Semantic Web 0 (2016) 1–0
IOS Press

Knowledge Graph Refinement: A Survey of Approaches and Evaluation Methods

Editor(s): Philipp Cimiano, Universität Bielefeld, Germany
Solicited review(s): Natasha Noy, Google Inc., USA; Philipp Cimiano, Universität Bielefeld, Germany; two anonymous reviewers

Heiko Paulheim,
Data and Web Science Group, University of Mannheim, B6 26, 68159 Mannheim, Germany
E-mail: heiko@informatik.uni-mannheim.de

Abstract. In the recent years, different Web knowledge graphs, both free and commercial, have been created. While Google coined the term "Knowledge Graph" in 2012, there are also a few openly available knowledge graphs, with DBpedia, YAGO, and Freebase being among the most prominent ones. Those graphs are often constructed from semi-structured knowledge, such as Wikipedia, or harvested from the web with a combination of statistical and linguistic methods. The result are large-scale knowledge graphs that try to make a good trade-off between completeness and correctness. In order to further increase the utility of such knowledge graphs, various refinement methods have been proposed, which try to infer and add missing knowledge to the graph, or identify erroneous pieces of information. In this article, we provide a survey of such *knowledge graph refinement* approaches, with a dual look at both the methods being proposed as well as the evaluation methodologies used.

Keywords: Knowledge Graphs, Refinement, Completion, Correction, Error Detection, Evaluation

1. Introduction

Knowledge graphs on the Web are a backbone of many information systems that require access to structured knowledge, be it domain-specific or domain-independent. The idea of feeding intelligent systems and agents with general, formalized knowledge of the world dates back to classic Artificial Intelligence research in the 1980s [91]. More recently, with the advent of Linked Open Data [5] sources like DBpedia [56], and by Google's announcement of the Google Knowledge Graph in 2012[1], representations of general world knowledge as graphs have drawn a lot of attention again.

There are various ways of building such knowledge graphs. They can be curated like *Cyc* [57], edited by the crowd like *Freebase* [9] and *Wikidata* [104], or extracted from large-scale, semi-structured web knowledge bases such as Wikipedia, like *DBpedia* [56] and *YAGO* [101]. Furthermore, information extraction methods for unstructured or semi-structured information are proposed, which lead to knowledge graphs like *NELL* [14], *PROSPERA* [70], or *KnowledgeVault* [21].

Whichever approach is taken for constructing a knowledge graph, the result will never be perfect [10]. As a model of the real world or a part thereof, formalized knowledge cannot reasonably reach *full coverage*, i.e., contain information about each and every entity in the universe. Furthermore, it is unlikely, in particular when heuristic methods are applied, that the knowledge graph is *fully correct* – there is usually a trade-off between coverage and correctness, which is addressed differently in each knowledge graph. [111]

To address those shortcomings, various methods for *knowledge graph refinement* have been proposed.

[1] `http://googleblog.blogspot.co.uk/2012/05/introducing-knowledge-graph-things-not.html`

In many cases, those methods are developed by researchers outside the organizations or communities which *create* the knowledge graphs. They rather take an existing knowledge graph and try to increase its coverage and/or correctness by various means. Since such works are reviewed in this survey, the focus of this survey is not knowledge graph *construction*, but knowledge graph *refinement*.

For this survey, we view knowledge graph *construction* as a construction from scratch, i.e., using a set of operations on one or more sources to create a knowledge graph. In contrast, knowledge graph *refinement* assumes that there is already a knowledge graph given which is improved, e.g., by adding missing knowledge or identifying and removing errors. Usually, those methods directly use the information given in a knowledge graph, e.g., as training information for automatic approaches. Thus, the methods for both *construction* and *refinement* may be similar, but not the same, since the latter work on a given graph, while the former are not.

It is important to note that for many knowledge graphs, one or more refinement steps are applied when creating and/or before publishing the graph. For example, logical reasoning is applied on some knowledge graphs for validating the consistency of statements in the graph, and removing the inconsistent statements. Such post processing operations (i.e., operations applied after the initial construction of the graph) would be considered as *refinement* methods for this survey, and are included in the survey.

Decoupling knowledge base construction and refinement has different advantages. First, it allows – at least in principle – for developing methods for refining arbitrary knowledge graphs, which can then be applied to improve multiple knowledge graphs.[2] Other than fine-tuning the heuristics that create a knowledge graph, the impact of such generic refinement methods can thus be larger. Second, evaluating refinement methods in isolation of the knowledge graph construction step allows for a better understanding and a cleaner separation of effects, i.e., it facilitates more qualified statements about the effectiveness of a proposed approach.

The rest of this article is structured as follows. Section 2 gives a brief introduction into knowledge graphs in the Semantic Web. In section 3 and 4, we present a categorization of approaches and evaluation methodologies. In section 5 and 6, we present the review of methods for completion (i.e., increasing coverage) and error detection (i.e., increasing correctness) of knowledge graphs. We conclude with a critical reflection of the findings in section 7, and a summary in section 8.

2. Knowledge Graphs in the Semantic Web

From the early days, the Semantic Web has promoted a graph-based representation of knowledge, e.g., by pushing the RDF standard[3]. In such a graph-based knowledge representation, *entities*, which are the nodes of the graph, are connected by *relations*, which are the edges of the graph (e.g., *Shakespeare has written Hamlet*), and entities can have types, denoted by *is a* relations (e.g., *Shakespeare is a writer*, *Hamlet is a play*). In many cases, the sets of possible types and relations are organized in a *schema* or *ontology*, which defines their interrelations and restrictions of their usage.

With the advent of Linked Data [5], it was proposed to interlink different datasets in the Semantic Web. By means of interlinking, the collection of could be understood as one large, global knowledge graph (although very heterogenous in nature). To date, roughly 1,000 datasets are interlinked in the *Linked Open Data cloud*, with the majority of links connecting *identical* entities in two datasets [95].

The term *Knowledge Graph* was coined by Google in 2012, referring to their use of semantic knowledge in Web Search ("Things, not strings"), and is recently also used to refer to Semantic Web knowledge bases such as DBpedia or YAGO. From a broader perspective, any graph-based representation of some knowledge could be considered a *knowledge graph* (this would include any kind of RDF dataset, as well as description logic ontologies). However, there is no common definition about what a knowledge graph is and what it is not. Instead of attempting a formal definition of what a knowledge graph is, we restrict ourselves to a minimum set of characteristics of knowledge graphs, which we use to tell knowledge graphs from other collections of knowledge which we would *not* consider as knowledge graphs. A knowledge graph

1. mainly describes real world entities and their interrelations, organized in a graph.

[2]See section 7.2 for a critical discussion.

[3]`http://www.w3.org/RDF/`

2. defines possible classes and relations of entities in a schema.
3. allows for potentially interrelating arbitrary entities with each other.
4. covers various topical domains.

The first two criteria clearly define the focus of a knowledge graph to be the actual *instances* (A-box in description logic terminology), with the schema (T-box) playing only a minor role. Typically, this means that the number of instance-level statements is by several orders of magnitude larger than that of schema level statements (cf. Table 1). In contrast, the schema can remain rather shallow, at a small degree of formalization. In that sense, mere ontologies without any instances (such as *DOLCE* [27]) would not be considered as knowledge graphs. Likewise, we do not consider *WordNet* [67] as a knowledge graph, since it is mainly concerned with common nouns and words[4] and their relations (although a few proper nouns, i.e., instances are also included).[5]

The third criterion introduces the possibility to define arbitrary relations between instances, which are not restricted in their domain and/or range. This is a property which is hardly found in relational databases, which follow a strict schema.

Furthermore, knowledge graphs are supposed to cover at least a major portion of the domains that exist in the real world, and are not supposed to be restricted to only one domain (such as geographic entities). In that sense, large, but single-domain datasets, such as *GeoNames*[6], would not be considered a knowledge graph.

Knowledge graphs on the Semantic Web are typically provided using Linked Data [5] as a standard. They can be built using different methods: they can be curated by an organization or a small, closed group of people, crowd-sourced by a large, open group of individuals, or created with heuristic, automatic or semi-automatic means. In the following, we give an overview of existing knowledge graphs, both open and company-owned.

[4]The question of whether words as such are real world entities or not is of philosophical nature and not answered within the scope of this article.

[5]Nevertheless, it is occasionally used for evaluating knowledge graph refinement methods, as we will show in the subsequent sections.

[6]`http://www.geonames.org/`

2.1. Cyc and OpenCyc

The *Cyc* knowledge graph is one of the oldest knowledge graphs, dating back to the 1980s [57]. Rooted in traditional artificial intelligence research, it is a *curated* knowledge graph, developed and maintained by CyCorp Inc.[7] OpenCyc is a reduced version of Cyc, which is publicly available. A Semantic Web endpoint to OpenCyc also exists, containing links to DBpedia and other LOD datasets.

OpenCyc contains roughly 120,000 instances and 2.5 million facts defined for those instances; its schema comprises a type hierarchy of roughly 45,000 types, and 19,000 possible relations.[8]

2.2. Freebase

Curating a universal knowledge graph is an endeavour which is infeasible for most individuals and organizations. To date, more than 900 person years have been invested in the creation of Cyc [92], with gaps still existing. Thus, distributing that effort on as many shoulders as possible through *crowdsourcing* is a way taken by *Freebase*, a public, editable knowledge graph with schema templates for most kinds of possible entities (i.e., persons, cities, movies, etc.). After MetaWeb, the company running Freebase, was acquired by Google, Freebase was shut down on March 31st, 2015.

The last version of Freebase contains roughly 50 million entities and 3 billion facts[9]. Freebase's schema comprises roughly 27,000 entity types and 38,000 relation types.[10]

2.3. Wikidata

Like Freebase, *Wikidata* is a collaboratively edited knowledge graph, operated by the Wikimedia foundation[11] that also hosts the various language editions of Wikipedia. After the shutdown of Freebase, the data contained in Freebase is subsequently moved to Wikidata.[12] A particularity of Wikidata is that for each ax-

[7]`http://www.cyc.com/`

[8]These numbers have been gathered by own inspections of the 2012 of version of OpenCyc, available from `http://sw.opencyc.org/`

[9]`http://www.freebase.com`

[10]These numbers have been gathered by queries against Freebase's query endpoint.

[11]`http://wikimediafoundation.org/`

[12]`http://plus.google.com/109936836907132434202/posts/3aYFVNf92A1`

iom, provenance metadata can be included – such as the source and date for the population figure of a city [104].

To date, Wikidata contains roughly 16 million instances[13] and 66 million statements[14]. Its schema defines roughly 23,000 types[15] and 1,600 relations[16].

2.4. DBpedia

DBpedia is a knowledge graph which is extracted from structured data in Wikipedia. The main source for this extraction are the key-value pairs in the Wikipedia infoboxes. In a crowd-sourced process, types of infoboxes are mapped to the DBpedia ontology, and keys used in those infoboxes are mapped to properties in that ontology. Based on those mappings, a knowledge graph can be extracted [56].

