Information Extraction From Co-Occurring Similar Entities

Nicolas Heist
nico@informatik.uni-mannheim.de
Data and Web Science Group
University of Mannheim, Germany

Heiko Paulheim
heiko@informatik.uni-mannheim.de
Data and Web Science Group
University of Mannheim, Germany

ABSTRACT

Knowledge about entities and their interrelations is a crucial factor of success for tasks like question answering or text summarization. Publicly available knowledge graphs like Wikidata or DBpedia are, however, far from being complete. In this paper, we explore how information extracted from similar entities that co-occur in structures like tables or lists can help to increase the coverage of such knowledge graphs. In contrast to existing approaches, we do not focus on relationships within a listing (e.g., between two entities in a table row) but on the relationship between a listing's subject entities and the context of the listing. To that end, we propose a descriptive rule mining approach that uses distant supervision to derive rules for these relationships based on a listing's context. Extracted from a suitable data corpus, the rules can be used to extend a knowledge graph with novel entities and assertions. In our experiments we demonstrate that the approach is able to extract up to 3M novel entities and 30M additional assertions from listings in Wikipedia. We find that the extracted information is of high quality and thus suitable to extend Wikipedia-based knowledge graphs like DBpedia, YAGO, and CaLiGraph. For the case of DBpedia, this would result in an increase of covered entities by roughly 50%.

CCS CONCEPTS

• **Information systems → Information extraction**; *Data extraction and integration*; *Association rules.*

KEYWORDS

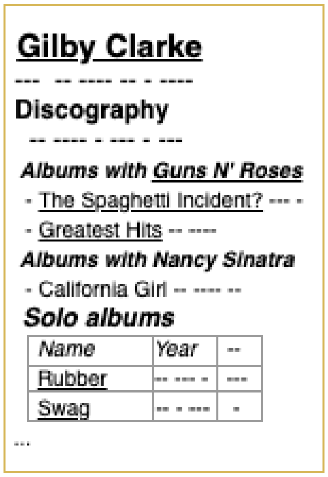
Entity co-occurrence, Information extraction, Novel entity detection, CaLiGraph, DBpedia

ACM Reference Format:
Nicolas Heist and Heiko Paulheim. 2021. Information Extraction From Co-Occurring Similar Entities. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3442381.3449836

1 INTRODUCTION

1.1 Motivation and Problem

In tasks like question answering, text summarization, or entity disambiguation, it is essential to have background information about the involved entities. With entity linking tools like DBpedia Spotlight [19] or Falcon [26], one can easily identify named entities in text and retrieve the respective entity in a background entity hub of the linking tool (e.g. in a wiki like Wikipedia or in a knowledge graph like DBpedia [14]). This is, however, only possible if the entity in question is contained in the respective entity hub [29].



Gilby Clarke

Discography

Albums with Guns N' Roses

- The Spaghetti Incident?
- Greatest Hits

Albums with Nancy Sinatra

- California Girl

Solo albums

Name	*Year*	--
Rubber	-- --- -	---
Swag	-- - ---	-

...

Figure 1: Simplified view on the Wikipedia page of Gilby Clarke with a focus on its title, sections, and listings.

The trend of entities added to publicly available knowledge graphs in recent years indicates that they are far from being complete. The number of entities in Wikidata [31], for example, grew by 37% in the time from October 2019 (61.7M) to October 2020 (84.5M). In the same time, the number of statements increased by 41% from 770M to 1085M.[1] According to [9], Wikidata describes the largest number of entities and comprises – in terms of entities – other open knowledge graphs to a large extent. Consequently, this problem applies to all public knowledge graphs, and particularly so for long-tail and emerging entities [6].

Automatic information extraction approaches can help mitigating this problem if the approaches can make sure that the extracted information is of high quality. While the performance of open information extraction systems (i.e. systems that extract information from general web text) has improved in recent years [4, 16, 27], the quality of extracted information has not yet reached a level where an integration into knowledge graphs like DBpedia should be done without further filtering.

The extraction of information from semi-structured data is in general less error-prone and already proved to yield high-quality results as, for example, DBpedia itself is extracted primarily from Wikipedia infoboxes; further approaches use the category system of Wikipedia [10, 28, 33] or its list pages [11, 24]. Many more

WWW '21, April 19–23, 2021, Ljubljana, Slovenia

ACM ISBN 978-1-4503-8312-7/21/04.
https://doi.org/10.1145/3442381.3449836

[1] https://tools.wmflabs.org/wikidata-todo/stats.php

approaches focus on tables (in Wikipedia or the web) as semi-structured data source to extract entities and relations (see [36] for a comprehensive survey). The focus of recent web table-based approaches like Zhang et al. [35] is set on recognizing entities and relationships within a table. Considering Fig. 1, the table below the section *Solo albums* may be used to discover the publication years of albums (relation extraction) or discover additional unknown albums that are listed in further rows below *Rubber* and *Swag* (entity and type detection).

