzing, Germany, 20–22 September 2023; Volume 3526.
355. Galkin, M.; Auer, S.; Vidal, M.E.; Scerri, S. Enterprise Knowledge Graphs: A Semantic Approach for Knowledge Management in the Next Generation of Enterprise Information Systems. In Proceedings of the 19th International Conference on Enterprise Information Systems (ICEIS), Porto, Portugal, 26–29 April 2017; pp. 88–98. [CrossRef]
356. Färber, M. The Microsoft Academic Knowledge Graph: A Linked Data Source with 8 Billion Triples of Scholarly Data. In Proceedings of the Semantic Web-ISWC 2019—18th International Semantic Web Conference, Auckland, New Zealand, 26–30 October 2019; Volume 11779, pp. 113–129. [CrossRef]
357. Bollacker, K.; Cook, R.; Tufts, P. Freebase: A shared database of structured general human knowledge. In Proceedings of the AAAI, Vancouver, BC, Canada, 22–26 July 2007; Volume 7, pp. 1962–1963.
358. Suchanek, F.M.; Kasneci, G.; Weikum, G. Yago: A core of semantic knowledge. In Proceedings of the 16th International Conference on World Wide Web (WWW), Banff, AB, Canada, 8–12 May 2007.
359. Pellissier Tanon, T.; Weikum, G.; Suchanek, F. Yago 4: A reasonable knowledge base. In Proceedings of the European Semantic Web Conference (ESWC), Virtual, 2–5 June 2020; pp. 583–596.
360. Vrandečić, D.; Krötzsch, M. Wikidata: A free collaborative knowledgebase. *Commun. ACM* **2014**, *57*, 78–85. [CrossRef]
361. Morsey, M.; Lehmann, J.; Auer, S.; Stadler, C.; Hellmann, S. DBpedia and the live extraction of structured data from wikipedia. *Program* **2012**, *46*, 157–181. [CrossRef]
362. Gawriljuk, G.; Harth, A.; Knoblock, C.A.; Szekely, P. A scalable approach to incrementally building Knowledge Graphs. In Proceedings of the International Conference on Theory and Practice of Digital Libraries (TPDL), Hannover, Germany, 5–9 September 2016; Springer: Cham, Switzerland, 2016; pp. 188–199.
363. Auer, S.; Oelen, A.; Haris, M.; Stocker, M.; D'Souza, J.; Farfar, K.E.; Vogt, L.; Prinz, M.; Wiens, V.; Jaradeh, M.Y. Improving access to scientific literature with Knowledge Graphs. *Bibl. Forsch. Prax.* **2020**, *44*, 516–529. [CrossRef]

364. Dessì, D.; Osborne, F.; Reforgiato Recupero, D.; Buscaldi, D.; Motta, E.; Sack, H. Ai-kg: An automatically generated Knowledge Graph of artificial intelligence. In Proceedings of the International Semantic Web Conference (ISWC), Athens, Greece, 2–6 November 2020; pp. 127–143.
365. Alberts, H.; Huang, N.; Deshpande, Y.; Liu, Y.; Cho, K.; Vania, C.; Calixto, I. VisualSem: A high-quality Knowledge Graph for vision and language. In Proceedings of the 1st Workshop on Multilingual Representation Learning, Punta Cana, Dominican Republic, 11 November 2021; Ataman, D., Birch, A., Conneau, A., Firat, O., Ruder, S., Sahin, G.G., Eds.; ACL: Stroudsburg, PA, USA, 2021; pp. 138–152. [CrossRef]
366. Dsouza, A.; Tempelmeier, N.; Yu, R.; Gottschalk, S.; Demidova, E. WorldKG: A World-Scale Geographic Knowledge Graph. In Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM), Virtual, 1–5 November 2021; ACM: New York, NY, USA, 2021.
367. Pellissier Tanon, T.; Vrandečić, D.; Schaffert, S.; Steiner, T.; Pintscher, L. From freebase to wikidata: The great migration. In Proceedings of the 25th International Conference on World Wide Web, Montréal, QC, Canada, 11–15 April 2016; ACM: New York, NY, USA, 2016; pp. 1419–1428.