The most recent version of the main DBpedia (i.e., DBpedia 2015-04, extracted from the English Wikipedia based on dumps from February/March 2015) contains 4.8 million entities and 176 million statements about those entities.[17] The ontology comprises 735 classes and 2,800 relations.[18]

2.5. YAGO

Like DBpedia, YAGO is also extracted from DBpedia. YAGO builds its classification implicitly from the category system in Wikipedia and the lexical resource *WordNet* [67], with infobox properties manually mapped to a fixed set of attributes. While DBpedia creates different interlinked knowledge graphs for each language edition of Wikipedia [12], YAGO aims at an automatic fusion of knowledge extracted from various Wikipedia language editions, using different heuristics [65].

The latest release of YAGO, i.e., YAGO3, contains 4.6 million entities and 26 million facts about those types. The schema comprises rouhgly 488,000 types and 77 relations [65].

[13] http://www.wikidata.org/wiki/Wikidata:Statistics
[14] http://tools.wmflabs.org/wikidata-todo/stats.php
[15] http://tools.wmflabs.org/wikidata-exports/miga/?classes#_cat=Classes
[16] http://www.wikidata.org/wiki/Special:ListProperties
[17] http://dbpedia.org/services-resources/datasets/dataset-2015-04/dataset-2015-04-statistics
[18] http://dbpedia.org/dbpedia-data-set-2015-04

2.6. NELL

While DBpedia and YAGO use semi-structured content as a base, methods for extracting knowledge graphs from unstructured data have been proposed as well. One of the earliest approaches working at web-scale was the *Never Ending Language Learning (NELL)* project [14]. The project works on a large-scale corpus of web sites and exploits a coupled process which learns text patterns corresponding type and relation assertions, as well as applies them to extract new entities and relations. Reasoning is applied for consistency checking and removing inconsistent axioms. The system is still running today, continuously extending its knowledge base. While not published using Semantic Web standards, it has been shown that the data in NELL can be transformed to RDF and provided as Linked Open Data as well [113].

In its most recent version (i.e., the 945th iteration), NELL contains roughly 2 million entities and 433,000 relations between those. The NELL ontology defines 285 classes and 425 relations.[19]

2.7. Google's Knowledge Graph

Google's Knowledge Graph was introduced to the public in 2012, which was also when the term *knowledge graph* as such was coined. Google itself is rather secretive about how their Knowledge Graph is constructed; there are only a few external sources that discuss some of the mechanisms of information flow into the Knowledge Graph based on experience[20]. From those, it can be assumed that major semi-structured web sources, such as Wikipedia, contribute to the knowledge graph, as well as structured markup (like schema.org Microdata [66]) on web pages and contents from Google's online social network Google+.

According to [21], Google's Knowledge Graph contains 18 billion statements about 570 million entities, with a schema of 1,500 entity types and 35,000 relation types.

[19] These numbers have been derived from the promotion heatmap at http://rtw.ml.cmu.edu/resources/results/08m/NELL.08m.945.heatmap.html.
[20] E.g., http://www.techwyse.com/blog/search-engine-optimization/seo-efforts-to-get-listed-in-google-knowledge-graph/

2.8. Google's Knowledge Vault

The *Knowledge Vault* is another project by Google. It extracts knowledge from different sources, such as text documents, HTML tables, and structured annotations on the Web with Microdata or MicroFormats. Extracted facts are combined using both the extractor's confidence values, as well as prior probabilities for the statements, which are computed using the Freebase knowledge graph (see above). From those components, a confidence value for each fact is computed, and only the confident facts are taken into Knowledge Vault [21].

According to [21], the Knowledge Vault contains roughly 45 million entities and 271 million fact statements, using 1,100 entity types and 4,500 relation types.

2.9. Yahoo!'s Knowledge Graph

Like Google, Yahoo! also has their internal knowledge graph, which is used to improve search results. The knowledge graph builds on both public data (e.g., Wikipedia and Freebase), as well as closed commercial sources for various domains. It uses wrappers for different sources and monitors evolving sources, such as Wikipedia, for constant updates.

Yahoo's knowledge graph contains roughly 3.5 million entities and 1.4 billion relations. Its schema, which is aligned with schema.org, comprises 250 types of entities and 800 types of relations. [6]

2.10. Microsoft's Satori

Satori is Microsoft's equivalent to Google's Knowledge Graph.[21] Although almost no public information on the construction, the schema, or the data volume of Satori is available, it has been said to consist of 300 million entities and 800 million relations in 2012, and its data representation format to be RDF.[22]

2.11. Facebook's Entities Graph

Although the majority of the data in the online social network *Facebook*[23] is perceived as connections between people, Facebook also works on extracting a knowledge graph which contains a larger variety of entities. The information people provide as personal information (e.g., their home town, the schools they went to), as well as their likes (movies, bands, books, etc.), often represent entities, which can be linked both to people as well as among each other. By parsing textual information and linking to Wikipedia, the graph also contains links among entities, e.g., the writer of a book. Although not many public numbers about Facebook's Entities Graph exist, it is said to contain more than 100 billion connections between entities.[24]

2.12. Summary

Table 1 summarizes the characteristics of the knowledge graphs discussed above. It can be observed that the graphs differ in the basic measures, such as the number of entities and relations, as well as in the size of the schema they use, i.e., the number of classes and relations. From these differences, it can be concluded that the knowledge graphs must differ in other characteristics as well, such as average node degree, density, or connectivity.

3. Categorization of Knowledge Graph Refinement Approaches

Knowledge graph refinement methods can differ along different dimensions. For this survey, we distinguish the overall goal of the method, i.e., completion vs. correction of the knowledge graph, the refinement target (e.g., entity types, relations between entities, or literal values), as well as the data used by the approach (i.e., only the knowledge graph itself, or further external sources). All three dimensions are orthogonal.

There are a few research fields which are related to knowledge graph refinement: *Ontology learning* mainly deals with learning a concept level description of a domain, such as a hierarchy (e.g., *Cities* are *Places*) [13,64]. Likewise, *description logic learning* is mainly concerned with refining such concept level descriptions [55]. As stated above, the focus of knowledge graphs, in contrast, is rather the instance (A-box) level, not so much the concept (T-box) level. Following that notion, we only consider those works as *knowl-*

[21] `http://blogs.bing.com/search/2013/03/21/understand-your-world-with-bing/`

[22] `http://research.microsoft.com/en-us/projects/trinity/query.aspx`, currently offline, accessible through the Internet Archive.

[23] `http://www.facebook.com/`

[24] `http://www.facebook.com/notes/facebook-engineering/under-the-hood-the-entities-graph/10151490531588920`

Table 1

Overview of popular knowledge graphs. The table depicts the number of instances and facts; as well as the number of different types and relations defined in their schema. *Instances* denotes the number of instances or A-box concepts defined in the graph, *Facts* denotes the number of statements about those instances, *Entity types* denotes the number of different types or classes defined in the schema, and *Relation types* denotes the number of different relations defined in the schema. Microsoft's Satori and Facebook's Entities Graph are not shown, because to the best of our knowledge, no detailed recent numbers on the graph are publicly available.

Name	Instances	Facts	Types	Relations
DBpedia (English)	4,806,150	176,043,129	735	2,813
YAGO	4,595,906	25,946,870	488,469	77
Freebase	49,947,845	3,041,722,635	26,507	37,781
Wikidata	15,602,060	65,993,797	23,157	1,673
NELL	2,006,896	432,845	285	425
OpenCyc	118,499	2,413,894	45,153	18,526
Google's Knowledge Graph	570,000,000	18,000,000,000	1,500	35,000
Google's Knowledge Vault	45,000,000	271,000,000	1,100	4,469
Yahoo! Knowledge Graph	3,443,743	1,391,054,990	250	800

edge graph refinement approaches which focus on refining the A-box. Approaches that only focus on the refining T-box are not considered for this survey, however, if the schema or ontology is refined as a means to ultimately improve the A-box, those works are included in the survey.

3.1. Completion vs. Error Detection

There are two main goals of knowledge graph refinement: (a) adding missing knowledge to the graph, i.e., *completion*, and (b) identifying wrong information in the graph, i.e., *error detection*. From a data quality perspective, those goals relate to the data quality dimensions *free-of-error* and *completeness* [86].

3.2. Target of Refinement

Both completion and error detection approaches can be further distinguished by the targeted kind of information in the knowledge graph. For example, some approaches are targeted towards completing/correcting entity type information, while others are targeted to (either specific or any) relations between entities, or interlinks between different knowledge graphs, or literal values, such as numbers. While the latter can be of any datatype (strings, numbers, dates, etc.), most research focuses on numerical or date-valued literal values.

Another strand of research targets the extension of the *schema* used by the knowledge graph (i.e., the T-box), not the data (the A-box). However, as discussed above, approaches focusing purely on the schema without an impact on the instance level are not considered for this survey.

3.3. Internal vs. External Methods

A third distinguishing property is the data used by an approach. While *internal* approaches only use the knowledge graph itself as input, *external* methods use additional data, such as text corpora. In the widest sense, approaches making use of human knowledge, such as crowdsourcing [1] or games with a purpose [105], can also be viewed as external methods, although not fully automatic ones.

4. Categorization of Evaluation Methods

There are different possible ways to evaluate knowledge graph refinement. On a high level, we can distinguish methodologies that use only the knowledge graph at hand, and methodologies that use external knowledge, such as annotations provided by humans.

4.1. Partial Gold Standard

One common evaluation strategy is to use a partial gold standard. In this methodology, a subset of graph entities or relations are selected and labeled manually. Other evaluations use external knowledge graphs and/or databases as partial gold standards.

For completion tasks, this means that all axioms that *should exist in the knowledge graph* are collected, whereas for correction tasks, a set of axioms in the graph is manually labeled as correct or incorrect. The quality of completion approaches is usually measured in recall, precision, and F-measure, whereas for correction methods, accuracy and/or area under the ROC curve (AUC) are often used alternatively or in addition.

Sourcing partial gold standards from humans can lead to high quality data (given that the knowledge graph and the ontology it uses are not overly complex), but is costly, so that those gold standards are usually small. Exploiting other knowledge graphs based on knowledge graph interlinks (e.g., using Freebase data as a gold standard to evaluate DBpedia) is sometimes proposed to yield larger-scale gold standards, but has two sources of errors: errors in the target knowledge graph, and errors in the linkage between the two. For example, it has been reported that 20% of the interlinks between DBpedia and Freebase are incorrect [110], and that roughly half of the `owl:sameAs` links between knowledge graphs connect two things which are related, but not *exactly* the same (such as the company *Starbucks* and a particular Starbucks coffee shop) [33].

4.2. Knowledge Graph as Silver Standard

Another evaluation strategy is to use the given knowledge graph itself as a test dataset. Since the knowledge graph is not perfect (otherwise, refinement would not be necessary), it cannot be considered as a *gold standard*. However, assuming that the given knowledge graph is already of reasonable quality, we call this method *silver standard* evaluation, as already proposed in other works [32,45,74].

The silver standard method is usually applied to measure the performance of knowledge graph *completion* approaches, where it is analyzed how well relations in a knowledge graph can replicated by a knowledge graph completion method. As for gold standard evaluations, the result quality is usually measured in recall, precision, and F-measure. In contrast to using human annotations, large-scale evaluations are easily possible. The silver standard method is only suitable for evaluating knowledge graph completion, not for error detection, since it assumes the knowledge graph to be correct.