The focus of this paper is broader with respect to two dimensions: First, we extract information from any kind of structure where similar entities co-occur. In Fig. 1, we would consider both tables and lists (e.g. the list in the section *Albums with Guns N' Roses*). We refer to these co-occurrence structures as listings. Second, we consider only the subject entities (SE) of listings. In our previous work we defined SE with respect to Wikipedia list pages as *"the instances of the concept expressed by the list page"* [11]. Considering the *List of Japanese speculative fiction writers*, its SE comprise all Japanese speculative fiction writers mentioned in listings of the page. While in [11] the concept of SE is made explicit by the list page, we deal with arbitrary listings in this paper. We thus assume the concept may not be explicit or it may be indicated as part of the page in which the listing appears (e.g. in the table header, or the page title). Therefore, to each entity in a listing appearing as instance to a common concept, we will further refer as subject entity. The purpose of this work is to exploit the relationship between the SE of a listing and the listing context. For Fig. 1, this means we extract that all SE on the page's listings are albums with the artist *Gilby Clarke*, that *The Spaghetti Incident?* is an album by *Guns N' Roses*, and so on.

To that end, we propose to learn these characteristics of a listing with respect to the types and contextual relations of its SE. In an ideal setting we know the SE of a listing and we are able to retrieve all information about them from a knowledge graph – the characteristics of a listing are then simply the types and relations that are shared by all SE. But uncertainty is introduced by several factors:

- SE can only be determined heuristically. In previous work [11], we achieved a precision of 90% for the recognition of SE in Wikipedia listings.
- Cross-domain knowledge graphs are not complete. According to the open world assumption (OWA), the absence of a fact in a knowledge graph does not imply its incorrectness.
- Web tables have a median of 6 rows,[2] and Wikipedia listings have a median of 8 rows. Consequently, many listings only have a small number of SE from which the characteristics can be inferred.

As a result, considering each listing in isolation either leads to a substantial loss of information (as listings with insufficient background information are disregarded) or to a high generalization error (as decisions are made based on insufficient background information).

We observe that the context of a listing is often a strong indicator for its characteristics. In Fig. 1, the title of the top section *Discography* indicates that its listings contain some kind of musical works, and the section title *Albums with Guns N' Roses* provides more detailed information. Our second observation is that these patterns repeat when looking at a coherent data corpus. The Wikipedia page of *Axl Rose*,[3] for example, contains the same constellation of sections.

Considering listing characteristics with respect to their context can thus yield in more general insights than considering every listing in isolation. For example, the musical works of many artists in Wikipedia are listed under the top section *Discography*. Hence, we could learn the axioms

$$\exists topSection.\{\text{"Discography"}\} \sqsubseteq \texttt{MusicalWork} \qquad (1)$$

and

$$\exists topSection.\{\text{"Discography"}\} \sqsubseteq \exists artist.\{<PageEntity>\} \qquad (2)$$

which are then applicable to any listing with the top section *Discography* in Wikipedia.

1.2 Approach and Contributions

In this work, we frame the task of finding descriptive rules for listings based on their context as association rule mining problem [1]. We define rule metrics that take the inherent uncertainty into account and make sure that rules are frequent (rule support), correct (rule confidence), and consistent over all listings (rule consistency). Furthermore, we present an approach that executes the complete pipeline from identification of SE to the extraction of novel entities and assertions with Wikipedia as data corpus. To find a reasonable balance between correctness and coverage of the rules, we set the thresholds based on a heuristic that takes the distribution of named entity tags over entities as well as existing knowledge in a knowledge graph into account. Applying the approach, we show that we can enhance the knowledge graphs DBpedia with up to 2.9M entities and 8.3M assertions, and CaLiGraph[4] with up to 3M entities and 30.4M assertions with an overall correctness of more than 90%.

To summarize, the contributions of this paper are as follows:

- We formulate the task of information extraction from co-occurring similar entities in listings and show how to derive descriptive rules for listing characteristics based on the listing context (Sec. 3).
- We present an approach that learns descriptive rules for listings in Wikipedia and is capable of extracting several millions of novel entities and assertions for Wikipedia-based knowledge graphs (Sec. 4).
- In our evaluation we demonstrate the high quality of the extracted information and analyze the shortcomings of the approach (Sec. 5).