368. Piscopo, A.; Kaffee, L.A.; Phethean, C.; Simperl, E. Provenance information in a collaborative Knowledge Graph: An evaluation of Wikidata external references. In Proceedings of the International Semantic Web Conference (ISWC) 2017, Vienna, Austria, 21–25 October 2017; Springer: Cham, Switzerland, 2017; pp. 542–558.
369. Zhang, Y.; Sheng, M.; Zhou, R.; Wang, Y.; Han, G.; Zhang, H.; Xing, C.; Dong, J. HKGB: An Inclusive, Extensible, Intelligent, Semi-auto-constructed Knowledge Graph Framework for Healthcare with Clinicians' Expertise Incorporated. *Inf. Process. Manag.* **2020**, *57*, 102324. [CrossRef]
370. Jaradeh, M.Y.; Singh, K.; Stocker, M.; Both, A.; Auer, S. Better Call the Plumber: Orchestrating Dynamic Information Extraction Pipelines. In Proceedings of the Web Engineering—21st International Conference, ICWE 2021, Biarritz, France, 18–21 May 2021; Volume 12706, pp. 240–254. [CrossRef]
371. Pan, Z.; Su, C.; Deng, Y.; Cheng, J.C.P. Image2Triplets: A computer vision-based explicit relationship extraction framework for updating construction activity Knowledge Graphs. *Comput. Ind.* **2022**, *137*, 103610. [CrossRef]
372. Cimmino, A.; García-Castro, R. Helio: A framework for implementing the life cycle of knowledge graphs. *Semant. Web* **2024**, *15*, 223–249. [CrossRef]
373. Kazakov, Y.; Klinov, P. Incremental Reasoning in OWL EL without Bookkeeping. In Proceedings of the Semantic Web—ISWC 2013—12th International Semantic Web Conference, Sydney, Australia, 21–25 October 2013; Volume 8218, pp. 232–247. [CrossRef]
374. Jagvaral, B.; Wangon, L.; Park, H.; Jeon, M.; Lee, N.; Park, Y. Large-scale incremental OWL/RDFS reasoning over fuzzy RDF data. In Proceedings of the 2017 IEEE International Conference on Big Data and Smart Computing, BigComp 2017, Jeju Island, Republic of Korea, 13–16 February 2017; pp. 269–273. [CrossRef]
375. Bhattarai, P.; Ghassemi, M.; Alhanai, T. Open-Source Code Repository Attributes Predict Impact of Computer Science Research. In Proceedings of the 22nd ACM/IEEE Joint Conference on Digital Libraries, Cologne, Germany, 20–24 June 2022; pp. 1–7. [CrossRef]
376. Mahdavi, M.; Neutatz, F.; Visengeriyeva, L.; Abedjan, Z. Towards automated data cleaning workflows. *Mach. Learn.* **2019**, *15*, 16.
377. Liang, K.; Meng, L.; Liu, M.; Liu, Y.; Tu, W.; Wang, S.; Zhou, S.; Liu, X.; Sun, F.; He, K. A Survey of Knowledge Graph Reasoning on Graph Types: Static, Dynamic, and Multi-Modal. *IEEE Trans. Pattern Anal. Mach. Intell.* **2024**, 1–20. [CrossRef] [PubMed]
378. Zhao, X.; Jia, Y.; Li, A.; Jiang, R.; Song, Y. Multi-source knowledge fusion: A survey. *World Wide Web* **2020**, *23*, 2567–2592. [CrossRef]
379. Shenoy, K.; Ilievski, F.; Garijo, D.; Schwabe, D.; Szekely, P.A. A study of the quality of Wikidata. *J. Web Semant.* **2022**, *72*, 100679. [CrossRef]
380. Nuzzolese, A.G.; Gentile, A.L.; Presutti, V.; Gangemi, A.; Garigliotti, D.; Navigli, R. Open Knowledge Extraction Challenge. In Proceedings of the SemWebEval (ESWC 2015), Portorož, Slovenia, 31 May–4 June 2015.