There are two variants of silver standard evaluations: in the more common ones, the entire knowledge graph is taken as input to the approach at hand, and the evaluation is then also carried out on the entire knowledge graph. As this may lead to an overfitting effect (in particular for internal methods), some works also foresee the splitting of the graph into a training and a test partition, which, however, is not as straight forward as, e.g., for propositional classification tasks [72], which is why most papers use the former method. Furthermore, split and cross validation do not fully solve the overfitting effect. For example, if a knowledge graph, by construction, has a bias towards certain kinds of information (e.g., relations are more complete for some classes than for others), approaches overadapting to that bias will be rated better than those which do not (and which may actually perform better in the general case).

A problem with this approach is that the knowledge graph itself is not perfect (otherwise, it would not need refinement), thus, this evaluation method may sometimes underrate the evaluated approach. More precisely, most knowledge graphs follow the *open world assumption*, i.e., an axiom not present in the knowledge graph may or may not hold. Thus, if a completion approach correctly predicts the existence of an axiom missing in the knowledge graph, this would count as a false positive and thus lower precision. Approaches overfitting to the coverage bias of a knowledge graph at hand may thus be overrated.

4.3. Retrospective Evaluation

For retrospective evaluations, the output of a given approach is given to human judges for annotation, who then label suggested completions or identified errors as correct and incorrect. The quality metric is usually accuracy or precision, along with a statement about the total number of completions or errors found with the approach, and ideally also with a statement about the agreement of the human judges.

In many cases, automatic refinement methods lead to a very large number of findings, e.g., lists of tens of thousands of axioms which are potentially erroneous. Thus, retrospective evaluations are often carried out only on samples of the results. For some approaches which produce higher level artifacts – such as error patterns or completion rules – as intermediate results, a feasible alternative is to evaluate those artifacts instead of the actually affected axioms.

While partial gold standards can be reused for comparing different methods, this is not the case for retrospective evaluations. On the other hand, retrospective evaluations may make sense in cases where the interesting class is rare. For example, when evaluating error detection methods, a sample for a partial gold standard from a high-quality graph is likely not to contain a meaningful number of errors. In those cases, retrospective evaluation methodologies are often preferred over partial gold standards.

Another advantage of retrospective evaluations is that they allow a very detailed analysis of an approach's results. In particular, inspecting the errors made by an approach often reveals valuable findings about the advantages and limitations of a particular approach.

Table 2 sums up the different evaluation methodologies and contrasts their advantages and disadvantages.

4.4. Computational Performance

In addition to the performance w.r.t. correctness and/or completeness of results, computational performance considerations become more important as knowledge graphs become larger. Typical performance measures for this aspect are runtime measurements, as well as memory consumption.

Besides explicit measurement of computational performance, a "soft" indicator for computational performance is whether an approach has been evaluated (or at least the results have been materialized) on an entire large-scale knowledge graph, or only on a subgraph. The latter is often done when applying evaluations on a partial gold standard, where the respective approach is only executed on entities contained in that partial gold standard.

5. Approaches for Completion of Knowledge Graphs

Completion of knowledge graphs aims at increasing the coverage of a knowledge graph. Depending on the target information, methods for knowledge graph completion either predict missing entities, missing types for entities, and/or missing relations that hold between entities.

In this section, we survey methods for knowledge graph completion. We distinguish internal and external methods, and further group the approaches by the completion target.

5.1. Internal Methods

Internal methods use only the knowledge contained in the knowledge graph itself to predict missing information.

5.1.1. Methods for Completing Type Assertions

Predicting a type or class for an entity given some characteristics of the entity is a very common problem in machine learning, known as *classification*. The classification problem is *supervised*, i.e., it learns a classification model based on labeled training data, typically the set of entities in a knowledge graph (or a subset thereof) which have types attached. In machine learning, *binary* and *multi-class* prediction problems are distinguished. In the context of knowledge graphs, in particular the latter are interesting, since most knowledge graphs contain entities of more than two different types. Depending on the graph at hand, it might be worthwile distinguishing *multi-label* classification, which allows for assigning more than one class to an instance (e.g., *Arnold Schwarzenegger* being both an *Actor* and a *Politician*), and single-label classification, which only assigns one class to an instance [103].

For internal methods, the features used for classification are usually the relations which connect an entity to other entities [81,88], i.e., they are a variant of *link-based classification* problems [31]. For example, an entity which has a *director* relation is likely to be a *Movie*.

In [79,80], we propose a probabilistic method, which is based on conditional probabilities, e.g., the probability of a node being of type *Actor* is high if there are ingoing edges of type *cast*. Such probabilities are exploited by the *SDType* algorithm, which is currently deployed for DBpedia and adds around 3.4 million additional type statements to the knowledge graph.

In [98], the use of Support Vector Machines (SVMs) has been proposed to type entities in DBpedia and Freebase. The authors also exploit interlinks between the knowledge graphs and classify instances in one knowledge graph based on properties present in the other, in order to increase coverage and precision. Nickel et al. [73] propose the use of *matrix factorization* to predict entity types in YAGO.

Since many knowledge graphs come with a class hierarchy, e.g., defined in a formal ontology, the type prediction problem could also be understood as a *hierarchical classification* problem. Despite a larger body of work existing on methods for hierarchical classifi-

Table 2
Overview on evaluation methods with their advantages and disadvantages

Methodology	Advantages	Disadvantages
Partial Gold Standard	highly reliable results reusable	costly to produce balancing problems
Knowledge Graph as Silver Standard	large-scale evaluation feasible subjectiveness is minimized	less reliable results prone to overfitting
Retrospective Evaluation	applicable to disbalanced problems allows for more detailed analysis of approaches	not reusable approaches cannot be compared directly

cation [96], there are, to the best of our knowledge, no applications of those methods to knowledge graph completion.

In data mining, association rule mining [38] is a method that analyzes the co-occurence of items in itemsets and derives association rules from those co-occurences. For predicting missing information in knowledge graphs, those methods can be exploited, e.g., in the presence of redundant information. For example, in DBpedia, different type systems (i.e., the DBpedia ontology and YAGO, among others) are used in parallel, which are populated with different methods (Wikipedia infoboxes and categories, respectively). This ensures both enough overlap to learn suitable association rules, as well as a number of entities that only have a type in one of the systems, to which the rules can be applied. In [77], we exploit such association rules to predict missing types in DBpedia based on such redundancies.

In [99], the use of topic modeling for type prediction is proposed. Entities in a knowledge graph are represented as documents, on which Latent Dirichlet Allocation (LDA) [7] is applied for finding topics. By analyzing the co-occurrence of topics and entity types, new types can be assigned to entities based on the topics detected for those entities.

5.1.2. Methods for Predicting Relations

While primarily used for adding missing type assertions, classification methods can also be used to predict the existence of relations. To that end, Socher et al. [100] propose to train a tensor neural network to predict relations based on chains of other relations, e.g., if a person is born in a city in *Germany*, then the approach can predict (with a high probability) that the nationality of that person is *German*. The approach is applied to Freebase and WordNet. A similar approach is presented in [50], where the authors show that refining such a problem with schema knowledge – either defined or induced – can significantly improve the performance of link prediction. In [49], an approach similar to association rule mining is used to find meaningful chains of relations for relation prediction. Similarly, in [112], an embedding of pairwise entity relations into a lower dimensional space is learned, which is then used to predict the existence of relations in Freebase.

Likewise, association rule mining can be used for predicting relations as well. In [46], the mining of association rules which predict relations between entities in DBpedia from Wikipedia categories is proposed.[25]

5.2. External Methods

External methods use sources of knowledge – such as text corpora or other knowledge graphs – which are not part of the knowledge graph itself. Those external sources can be linked from the knowledge graph, such as knowledge graph interlinks or links to web pages, e.g., Wikipedia pages describing an entity, or exist without any relation to the knowledge graph at hand, such as large text corpora.

5.2.1. Methods for Completing Type Assertions

For type prediction, there are also classification methods that use external data. In contrast to the internal classification methods described above, external data is used to create a feature representation of an entity.

Nuzzolese et al. [75] propose the usage of the Wikipedia link graph to predict types in a knowledge graph using a k-nearest neighbors classifier. Given that a knowledge graph contains links to Wikipedia, interlinks between Wikipedia pages are exploited to create feature vectors, e.g., based on the categories of the related pages. Since links between Wikipedia pages are not constrained, there are typically more interlinks between Wikipedia pages than between the corresponding entities in the knowledge graph.

[25] Note that since Wikipedia categories are part of the DBpedia knowledge graph, we consider this approach an internal one.

Apriosio et al. [3] use types of entities in different DBpedia language editions (each of which can be understood as a knowledge graph connected to the others) as features for predicting missing types. The authors use a k-NN classifier with different distance measures (i.e., kernel functions), such as the overlap of two articles' categories. In their setting, a combination of different distance measures is reported to provide the best results.

Another set of approaches uses abstracts in DBpedia to extract definitionary clauses, e.g., using Hearst patterns [36]. Such approaches have been proposed by Gangemi et al. [28] and Kliegr [47], where the latter uses abstracts in the different languages in order to increase coverage and precision.

5.2.2. Methods for Predicting Relations

Like types, relations to other entities can also be predicted from textual sources, such as Wikipedia pages. Lange et al. [52] learn patterns on Wikipedia abstracts using Conditional Random Fields [51]. A similar approach, but on entire Wikipedia articles, is proposed by [109].[26]

Another common method for the prediction of a relation between two entities is *distant supervision*. Typically, such approaches use large text corpora. As a first step, entites in the knowledge graph are linked to the text corpus by means of Named Entity Recognition [40,90]. Then, based on the relations in the knowledge graph, those approaches seek for text pattern which correspond to relation types (such as: *Y's book X* being a pattern for the relation *author* holding between *X* and *Y*), and apply those patterns to find additional relations in the text corpus. Such methods have been proposed by Mintz et al. [68] for Freebase, and by Aprosio et al. [4] for DBpedia. In both cases, Wikipedia is used as a text corpus. In [30], a similar setting with DBpedia and two text corpora – the English Wikipedia and an English-language news corpus – is used, the latter showing less reliable results. A similar approach is followed in the *RdfLiveNews* prototype, where RSS feeds of news companies are used to address the aspect of timeliness in DBpedia, i.e., extracting new information that is either outdated or missing in DBpedia [29].

West et al. [107] propose the use of web search engines to fill gaps in knowledge graphs. Like in the works discussed above, they first discover lexicalizations for relations. Then, they use those lexicalizations to formulate search engine queries for filling missing relation values. Thus, they use the whole Web as a corpus, and combine information retrieval and extraction for knowledge graph completion.

While text is unstructured, some approaches have been proposed that use *semi-structured* data for completing knowledge graphs. In particular, approaches leveraging on structured data in Wikipedia are found in the literature. Those are most often used together with DBpedia, so that there are already links between the entities and the corpus of background knowledge, i.e., no Named Entity Recognition has to be performed, in contrast to the distant supervision approaches discussed above.

Muñoz et al. [69] propose extraction from tables in Wikipedia. They argue that for two entities co-occurring in a Wikipedia table, it is likely that the corresponding entities should share an edge in the knowledge graph. To fill in those edges, they first extract a set of candidates from the tables, using all possible relations that hold between at least one pair of entities in two columns. Then, based on a labeled subset of that extraction, they apply classification using various features to identify those relations that should *actually* hold in the knowledge graph.

Ritze et al. [89] extend this approach to arbitrary HTML tables. This requires that not only that pairs of table columns have to be matched to properties in the DBpedia ontology, but also that rows in the table need to be matched to entities in DBpedia. The authors propose an iterative approach to solve those two problems. The approach is evaluated on a gold standard mapping for a sample of HTML tables from the WebDataCommons Web Table corpus[27]. Since such tables can also contain literal values (such as population figures), the approach is capable of completing both relations between entities, and literal values for entities.

In [84], we have proposed the use of list pages in Wikipedia for generating both type and relation assertions in knowledge graphs, based on statistical methods. The idea is that entities appear together in list pages for a reason, and it should be possible to identify that common pattern appearing for the majority of the instance in the list page. For example, instances linked from the page *List of Jewish-American Writers* should all be typed as *Writer* and include an edge *religion* to *Jewish*, as well as an edge *nationality* to *United States*

[26] Although both approaches do not explicitly mention DBpedia, but aim at completing missing key-value pairs in infoboxes, this can be directly transferred to extending DBpedia.

[27] `http://webdatacommons.org/webtables/`

of America. Once such patterns are found for the majority of the list items, they can be applied to the remaining ones to fill gaps in the knowledge graph.

Many knowledge graphs contain links to other knowledge graphs. Those are often created automatically [71]. Interlinks between knowledge graphs can be used to fill gaps in one knowledge graph from information defined in another knowledge graph. If a mapping both on the instance and on the schema level is known, it can be exploited for filling gaps in knowledge graphs on both sides.

One work in this direction is presented by Bryl and Bizer [12], where different language versions of DBpedia (each of which can be seen as a knowledge graph of its own) are used to fill missing values in the English language DBpedia (the one which is usually meant when referring to *DBpedia*).

Dutta et al. [23] propose a probabilistic mapping between knowledge graphs. Based on distributions of types and properties, they create a mapping between knowledge graphs, which can then be used to derive additional, missing facts in the knowledge graphs. To that end, the type systems used by two knowledge graphs are mapped to one another. Then, types holding in one knowledge graph can be used to predict those that should hold in another.

6. Approaches for Error Detection in Knowledge Graphs

Like completion methods discussed in the previous section, methods for identifying errors in knowledge graphs can target various types of information, i.e., type assertions, relations between individuals, literal values, and knowledge graph interlinks.

In this section, we survey methods for detecting errors in knowledge graphs. Like for the previous section, we distinguish internal and external methods, and further group the approaches by the error detection target.

6.1. Internal Methods

Internal methods use only the information given in a knowledge graph to find out whether an axiom in the knowledge graph is plausible or not.

6.1.1. Methods for Finding Erroneous Type Assertions

In contrast to relation assertions, type assertions are most often more correct in knowledge graphs than relation assertions [80]. Hence, methods for finding erroneous type assertions are rather rare. One such method is proposed by Ma et al. [63], who use inductive logic programming for learning disjointness axioms, and then apply those disjointness axioms for identifying potentially wrong type assertions.

6.1.2. Methods for Finding Erroneous Relations

For building Knowledge Vault, Dong et al. use classification to tell relations which should hold in a knowledge graph from those which should not [21]. Like the work by Muñoz et al. discussed above, each relation is used as an instance in the classification problem, with the existence of the relation in the knowledge graph being used as a binary class. This classification is used as a cleansing step after the knowledge extraction process. While the creation of *positive* training examples from the knowledge graph is quite straight forward, the authors propose the creation of *negative* training examples by applying a *Local Closed World Assumption*, assuming that a relation r between two entities e_1 and e_2 does not hold if it is not present in the knowledge graph, and there is a relation r between e_1 and another e_3.

In [80], we have proposed a statistical method for finding wrong statements within a knowledge graph. For each type of relation, we compute the characteristic distribution of subject and object types for edge, i.e., each instantiation of the relation. Edges in the graph whose subject and object type strongly deviate from the characteristic distributions are then identified as potential errors.

Reasoning is a field of study in the artificial intelligence community which deals with automatically deriving proofs for theorems, and for uncovering contradictions in a set of axioms [91]. The techniques developed in this field have been widely adopted in the Semantic Web community, leading to the development of a larger number of ontology reasoners [19,20,62].

For exploiting reasoning for error checking in knowledge graphs, a rich ontology is required, which defines the possible types of nodes and edges in a knowledge graph, as well as the restrictions that hold on them. For example, if a person is defined to be the capital of a state, this is a contradiction, since capitals are cities, and cities and persons are disjoint, i.e., no entity can be a city and a person at the same time. Reasoning is often used at the building stage of a knowledge graph, i.e., when new axioms are about to be added. For example, NELL and PROSPERA perform reasoning at that point to determine whether the new axiom is plausible or not, and discard implausible ones [14,70]. For real-world knowledge graphs, reasoning can be difficult due to the presence of errors and noise in the data [43,87].

Works using reasoning as a refinement operation for knowledge graphs have also been proposed. However, many knowledge graphs, such as DBpedia, come with ontologies that are not rich enough to perform reasoning for inconsistency detection – for example, they lack class disjointness assertions needed for an inference as in the example above. Therefore, approaches exploiting reasoning are typically used in conjunction with methods for enriching ontologies, such as statistical methods, as proposed in [42] and [102], or association rule mining, as in [53]. In all of those works, the ontology at hand is enriched with further axioms, which can then be used for detecting inconsistencies. For example, if a reasoner concludes that an entity should both be a person and an organization, and from the enrichment steps has a disjointness axiom between the two types added, a reasoner can state that one out of a few axioms in the knowledge graph has to be wrong. Another source of additional disjointness axioms is the use of a top level ontology like *DOLCE* [27], which provides high-level disjointness axioms [82].

In [85], a light-weight reasoning approach is proposed to compare actual and defined domains and ranges of relations in a knowledge graph schema. The authors propose a set of heuristics for fixing the schema if the actual and the defined domain or range strongly deviate.

6.1.3. Methods for Finding Erroneous Literal Values

Outlier detection or *anomaly detection* methods aim at identifying those instances in a dataset that deviate from the majority from the data, i.e., that follow different characteristics than the rest of the data [15,39].

As outlier detection in most cases deals with *numeric* data, numeric literals are a natural target for those methods. In [108], we have proposed the application of different univariate outlier detection methods (such as interquartile range or kernel density estimation) to DBpedia. Although outlier detection does not necessarily identify errors, but also natural outliers (such as the population of very large cities), it has been shown that the vast majority of outliers identified are actual errors in DBpedia, mostly resulting from mistakes made when parsing strings using various number formats and units of measurement.

To lower the influence of natural outliers, an extension of that approach has been presented in [24], where the instance set under inspection is first split into smaller subsets. For example, population values are inspected for countries, cities, and towns in isolation, thus, the distributions are more homogenous, which leads to a higher precision in error identification. Furthermore, the approach foresees *cross-checking* the outliers that have been found, using other knowledge graphs in order to further reduce the influence of natural outliers, which makes it a mixed approach with both an internal and an external component.

6.1.4. Methods for Finding Erroneous Knowledge Graph Interlinks

In [78], we have shown that outlier detection is not only applicable to numerical values, but also to other targets, such as knowledge graph interlinks. To that end, the interlinks are represented as a multi-dimensional feature vector, e.g., with each type of the respective entity in both knowledge graphs being a binary feature. In that feature space, standard outlier detection techniques such as *Local Outlier Factor* [11] or *cluster-based outlier detection* [35] can be used to assign outlier scores. Based on those scores, *implausible* links, such as a `owl:sameAs` assertion between a person and a book, can be identified based only on the overall distribution of all links, where such a combination is infrequent.

The work in [58] tries to learn arithmetic relations between attributes, e.g., *lessThan* or *greaterThan*, using probabilistic modeling. For example, the birth date of a person must be before her death date, the total area of a country must be larger than the area covered by water, etc. Violations of those relations are then used to identify errors.[28]

6.2. External Methods

Purely automatic external methods for error detection in knowledge graphs have limitations, e.g., in telling apart actual errors from unusual findings [94]. Semi-automatic approaches, which exploit human knowledge, have also been proposed.

6.2.1. Methods for Finding Erroneous Relations

Most external methods are targeted on finding erroneous relations in knowledge graphs. One of the few

[28]Note that an error here is not a single statement, but a pair of statements that cannot be true at the same time. Thus, the approach does not trivially lead to a fully automatic repairing mechanism (unless both statements are removed, which means that most likely, one correct statement is removed as well).

works is *DeFacto* [54]. The system uses a database of lexicalizations for predicates in DBpedia. Based on those lexicalizations, it transforms statements in DBpedia to natural language sentences, and uses a web search engine to find web pages containing those sentences. Statements with no or only very few web pages supporting the corresponding sentences are then assigned a low confidence score.

Apart from fully automatic methods, semi-automatic methods involving users have been proposed for validating knowledge graphs, such as crowdsourcing with microtasks [1]. In order to increase the user involvement and motivation, game-based approaches (i.e., games with a purpose) have been proposed [37,48,97,105]. In a wider sense, those can also be viewed as external methods, with the human in the loop being the external source of information.

Generally, a crucial issue with human computation is the size of web scale knowledge graphs. In [80], it has been argued that the time needed to validate the entire DBpedia knowledge graph with the crowdsourcing approach proposed in [1] – extrapolating the task completion times reported – would take more than 3,000 years. To overcome such scaling problems, we have recently proposed a clustering of inconsistencies identified by automatic means, which allows to present only representative examples to the human for inspection [82]. We have shown that most of the clusters have a common root cause in the knowledge graph construction (e.g., a wrong mapping rule or a programming error), so that by inspecting only a few dozen examples (and addressing the respective root causes), millions of statements can be corrected.

6.2.2. *Methods for Finding Erroneous Literal Values*

While most of the crowdsourcing approaches above are focusing on relations in the knowledge graph, the work in [1] uses similar mechanisms for validating knowledge graph interlinks and literal values.

In [60], an automatic approach using knowledge graph interlinks for detecting wrong numerical values is proposed. The authors exploit links between identical resources and apply different matching functions between properties in the individual sources. Facts in one knowledge graph are assumed to be wrong if multiple other sources have a consensus for a conflicting fact (e.g., a radically different population figure).

7. Findings from the Survey

From the survey in the last two sections, we can observe that is a large number of approaches proposed for knowledge graph refinement, both for automatic completion and for error detection. Tables 3 to 5 sum up the results from the previous section.

By taking a closer look at those results, we can derive some interesting findings, both with respect to the approaches, as well as with respect to evaluation methodologies.

7.1. *Approaches*

A first interesting observation is that our distinction into completion and error detection is a strict one. That is, there exist no approaches which do *both* completion and correction at the same time. The only exception we found is the pairing of the two approaches *SDType* and *SDValidate* [80], which are two closely related algorithms which share the majority of the computations and can output both completion axioms and errors.

For many of the approaches, it is not obvious why they were only used for one purpose. For example, many of the probabilistic and NLP-based completion approaches seek for evidence for missing axioms, e.g., by means of scanning text corpora. Similarly, many completion approaches ultimately compute a *confidence score*, which is then combined with a suitable threshold for completing a knowledge graph. In principle, they could also be used for error detection by flagging axioms for which *no* or only little evidence was found, or those with a low confidence score, as wrong.