381. Rodríguez, J.M.; Merlino, H.D.; Pesado, P.; García-Martínez, R. Performance Evaluation of Knowledge Extraction Methods. In Proceedings of the International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE), Morioka, Japan, 2–4 August 2016.
382. Zhang, Y.; Zhong, V.; Chen, D.; Angeli, G.; Manning, C.D. Position-aware Attention and Supervised Data Improve Slot Filling. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Copenhagen, Denmark, 7–11 September 2017; ACL: Stroudsburg, PA, USA, 2017.
383. Euzenat, J.; Meilicke, C.; Stuckenschmidt, H.; Shvaiko, P.; dos Santos, C.T. Ontology Alignment Evaluation Initiative: Six Years of Experience. *J. Data Semant.* **2011**, *15*, 158–192.
384. Köpcke, H.; Thor, A.; Rahm, E. Evaluation of entity resolution approaches on real-world match problems. *Proc. VLDB Endow.* **2010**, *3*, 484–493. [CrossRef]
385. Galkin, M.; Berrendorf, M.; Hoyt, C.T. An Open Challenge for Inductive Link Prediction on Knowledge Graphs. *arXiv* **2022**, arXiv:2203.01520. [CrossRef]
386. Hu, W.; Fey, M.; Ren, H.; Nakata, M.; Dong, Y.; Leskovec, J. OGB-LSC: A Large-Scale Challenge for Machine Learning on Graphs. In Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, Online Event, 6–14 December 2021.
387. Portisch, J.; Hladik, M.; Paulheim, H. Background knowledge in ontology matching: A survey. *Semant. Web* **2022**, 1–55. [CrossRef]

388. Oliveira, I.L.; Fileto, R.; Speck, R.; Garcia, L.P.; Moussallem, D.; Lehmann, J. Towards holistic Entity Linking: Survey and directions. *Inf. Syst.* **2021**, *95*, 101624. [CrossRef]
389. Pan, J.Z.; Razniewski, S.; Kalo, J.; Singhania, S.; Chen, J.; Dietze, S.; Jabeen, H.; Omeliyanenko, J.; Zhang, W.; Lissandrini, M.; et al. Large Language Models and Knowledge Graphs: Opportunities and Challenges. *TGDK* **2023**, *1*, 38. [CrossRef]
390. Hofer, M.; Frey, J.; Rahm, E. Towards self-configuring Knowledge Graph Construction Pipelines using LLMs—A Case Study with RML. In Proceedings of the 5th International Workshop on Knowledge Graph Construction Co-Located with 21th Extended Semantic Web Conference (ESWC 2024), Hersonissos, Greece, 27 May 2024; Volume 3718.
391. Sansford, H.J.; Richardson, N.; Maretic, H.P.; Saada, J.N. GraphEval: A Knowledge-Graph Based LLM Hallucination Evaluation Framework. *arXiv* **2024**, arXiv:2407.10793. [CrossRef]
392. Wu, Z.; Qiu, L.; Ross, A.; Akyürek, E.; Chen, B.; Wang, B.; Kim, N.; Andreas, J.; Kim, Y. Reasoning or Reciting? Exploring the Capabilities and Limitations of Language Models Through Counterfactual Tasks. In Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Mexico City, Mexico, 16–21 June 2024; ACL: Stroudsburg, PA, USA, 2024; pp. 1819–1862.
393. Tamašauskaitė, G.; Groth, P. Defining a Knowledge Graph Development Process Through a Systematic Review. *ACM Trans. Softw. Eng. Methodol.* **2022** *32*, 1–40. [CrossRef]
394. Simsek, U.; Angele, K.; Kärle, E.; Opdenplatz, J.; Sommer, D.; Umbrich, J.; Fensel, D. Knowledge Graph Lifecycle: Building and maintaining Knowledge Graphs. In Proceedings of the 2nd International Workshop on Knowledge Graph Construction (KGC) Co-Located with 18th Extended Semantic Web Conference (ESWC 2021), Virtual, 6–10 June 2021.