Furthermore, in particular in the machine learning area, approaches exist which can be used for simultaneously creating a predictive model and creating weights for pieces of information. For example, random forests can assign weights to attributes [59], whereas boosting assign weights to instances [25], which can also be interpreted as outlier scores [16]. Likewise, there are anomaly detection methods that build on learning predictive models [34,83]. Such approaches could be a starting point for developing methods for simultaneous completion and error detection in knowledge graphs.

Along the same lines, there are hardly any among the error detection approaches which are also suitable for *correcting* errors, i.e., suggest fixes for the errors found. Here, a combination between completion and error detection methods could be of great value: once an error is detected, the erroneous axiom(s) could be

Table 3: Overview of knowledge graph completion approaches (part 1). Abbreviations used: Target (T=Types, R=Relations), Type (I=internal, E=external), Evaluation (RE=retrospective evaluation, PG=partial gold standard, either available (a) or unavailable (n/a), KG=evaluation against knowledge graph, SV=split validation, CV=cross validation), Metrics (P/R=precision and recall, A=accuracy, AUC-PR=area under precision-recall-curve, ROC=Area under the ROC curve, T=total new statements). Comp.: evaluation or materialization carried out on whole knowledge graph or not, Performance: computational performance reported or not.

Paper	Target	Type	Methods and Sources	Knowledge Graph(s)	Eval.	Metrics	Whole	Comp.
Paulheim [77]	T	I	Association Rule Mining	DBpedia	RE	P, T	no	yes
Nickel et al. [73]	T	I	Matrix Factorization	YAGO	KG (CV)	P/R, AUC-PR	yes	yes
Paulheim/Bizer [79,80]	T	I	Likelihood based	DBpedia, OpenCyc, Nell	KG, RE	P/R, T	yes	no
Sleeman et al. [99]	T	I	Topic Modeling	DBpedia	KG (SV)	P/R	no	no
Nuzzolese et al. [75]	T	E	different machine learning methods, Wikipedia link graph	DBpedia	PG (a)	P/R	yes	no
Gangemi et al. [28]	T	E	NLP on Wikipedia abstracts	DBpedia	PG (a)	P/R	no	yes
Kliegr [47]	T	E	NLP on Wikipedia abstracts	DBpedia	PG (n/a)	P/R	no	no
Aprosio et al. [3]	T	E	K-NN classification, different DBpedia language editions	DBpedia	PG (n/a)	P/R	yes	no
Dutta et al. [23]	T	E	Knowledge graph fusion (statistical)	NELL, DBpedia	PG (a)	P/R	no	no
Sleeman/Finin [98]	T	I, E	SVM, using other KGs	DBpedia, Freebase, Arnetminer	KG	P/R	no	no
Socher et al. [100]	R	I	Neural Tensor Network	WordNet, Freebase	KG (SV)	A	no	no
Krompaß et al. [50]	R	I	Latent variable models	DBpedia, Freebase, YAGO	KG (SV)	AUC-PR, ROC	no	no
Kolthoff and Dutta [49]	R	I	Rule mining	DBpedia, YAGO	KG	P/R	no	no
Zhao et al. [112]	R	I	Learning Embeddings	WordNet, Freebase	KG	P	no	no

Table 4: Overview of knowledge graph completion approaches (part 2). Abbreviations used: Target (T=Types, R=Relations), Type (I=internal, E=external), Evaluation (RE=retrospective evaluation, PG=partial gold standard, either available (a) or unavailable (n/a), KG=evaluation against knowledge graph, SV=split validation, CV=cross validation), Metrics (P/R=precision and recall, A=accuracy, AUC-PR=area under precision-recall-curve, T=total new statements). Comp.: evaluation or materialization carried out on whole knowledge graph or not, Performance: computational performance reported or not.

Paper	Target	Type	Methods and Sources	Knowledge Graph(s)	Eval.	Metrics	Whole	Comp.
Galárraga et al. [26]	R	I	Association Rule Mining	YAGO, DBpedia	KG, RE	A,T	yes	yes
Kim et al. [46]	R	I	Association Rule Mining	DBpedia	RE	P,T	yes	no
Bryl/Bizer [12]	R	E	Fusion of DBpedia language editions	DBpedia	PG (a)	A	no	no
Apriosio et al. [4]	R	E	Distant supervision and NLP techniques on Wikipedia text corpus	DBpedia	KG	P/R	no	no
Lange et al. [52]	R	E	Pattern learning on Wikipedia abstracts	(DBpedia)	KG (CV)	P/R	yes	yes
Wu et al. [109]	R	E	Pattern learning on Wikipedia articles, Web search	(DBpedia)	KG (CV)	P/R	no	no
Mintz et al. [68]	R	E	Distant supervision and NLP techniques on Wikipedia text corpus	Freebase	KG	P/R	no	no
Gerber/Ngonga Ngomo [30]	R	E	Distant supervision and NLP techniques on two corpora	DBpedia	KG, RE	P/R	no	no
Gerber et al. [29]	R	E	NLP and pattern mining on RSS news feeds	DBpedia	PG	P/R, A	yes	yes
West et al. [107]	R	E	search engines	Freebase	KG	P/R, rank	no	no
Mu noz et al. [69]	R	E	Statistical measures, machine learning, using Wikipedia tables	DBpedia	PG (a)	P/R	yes	no
Ritze et al. [89]	R, L	E	Schema and instance matching using HTML tables	DBpedia	PG (a)	P/R	yes	no
Paulheim/Ponzetto [84]	T, R	E	Statistical measures, Wikipedia list pages	DBpedia	–	–	no	no

Table 5: Overview of error detection approaches. Abbreviations used: Target (T=Types, R=Relations, L=Literals, I=Interlinks, S=Schema), Type (I=internal, E=external), Evaluation (RE=retrospective evaluation, PG=partial gold standard, either available (a) or unavailable (n/a), KG=evaluation against knowledge graph, SV=split validation, CV=cross validation), Metrics (P/R=precision and recall, A=accuracy, AUC-PR=area under precision-recall-curve, ROC=Area under the ROC curve, T=total new statements, RMSE=Root Mean Squared Error). Comp.: evaluation or materialization carried out on whole knowledge graph or not, Performance: computational performance reported or not.

Paper	Target	Type	Methods and Sources	Knowledge Graph(s)	Eval.	Metrics	Whole	Comp.
Ma et al. [63]	T	I	Reasoning, Association Rule Mining	DBpedia, Zhishi.me	RE	P, T	yes	no
Dong et al. [21]	R	I	Classification	Knowledge Vault	KG (SV)	AUC-PR	yes	no
Li et al. [58]	L	I	Probabilistic Modeling	DBpedia	RE	P, T	yes	yes
Nakashole et al. [70]	R	I	Reasoning	Prospera	RE	P, T	yes	yes
Paulheim/Bizer [80]	R	I	Probabilistic	DBpedia, NELL	RE	P	yes	no
Lehmann et al. [54]	R	E	Text pattern induction, Web search engines	DBpedia	KG (CV)	P/R, ROC, RMSE	no	yes
Töpper et al. [102]	R,S	I	Statistical methods, Reasoning	DBpedia	RE	P, T	yes	no
Jang et al. [42]	R	I	Statistical methods	DBpedia	RE	P,R	yes	no
Lehmann/Bühmann [53]	R,T	I	Reasoning, ILP	DBpedia, Open Cyc, seven smaller ontologies	RE	A	yes	yes
Wienand/Paulheim [108]	L	I	Outlier Detection	DBpedia	RE	P, T	no	no
Fleischhacker et al. [24]	L	I, E	Outlier Detection and Data Fusion with other KG	DBpedia, NELL	RE	AUC-ROC	no	no
Liu et al. [60]	L	E	Matching to other KGs	DBpedia	RE	P, T	no	no
Paulheim [78]	I	I	Outlier Detection	DBpedia + two linked graphs	PG (a)	P/R, ROC	yes	no
Acosta et al. [1]	L,I	E	Crowdsourcing	DBpedia	PG (a)	P	no	no
Waitelonis et al. [105]	R	E	Quiz game	DBpedia	RE	P, T	no	yes
Hees et al. [37]	R	E	Two-player game	DBpedia	RE	T	no	yes
Paulheim and Gangemi [82]	R	E	Reasoning, clustering, human inspection	DBpedia	RE	P, T	yes	yes

removed, and a correction algorithm could try to find a new (and, in the best case, more accurate) replacement for the removed axiom(s).

Another finding for error detection approaches is that those approaches usually output a list of potentially erroneous statements. Higher level patterns from those errors, which would hint at design level problems in the knowledge graph construction, are rarely derived (apart from the work presented in [82]). Such patterns, however, would be highly valuable for the parties involved in developing and curating knowledge graphs.

In addition to the strict separation of completion and correction, we also observe that most of the approaches focus on only one target, i.e., types, relations, literals, etc. Approaches that simultaneously try to complete or correct, e.g., type and relation assertions in a knowledge graph, are also quite rare.

For the approaches that perform completion, all works examined in this survey try to add missing types for or relations between *existing* entities in the knowledge graph. In contrast, we have not observed any approaches which populate the knowledge graph with *new* entities. Here, *entity set expansion* methods, which have been deeply investigated in the NLP field [76,93,106], would be an interesting fit to further increase the coverage of knowledge graphs, especially for less well-known long tail entities.

Another interesting observation is that, although the discussed works address knowledge *graphs*, only very few of them are, in the end, genuinely graph-based approaches. In many cases, simplistic transformations to a propositional problem formulation are taken. Here, methods from the graph mining literature still seek their application to knowledge graphs. In particular, for many of the methods applied in the works discussed above – such as outlier detection or association rule mining – graph-based variants have been proposed in the literature [2,44]. Likewise, graph kernel functions – which can be used in Support Vector Machines as well as other machine learning algorithms – have been proposed for RDF graphs [18,41,61] and hence could be applied to many web knowledge graphs.

7.2. Evaluation Methodologies

For evaluation methodologies, our first observation is that there are various different evaluation metrics being used in the papers examined. There is a clear tendency towards precision and recall (or precision and total number of statements for retrospective evaluations) are the most used metrics, with others – such as ROC curves, accuracy, or Root Mean Squared Error – occasionally being used as well.

With respect to the overall methodology, the results are more mixed. Evaluations using the knowledge graph as a silver standard, retrospective evaluations, and evaluations based on partial gold standards appear roughly at equal frequency, with retrospective validations mostly used for error detection. The latter is not too surprising, since due to the high quality of most knowledge graphs used for the evaluations, partial gold standards based on random samples are likely to contain only few errors. For partial gold standards, it is crucial to point out that the majority of authors make those partial gold standards public[29], which allows for replication and comparison.

DBpedia is the knowledge graph which is most frequently used for evaluation. This, in principle, makes the results comparable to a certain extent, although roughly each year, a new version of DBpedia is published, so that papers from different years are likely to be evaluated on slightly different knowledge graphs.

That being said, we have observed that roughly two out of three approaches evaluated on DBpedia are *only* evaluated on DBpedia. Along the same lines, about half of the approaches reviewed in this survey are only evaluated on *one* knowledge graph. This, in many cases, limits the significance of the results. For some works, it is clear that they can only work on a specific knowledge graph, e.g., DBpedia, *by design*, e.g., since they exploit the implicit linkage between a DBpedia entity and the corresponding Wikipedia page.

As discussed in section 2, knowledge graphs differ heavily in their characteristics. Thus, for an approach evaluated on only one graph, it is unclear whether it would perform similarly on another knowledge graph with different characteristics, or whether it exploits some (maybe not even obvious) characteristics of that knowledge graph, and/or overfits to particular characteristics of that graph.

Last, but not least, we have observed that only a minority of approaches have been evaluated on a whole, large-scale knowledge graph. Moreover, statements about computational performance are only rarely included in the corresponding papers[30]. In the age of

[29]For this survey, we counted a partial gold standard as public if there was a working download link in the paper, but we did not make any additional efforts to search for the gold standard, such as contacting the authors.

[30]Even though we were relaxed on this policy and counted also informal statements about the computational performance as a performance evaluation.

large-scale knowledge graphs, we think that this is a dimension that should not be neglected.

In order to make future works on knowledge graph evolution comparable, it would be useful to have a common selection of benchmarks. This has been done in other fields of the semantic web as well, such as for schema and instance matching [22], reasoning [8], or question answering [17]. Such benchmarks could serve both for comparison in the qualitative as well as the computational performance.

8. Conclusion

In this paper, we have presented a survey on knowledge base refinement methods. We distinguish completion from error detection, and internal from external methods. We have shown that a larger body of works exist which apply different methods, ranging from techniques from the machine learning field to NLP related techniques.

The survey has revealed that there are, at the moment, rarely any approaches which simultaneously try to improve completeness and correctness of knowledge graphs, and usually only address one target, such as type or relation assertions, or literal values. Holistic solutions which simultaneously improve the quality of knowledge graphs in many different aspects are currently not observed.

Looking at the evaluation methods, the picture is quite diverse. Different methods are applied, using either the knowledge graph itself as a silver standard, using a partial gold standard, or performing a retrospective evaluation, are about equally distributed. Furthermore, approaches are often only evaluated on one specific knowledge graph. This makes it hard to compare approaches and make general statements on their relative performance.

In addition, scalability issues are only rarely addressed by current research works. In the light of the advent of web-scale knowledge graphs, however, this is an aspect which will be of growing importance.

To sum up, this survey shows that automatic knowledge graph refinement is a relevant and flowering research area. At the same time, this survey has pointed out some uncharted territories on the research map, which we hope will inspire researchers in the area.

References

[1] Maribel Acosta, Amrapali Zaveri, Elena Simperl, Dimitris Kontokostas, Sören Auer, and Jens Lehmann. Crowdsourcing Linked Data quality assessment. In *The Semantic Web–ISWC 2013*, volume 8219 of *LNCS*, pages 260–276. Springer, Berlin Heidelberg, 2013. http://dx.doi.org/10.1007/978-3-642-41338-4_17.

[2] Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based anomaly detection and description: a survey. *Data Mining and Knowledge Discovery*, 29(3):1–63, 2014. http://dx.doi.org/10.1007/s10618-014-0365-y.

[3] Alessio Palmero Aprosio, Claudio Giuliano, and Alberto Lavelli. Automatic expansion of DBpedia exploiting Wikipedia cross-language information. In *The Semantic Web: Semantics and Big Data*, volume 7882 of *LNCS*, pages 397–411. Springer, Berlin Heidelberg, 2013. http://dx.doi.org/10.1007/978-3-642-38288-8_27.

[4] Alessio Palmero Aprosio, Claudio Giuliano, and Alberto Lavelli. Extending the Coverage of DBpedia Properties using Distant Supervision over Wikipedia. In *NLP&DBpedia*, volume 1064 of *CEUR Workshop Proceedings*, 2013. http://ceur-ws.org/Vol-1064/.

[5] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked Data – The Story So Far. *International journal on semantic web and information systems*, 5(3):1–22, 2009. http://dx.doi.org/10.4018/jswis.2009081901.

[6] Roi Blanco, Berkant Barla Cambazoglu, Peter Mika, and Nicolas Torzec. Entity Recommendations in Web Search. In *The Semantic Web–ISWC 2013*, volume 8219 of *LNCS*, pages 33–48. Springer, Berlin Heidelberg, 2013. http://dx.doi.org/10.1007/978-3-642-41338-4_3.

[7] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[8] Jürgen Bock, Peter Haase, Qiu Ji, and Raphael Volz. Benchmarking OWL reasoners. In *Workshop on Advancing Reasoning on the Web: Scalability and Commonsense*, volume 350 of *CEUR Workshop Proceedings*, 2008. http://ceur-ws.org/Vol-350/.

[9] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A Collaboratively Created Graph Database For Structuring Human Knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 1247–1250, New York, 2008. ACM. http://dx.doi.org/10.1145/1376616.1376746.

[10] Antoine Bordes and Evgeniy Gabrilovich. Constructing and Mining Web-scale Knowledge Graphs. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1967–1967, New York, 2014. ACM. http://dx.doi.org/10.1145/2623330.2630803.

[11] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. LOF: identifying density-based local outliers. *ACM SIGMOD Record*, 29(2):93–104, 2000. http://dx.doi.org/10.1145/335191.335388.

[12] Volha Bryl and Christian Bizer. Learning conflict resolution strategies for cross-language Wikipedia data fusion. In *Proceedings of the companion publication of the 23rd international conference on World wide web companion*, pages 1129–1134, Geneva, 2014. International World Wide Web Conferences Steering Committee. http://dx.doi.org/10.1145/2567948.2578999.

[13] Paul Buitelaar, Philipp Cimiano, and Bernardo Magnini. *On-*

tology Learning from Text: Methods, Evaluation and Applications, volume 123 of *Frontiers in Artificial Intelligence and Applications*. IOS press, Clifton, VA, 2005.

[14] Andrew Carlson, Justin Betteridge, Richard C Wang, Estevam R Hruschka Jr, and Tom M Mitchell. Coupled semi-supervised learning for information extraction. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 101–110, New York, 2010. ACM. http://dx.doi.org/10.1145/1718487.1718501.

[15] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41(3), 2009. http://dx.doi.org/10.1145/1541880.1541882.

[16] Nathalie Cheze and Jean-Michel Poggi. Iterated Boosting for Outlier Detection. In *Data Science and Classification*, pages 213–220. Springer, Berlin Heidelberg, 2006. http://dx.doi.org/10.1007/3-540-34416-0_23.

[17] Philipp Cimiano, Vanessa Lopez, Christina Unger, Elena Cabrio, Axel-Cyrille Ngonga Ngomo, and Sebastian Walter. Multilingual Question Answering over Linked Data (QALD-3): Lab Overview. In *Information Access Evaluation. Multilinguality, Multimodality, and Visualization*, volume 8138 of *LNCS*, pages 321–332. Springer, Berlin Heidelberg, 2013. http://dx.doi.org/10.1007/978-3-642-40802-1_30.

[18] Gerben KD de Vries. A fast approximation of the Weisfeiler-Lehman graph kernel for RDF data. In *Machine Learning and Knowledge Discovery in Databases*, volume 8188 of *LNCS*, pages 606–621. Springer, Berlin Heidelberg, 2013. http://dx.doi.org/10.1007/978-3-642-40988-2_39.

[19] Kathrin Dentler, Ronald Cornet, Annette ten Teije, and Nicolette de Keizer. Comparison of reasoners for large ontologies in the OWL 2 EL profile. *Semantic Web*, 2(2):71–87, 2011. http://dx.doi.org/10.3233/SW-2011-0034.

[20] Li Ding, Pranam Kolari, Zhongli Ding, and Sasikanth Avancha. Using ontologies in the semantic web: A survey. In *Ontologies*, volume 14 of *Integrated Series in Information Systems*, pages 79–113. Springer, US, 2007. http://dx.doi.org/10.1007/978-0-387-37022-4_4.

[21] Xin Luna Dong, K Murphy, E Gabrilovich, G Heitz, W Horn, N Lao, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge Vault: A Web-scale approach to probabilistic knowledge fusion. In *20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601–610, New York, 2014. ACM. http://dx.doi.org/10.1145/2623330.2623623.

[22] Zlatan Dragisic, Kai Eckert, Jerome Euzenat, Daniel Faria, Alfio Ferrara, Roger Granada, Valentina Ivanova, Ernesto Jimenez-Ruiz, Andreas Kempf, Patrick Lambrix, et al. Results of the Ontology Alignment Evaluation Initiative 2014. In *International Workshop on Ontology Matching*, volume 1317 of *CEUR Workshop Proceedings*, pages 61–104, 2014. http://ceur-ws.org/Vol-1317/.

[23] Arnab Dutta, Christian Meilicke, and Simone Paolo Ponzetto. A Probabilistic Approach for Integrating Heterogeneous Knowledge Sources. In *The Semantic Web: Trends and Challenges*, volume 8465 of *LNCS*, pages 286–301. Springer, Switzerland, 2014. http://dx.doi.org/10.1007/978-3-319-07443-6_20.

[24] Daniel Fleischhacker, Heiko Paulheim, Volha Bryl, Johanna Völker, and Christian Bizer. Detecting Errors in Numerical Linked Data Using Cross-Checked Outlier Detection. In *The Semantic Web–ISWC 2014*, volume 8796 of *LNCS*, pages 357–372. Springer, Switzerland, 2014. http://dx.doi.org/10.1007/978-3-319-11964-9_23.

[25] Yoav Freund and Robert E Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997. http://dx.doi.org/10.1006/jcss.1997.1504.

[26] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In *Proceedings of the 22nd international conference on World Wide Web*, pages 413–422, Geneva, 2013. International World Wide Web Conferences Steering Committee.

[27] Aldo Gangemi, Nicola Guarino, Claudio Masolo, and Alessandro Oltramari. Sweetening WordNet with DOLCE. *AI Magazine*, 24(3):13–24, 2003. http://dx.doi.org/10.1609/aimag.v24i3.1715.

[28] Aldo Gangemi, Andrea Giovanni Nuzzolese, Valentina Presutti, Francesco Draicchio, Alberto Musetti, and Paolo Ciancarini. Automatic typing of DBpedia entities. In *The Semantic Web–ISWC 2012*, volume 7649 of *LNCS*, pages 65–81. Springer, Berlin Heidelberg, 2012. http://dx.doi.org/10.1007/978-3-642-35176-1_5.

[29] Daniel Gerber, Sebastian Hellmann, Lorenz Bühmann, Tommaso Soru, Ricardo Usbeck, and Axel-Cyrille Ngonga Ngomo. Real-time RDF extraction from unstructured data streams. In *The Semantic Web–ISWC 2013*, volume 8218 of *LNCS*, pages 135–150. Springer, Berlin Heidelberg, 2013. http://dx.doi.org/10.1007/978-3-642-41335-3_9.

[30] Daniel Gerber and A-C Ngonga Ngomo. Bootstrapping the Linked Data web. In *Workshop on Web Scale Knowledge Extraction*, 2011. http://iswc2011.semanticweb.org/fileadmin/iswc/Papers/Workshops/WeKEx/paper_3.pdf.

[31] Lise Getoor and Christopher P Diehl. Link Mining: A Survey. *ACM SIGKDD Explorations Newsletter*, 7(2):3–12, 2005. http://dx.doi.org/10.1145/1117454.1117456.

[32] T. Groza, A. Oellrich, and N. Collier. Using silver and semi-gold standard corpora to compare open named entity recognisers. In *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 481–485, Piscataway, New Jersey, 2013. IEEE. http://dx.doi.org/10.1109/BIBM.2013.6732541.

[33] Harry Halpin, Patrick J. Hayes, James P. McCusker, Deborah L. McGuinness, and Henry S. Thompson. When owl:sameAs Isn't the Same: An Analysis of Identity in Linked Data. In *The Semantic Web – ISWC 2010*, volume 6496 of *LNCS*, pages 305–320. Springer, Berlin Heidelberg, 2010. http://dx.doi.org/10.1007/978-3-642-17746-0_20.

[34] Simon Hawkins, Hongxing He, Graham Williams, and Rohan Baxter. Outlier Detection Using Replicator Neural Networks. In *Data Warehousing and Knowledge Discovery*, volume 2454 of *LNCS*, pages 170–180. Springer, Berlin Heidelberg, 2002. http://dx.doi.org/10.1007/3-540-

46145-0_17.

[35] Zengyou He, Xiaofei Xu, and Shengchun Deng. Discovering cluster-based local outliers. *Pattern Recognition Letters*, 24(9):1641–1650, 2003. http://dx.doi.org/10.1016/S0167-8655(03)00003-5.

[36] Marti A Hearst. Automatic Acquisition of Hyponyms from Large Text Corpora. In *Proceedings of the 14th Conference on Computational Linguistics (Volume 2)*, pages 539–545, Stroudsburg, PA, 1992. Association for Computational Linguistics. http://dx.doi.org/10.3115/992133.992154.

[37] Jörn Hees, Thomas Roth-Berghofer, Ralf Biedert, Benjamin Adrian, and Andreas Dengel. BetterRelations: Using a Game to Rate Linked Data Triples. In *KI 2011: Advances in Artificial Intelligence*, volume 7006 of *LNCS*, pages 134–138. Springer, Berlin Heidelberg, 2011. http://dx.doi.org/10.1007/978-3-642-24455-1_12.

[38] Jochen Hipp, Ulrich Güntzer, and Gholamreza Nakhaeizadeh. Algorithms for Association Rule Mining – a General Survey and Comparison. *ACM SIGKDD Explorations Newsletter*, 2(1):58–64, 2000. http://dx.doi.org/10.1145/360402.360421.

[39] Victoria J Hodge and Jim Austin. A Survey of Outlier Detection Methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004. http://dx.doi.org/10.1007/s10462-004-4304-y.

[40] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. Robust Disambiguation of Named Entities in Text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792, Edinburgh, 2011. Association for Computational Linguistics. http://www.aclweb.org/anthology/D11-1072.

[41] Yi Huang, Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A Scalable Kernel Approach to Learning in Semantic Graphs with Applications to Linked Data. *Semantic Web*, 5(1):5–22, 2010. http://dx.doi.org/10.3233/SW-130100.

[42] Saemi Jang, Megawati, Jiyeon Choi, and Mun Yong Yi. Semi-Automatic Quality Assessment of Linked Data without Requiring Ontology. In *Workshop on NLP and DBpedia*, 2015. https://nlpdbpedia2015.files.wordpress.com/2015/08/nlpdbpedia_2015_submission_2.pdf.

[43] Qiu Ji, Zhiqiang Gao, and Zhisheng Huang. Reasoning with Noisy Semantic Data. In *The Semanic Web: Research and Applications*, volume 6644 of *LNCS*, pages 497–502. Springer, Berlin Heidelberg, 2011. http://dx.doi.org/10.1007/978-3-642-21064-8_42.

[44] Chuntao Jiang, Frans Coenen, and Michele Zito. A survey of Frequent Subgraph Mining Algorithms. *The Knowledge Engineering Review*, 28(1):75–105, 2013. http://dx.doi.org/10.1017/S0269888912000331.

[45] Ning Kang, Erik M van Mulligen, and Jan A Kors. Training text chunkers on a silver standard corpus: can silver replace gold? *BMC bioinformatics*, 13(1):17, 2012. http://dx.doi.org/10.1186/1471-2105-13-17.

[46] Jiseong Kim, Eun-Kyung Kim, Yousung Won, Sangha Nam, and Key-Sun Choi. The Association Rule Mining System for Acquiring Knowledge of DBpedia from Wikipedia Categories. In *Workshop on NLP and DBpedia*, 2015. https://nlpdbpedia2015.files.wordpress.com/2015/08/nlpdbpedia_2015_submission_13.pdf.

[47] Tomáš Kliegr. Linked Hypernyms: Enriching DBpedia with Targeted Hypernym Discovery. *Web Semantics: Science, Services and Agents on the World Wide Web*, 31:59–69, 2015. http://dx.doi.org/10.1016/j.websem.2014.11.001.

[48] Magnus Knuth, Johannes Hercher, and Harald Sack. Collaboratively Patching Linked Data. In *Workshop on Usage Analysis and the Web of Data (USEWOD*, 2012. http://arxiv.org/abs/1204.2715.

[49] Christian Kolthoff and Arnab Dutta. Semantic Relation Composition in Large Scale Knowledge Bases. In *3rd Workshop on Linked Data for Information Extraction*, volume 1467 of *CEUR Workshop Proceedings*, pages 34–47, 2015. http://ceur-ws.org/Vol-1467/.

[50] Denis Krompaß, Stephan Baier, and Volker Tresp. Type-Constrained Representation Learning in Knowledge Graphs. In *International Semantic Web Conference*, volume 9366 of *LNCS*, pages 640–655, Switzerland, 2015. Springer. http://dx.doi.org/10.1007/978-3-319-25007-6_37.

[51] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *18th International Conference on Machine Learning*, pages 282–289. San Francisco, CA, Morgan Kaufmann, 2001.

[52] Dustin Lange, Christoph Böhm, and Felix Naumann. Extracting structured information from Wikipedia articles to populate infoboxes. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management (CIKM)*, pages 1661–1664, New York, 2010. ACM. http://dx.doi.org/10.1145/1871437.1871698.

[53] Jens Lehmann and Lorenz Bühmann. ORE – a tool for repairing and enriching knowledge bases. In *The Semantic Web–ISWC 2010*, volume 6497 of *LNCS*, pages 177–193. Springer, Berlin Heidelberg, 2010. http://dx.doi.org/10.1007/978-3-642-17749-1_12.

[54] Jens Lehmann, Daniel Gerber, Mohamed Morsey, and Axel-Cyrille Ngonga Ngomo. DeFacto – Deep Fact Validation. In *The Semantic Web–ISWC 2012*, volume 7649 of *LNCS*, pages 312–327. Springer, Berlin Heidelberg, 2012. http://dx.doi.org/10.1007/978-3-642-35176-1_20.

[55] Jens Lehmann and Pascal Hitzler. Concept Learning in Description Logics Using Refinement Operators. *Machine Learning*, 78:203–250, 2010. http://dx.doi.org/10.1007/s10994-009-5146-2.

[56] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. DBpedia – A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web Journal*, 6(2), 2013. http://dx.doi.org/10.3233/SW-140134.

[57] Douglas B Lenat. CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38, 1995. http://dx.doi.org/10.1145/219717.219745.

[58] Huiying Li, Yuanyuan Li, Feifei Xu, and Xinyu Zhong. Prob-

abilistic Error Detecting in Numerical Linked Data. In *Database and Expert Systems Applications*, volume 9261 of *LNCS*, pages 61–75, International, 2015. Springer. http://dx.doi.org/10.1007/978-3-319-22849-5_5.

[59] Andy Liaw and Matthew Wiener. Classification and Regression by randomForest. *R news*, 2(3):18–22, 2002. http://www.bios.unc.edu/~dzeng/BIOS740/randomforest.pdf.

[60] Shuangyan Liu, Mathieu d'Aquin, and Enrico Motta. Towards Linked Data Fact Validation through Measuring Consensus. In *Workshop on Linked Data Quality*, volume 1376 of *CEUR Workshop Proceedings*, 2015. http://ceur-ws.org/Vol-1376/.

[61] Uta Lösch, Stephan Bloehdorn, and Achim Rettinger. Graph Kernels for RDF Data. In *The Semantic Web: Research and Applications*, volume 7295 of *LNCS*, pages 134–148. Springer, Berlin Heidelberg, 2012. http://dx.doi.org/10.1007/978-3-642-30284-8_16.

[62] Marko Luther, Thorsten Liebig, Sebastian Böhm, and Olaf Noppens. Who the Heck is the Father of Bob? In *The Semantic Web: Research and Applications*, volume 5554 of *LNCS*, pages 66–80. Springer, Berlin Heidelberg, 2009. http://dx.doi.org/10.1007/978-3-642-02121-3_9.

[63] Yanfang Ma, Huan Gao, Tianxing Wu, and Guilin Qi. Learning Disjointness Axioms With Association Rule Mining and Its Application to Inconsistency Detection of Linked Data. In Dongyan Zhao, Jianfeng Du, Haofen Wang, Peng Wang, Donghong Ji, and Jeff Z. Pan, editors, *The Semantic Web and Web Science*, volume 480 of *Communications in Computer and Information Science*, pages 29–41. Springer, Berlin Heidelberg, 2014. http://dx.doi.org/10.1007/978-3-662-45495-4_3.

[64] Alexander Maedche. *Ontology Learning for the Semantic Web*. Springer Science & Business Media, Luxembourg, 2002.

[65] Farzaneh Mahdisoltani, Joanna Biega, and Fabian M. Suchanek. YAGO3: A Knowledge Base from Multilingual Wikipedias. In *Conference on Innovative Data Systems Research*, 2015. http://www.cidrdb.org/cidr2015/Papers/CIDR15_Paper1.pdf.

[66] Robert Meusel, Petar Petrovski, and Christian Bizer. The WebDataCommons Microdata, RDFa and Microformat Dataset Series. In *The Semantic Web–ISWC 2014*, volume 8796 of *LNCS*, pages 277–292. Springer, International, 2014. http://dx.doi.org/10.1007/978-3-319-11964-9_18.

[67] George A Miller. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995. http://dx.doi.org/10.1145/219717.219748.

[68] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 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*, pages 1003–1011, Stroudsburg, PA, 2009. Association for Computational Linguistics.

[69] Emir Muñoz, Aidan Hogan, and Alessandra Mileo. Triplifying Wikipedia's Tables. In *Linked Data for Information Extraction*, volume 1057 of *CEUR Workshop Proceedings*, 2013. http://ceur-ws.org/Vol-1057/.

[70] Ndapandula Nakashole, Martin Theobald, and Gerhard Weikum. Scalable knowledge harvesting with high precision and high recall. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 227–236, New York, 2011. ACM. http://dx.doi.org/10.1145/1935826.1935869.

[71] Markus Nentwig, Michael Hartung, Axel-Cyrille Ngonga Ngomo, and Erhard Rahm. A Survey of Current Link Discovery Frameworks. *Semantic Web*, (to appear), 2015. http://dx.doi.org/10.3233/SW-150210.

[72] Jennifer Neville and David Jensen. Iterative classification in relational data. In *Proc. AAAI-2000 Workshop on Learning Statistical Models from Relational Data*, pages 13–20, Palo Alto, CA, 2000. AAAI. http://www.aaai.org/Library/Workshops/2000/ws00-06-007.php.

[73] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. Factorizing YAGO: Scalable Machine Learning for Linked Data. In *Proceedings of the 21st International Conference on World Wide Web*, pages 271–280, New York, 2012. ACM. http://dx.doi.org/10.1145/2187836.2187874.

[74] Joel Nothman, Nicky Ringland, Will Radford, Tara Murphy, and James R Curran. Learning multilingual named entity recognition from Wikipedia. *Artificial Intelligence*, 194:151–175, 2013. http://dx.doi.org/10.1016/j.artint.2012.03.006.

[75] Andrea Giovanni Nuzzolese, Aldo Gangemi, Valentina Presutti, and Paolo Ciancarini. Type inference through the analysis of Wikipedia links. In *Linked Data on the Web*, volume 937 of *CEUR Workshop Proceedings*, 2012. http://ceur-ws.org/Vol-937/.

[76] Patrick Pantel, Eric Crestan, Arkady Borkovsky, Ana-Maria Popescu, and Vishnu Vyas. Web-scale Distributional Similarity and Entity Set Expansion. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2*, pages 938–947, Stroudsburg, PA, 2009. Association for Computational Linguistics. http://www.aclweb.org/anthology/D09-1098.

[77] Heiko Paulheim. Browsing Linked Open Data with Auto Complete. *Semantic Web Challenge*, 2012. https://km.aifb.kit.edu/sites/swc/2012/submissions/swc2012_submission_15.pdf.

[78] Heiko Paulheim. Identifying Wrong Links between Datasets by Multi-dimensional Outlier Detection. In *International Workshop on Debugging Ontologies and Ontology Mappings*, volume 1162 of *CEUR Workshop Proceedings*, pages 27–38, 2014. http://ceur-ws.org/Vol-1162/.

[79] Heiko Paulheim and Christian Bizer. Type Inference on Noisy RDF Data. In *The Semantic Web–ISWC 2013*, volume 8218 of *LNCS*, pages 510–525. Springer, Berlin Heidelberg, 2013. http://dx.doi.org/10.1007/978-3-642-41335-3_32.

[80] Heiko Paulheim and Christian Bizer. Improving the Quality of Linked Data Using Statistical Distributions. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 10(2):63–86, 2014. http://dx.doi.org/10.4018/ijswis.2014040104.

[81] Heiko Paulheim and Johannes Fürnkranz. Unsupervised Generation of Data Mining Features from Linked Open Data. In *2nd international conference on web intelligence, mining and semantics*, page 31, New York, 2012. ACM. http://dx.doi.org/10.1145/2254129.2254168.

[82] Heiko Paulheim and Aldo Gangemi. Serving DBpedia with DOLCE–More than Just Adding a Cherry on Top. In *International Semantic Web Conference*, volume 9366 of *LNCS*, International, 2015. Springer. http://dx.doi.org/10.1007/978-3-319-25007-6_11.

[83] Heiko Paulheim and Robert Meusel. A Decomposition of the Outlier Detection Problem into a Set of Supervised Learning Problems. *Machine Learning*, 100(2-3):509–531, 2015. http://dx.doi.org/10.1007/s10994-015-5507-y.

[84] Heiko Paulheim and Simone Paolo Ponzetto. Extending DBpedia with Wikipedia List Pages. In *1st International Workshop on NLP and DBpedia*, volume 1064 of *CEUR Workshop Proceedings*, 2013. http://ceur-ws.org/Vol-1064/.

[85] Youen Péron, Frédéric Raimbault, Gildas Ménier, and Pierre-François Marteau. On the detection of inconsistencies in RDF data sets and their correction at ontological level. Technical Report 00635854, HAL, 2011. https://hal.archives-ouvertes.fr/hal-00635854/en/.

[86] Leo L Pipino, Yang W Lee, and Richard Y Wang. Data quality assessment. *Communications of the ACM*, 45(4):211–218, 2002. http://dx.doi.org/10.1145/505248.506010.

[87] Axel Polleres, Aidan Hogan, Andreas Harth, and Stefan Decker. Can we ever catch up with the Web? *Semantic Web*, 1(1):45–52, 2010. http://dx.doi.org/10.3233/SW-2010-0016.

[88] Petar Ristoski and Heiko Paulheim. A Comparison of Propositionalization Strategies for Creating Features from Linked Open Data. In *Linked Data for Knowledge Discovery*, volume 1232 of *CEUR Workshop Proceedigns*, 2014. http://ceur-ws.org/Vol-1232/.

[89] Dominique Ritze, Oliver Lehmberg, and Christian Bizer. Matching HTML Tables to DBpedia. In *Proceedings of the 5th International Conference on Web Intelligence, Mining and Semantics*, page 10, New York, 2015. ACM. http://dx.doi.org/10.1145/2797115.2797118.

[90] Giuseppe Rizzo and Raphaël Troncy. NERD: Evaluating Named Entity Recognition Tools in the Web of Data. In *Workshop on Web Scale Knowledge Extraction (WEKEX'11)*, 2011. http://nerd.eurecom.fr/ui/paper/Rizzo_Troncy-wekex2011.pdf.

[91] Stuart Russell and Peter Norvig. *Artificial Intelligence: a Modern Approach*. Pearson, London, 1995.

[92] Samuel Sarjant, Catherine Legg, Michael Robinson, and Olena Medelyan. "All you can eat" ontology-building: Feeding Wikipedia to Cyc. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology-Volume 01*, pages 341–348, Piscataway, NJ, 2009. IEEE Computer Society. http://dx.doi.org/10.1109/WI-IAT.2009.60.

[93] Luis Sarmento, Valentin Jijkuon, Maarten de Rijke, and Eugenio Oliveira. "More Like These": Growing Entity Classes from Seeds. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 959–962, New York, 2007. ACM. http://dx.doi.org/10.1145/1321440.1321585.

[94] Benjamin Schäfer, Petar Ristoski, and Heiko Paulheim. What is Special about Bethlehem, Pennsylvania? – Identifying Unexpected Facts about DBpedia Entities. In *ISWC 2015 - Posters and Demonstrations*, volume 1486 of *CEUR Workshop Proceedigns*, 2015. http://ceur-ws.org/Vol-1486/.

[95] Max Schmachtenberg, Christian Bizer, and Heiko Paulheim. Adoption of the Linked Data Best Practices in Different Topical Domains. In *International Semantic Web Conference*, volume 8796 of *LNCS*, International, 2014. Springer. http://dx.doi.org/10.1007/978-3-319-11964-9_16.

[96] Carlos N Silla Jr and Alex A Freitas. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72, 2011. http://dx.doi.org/10.1007/s10618-010-0175-9.

[97] Katharina Siorpaes and Martin Hepp. Games with a Purpose for the Semantic Web. *IEEE Intelligent Systems*, (3):50–60, 2008. http://dx.doi.org/10.1109/MIS.2008.45.

[98] Jennifer Sleeman and Tim Finin. Type Prediction for Efficient Coreference Resolution in Heterogeneous Semantic Graphs. In *Semantic Computing (ICSC), 2013 IEEE Seventh International Conference on*, pages 78–85, Piscataway, NJ, 2013. IEEE. http://dx.doi.org/10.1109/ICSC.2013.22.

[99] Jennifer Sleeman, Tim Finin, and Anupam Joshi. Topic Modeling for RDF Graphs. In *3rd Workshop on Linked Data for Information Extraction*, volume 1467 of *CEUR Workshop Proceedings*, 2015. http://ceur-ws.org/Vol-1467/.

[100] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning With Neural Tensor Networks for Knowledge Base Completion. In *Advances in Neural Information Processing Systems 26 (NIPS 2013)*, pages 926–934. Curran Associates, Inc., Newry, 2013. http://papers.nips.cc/paper/5028-reasonin.

[101] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. YAGO: A Core of Semantic Knowledge Unifying WordNet and Wikipedia. In *16th international conference on World Wide Web*, pages 697–706, New York, 2007. ACM. http://dx.doi.org/10.1145/1242572.1242667.

[102] Gerald Töpper, Magnus Knuth, and Harald Sack. DBpedia Ontology Enrichment for Inconsistency Detection. In *Proceedings of the 8th International Conference on Semantic Systems*, pages 33–40, New York, 2012. ACM. http://dx.doi.org/10.1145/2362499.2362505.

[103] G Tsoumakas et al. Multi Label Classification: An Overview. *International Journal of Data Warehousing and Mining*, 3(3):1–13, 2007. http://dx.doi.org/10.4018/jdwm.2007070101.

[104] Denny Vrandečić and Markus Krötzsch. Wikidata: a Free Collaborative Knowledge Base. *Communications of the ACM*, 57(10):78–85, 2014. http://dx.doi.org/10.1145/2629489.

[105] Jörg Waitelonis, Nadine Ludwig, Magnus Knuth, and Harald Sack. WhoKnows? – Evaluating Linked Data Heuristics with a Quiz that Cleans Up DBpedia. *International Journal of Interactive Technology and Smart Education*, 8(4):236–248, 2011. http://dx.doi.org/10.1108/17415651111189478.

[106] Richard C Wang and William W Cohen. Iterative Set Expansion of Named Entities using the Web. In *Eighth IEEE International Conference on Data Mining*, pages 1091–1096, Piscataway, NJ, 2008. IEEE.

http://dx.doi.org/10.1109/ICDM.2008.145.

[107] Robert West, Evgeniy Gabrilovich, Kevin Murphy, Shaohua Sun, Rahul Gupta, and Dekang Lin. Knowledge Base Completion via Search-Based Question Answering. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 515–526, New York, 2014. ACM. http://dx.doi.org/10.1145/2566486.2568032.

[108] Dominik Wienand and Heiko Paulheim. Detecting Incorrect Numerical Data in DBpedia. In *The Semantic Web: Trends and Challenges*, volume 8465 of *LNCS*, pages 504–518. Springer, International, 2014. http://dx.doi.org/10.1007/978-3-319-07443-6_34.

[109] Fei Wu, Raphael Hoffmann, and Daniel S Weld. Information extraction from Wikipedia: Moving down the long tail. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 731–739, New York, 2008. ACM. http://dx.doi.org/10.1145/1401890.1401978.

[110] Amrapali Zaveri, Dimitris Kontokostas, Mohamed A Sherif, Lorenz Bühmann, Mohamed Morsey, Sören Auer, and Jens Lehmann. User-driven Quality Evaluation of DBpedia. In *9th International Conference on Semantic Systems (I-SEMANTICS '13)*, pages 97–104, New York, 2013. ACM. http://dx.doi.org/10.1145/2506182.2506195.

[111] Amrapali Zaveri, Anisa Rula, Andrea Maurino, Ricardo Pietrobon, Jens Lehmann, Sören Auer, and Pascal Hitzler. Quality Assessment Methodologies for Linked Open Data. *Semantic Web Journal*, 7(1):63–93, 2015. http://dx.doi.org/10.3233/SW-150175.

[112] Yu Zhao, Sheng Gao, Patrick Gallinari, and Jun Guo. Knowledge base completion by learning pairwise-interaction differentiated embeddings. *Data Mining and Knowledge Discovery*, 29(5):1486–1504, 2015. http://dx.doi.org/10.1007/s10618-015-0430-1.

[113] Antoine Zimmermann, Christophe Gravier, Julien Subercaze, and Quentin Cruzille. Nell2RDF: Read the Web, and Turn it into RDF. In *Knowledge Discovery and Data Mining meets Linked Open Data*, volume 992 of *CEUR Workshop Proceedings*, pages 2–8, 2013. http://ceur-ws.org/Vol-992/.