The produced code is part of the CaLiGraph extraction framework and publicly available.[5]

[2] According to the WDC Web Table Corpus 2015: http://webdatacommons.org/webtables/.

[3] https://en.wikipedia.org/wiki/Axl_Rose

[4] http://caligraph.org

[5] https://github.com/nheist/CaLiGraph

2 RELATED WORK

The work presented in this paper is a flavour of *knowledge graph completion*, more precisely, of adding new entities to a knowledge graph [22]. We use rules based on page context to infer facts about co-occurring entities. In particular, we focus on co-occurrence of entities within document listings, where co-occurrence refers to proximity in page layout. Hence, in this section, we discuss related works w.r.t. knowledge graph completion from listings, exploitation of listing context, as well as rule learning for knowledge graphs.

2.1 Knowledge Graph Completion from Listings

Knowledge graph completion using information in web tables has already been an active research area in the last several years. In 2016, Ritze et al. [25] profiled the potential of web tables in the WDC Web Table Corpus. Using the T2K Match framework, they match web tables to DBpedia and find that the best results for the extraction of new facts can be achieved using knowledge-based trust [5] (i.e., judging the quality of a set of extracted triples by their overlap with the knowledge base). Zhang et al. [35] present an approach for detection of novel entities in tables. They first exploit lexical and semantic similarity for entity linking and column heading property matching. In a second step they use the output to detect novel entities in table columns. Oulabi and Bizer [21] tackle the same problem for Wikipedia tables with a bootstrapping approach based on expert-defined rules. Macdonald and Barbosa [17] extract new facts from Wikipedia tables to extend the Freebase knowledge base. With an LSTM that uses contextual information of the table, they extract new facts for 28 relations.

Lists have only very sparsely been used for knowledge graph completion. Paulheim and Ponzetto [24] frame the general potential of list pages as a source of knowledge in Wikipedia. They propose to use a combination of statistical and NLP methods to extract knowledge and show that, by applying them to a single list page, they are able to extract a thousand new statements.

Compared to all previously mentioned approaches, we take an abstract view on listings by considering only their subject entities. This provides the advantage that rules can be learned from and applied to arbitrary listings. In addition to that, we do not only discover novel entities, but also discover relations between those entities and the page subject.

In our previous work [11], we have already presented an approach for the identification of novel entities and the extraction of facts in Wikipedia list pages. List pages are pages in Wikipedia that start with *List of* and contain listings (i.e., tables or lists) of entities for a given topic (e.g. *List of Japanese speculative fiction writers*). The approach is divided into two phases: In a first phase, a dataset of tagged entities from list pages is extracted. With distant supervision from CaLiGraph, a knowledge graph with a detailed type hierarchy derived from Wikipedia categories and list pages, a part of the mentioned entities is heuristically labeled as subject entities and non-subject entities. In a second phase, the dataset is enriched with positional, lexical, and statistical features extracted from the list pages. On the basis of this data, an XGBoost classifier is able to identify more than two million subject entities with an average precision of 90%. As not all the information about the subject entities is contained in the knowledge graphs DBpedia and CaLiGraph, they can be enhanced with the missing information.

In this work, we reuse the approach presented in [11] for identifying subject entities. Further, as it is the only approach that also works with arbitrary listings, we use it as a baseline in our experiments. As, in its current state, it only works for list pages in Wikipedia, we extend it to arbitrary pages with a simple frequency-based approach.

2.2 Exploiting the Context of Listings

As tables are the more actively researched type of listings, we focus here on the types of context used when working with tables. The most obvious source of context is found directly on the page where the table is located. This page context is, for example, used by InfoGather [34] to detect possible synonyms in table headers for means of table matching.

Zhang [38] distinguishes between "in-table" features like the table header, and "out-table" features like captions, page title, and text of surrounding paragraphs. With both kinds of features, they perform entity disambiguation against Freebase.

The previously mentioned approach of Macdonald and Barbosa [17] focuses on tables in Wikipedia and hence uses specific context features like section titles, table headers and captions, and the text in the first paragraph of the table's section. Interestingly, they do not only discover relations between entities in the table, but also between a table entity and the page subject.

MENTOR [2] leverages patterns occurring in headers of Wikipedia tables to consistently discover DBpedia relations. Lehmberg et al. [15] tackle the problem of small web tables with table stitching, i.e., they combine several small tables with a similar context (e.g., same page or domain and a matching schema) into one large table, making it easier to extract facts from it.

Apart from page context, many approaches use the context of entities in tables to improve extraction results. Zhang et al. [37] generate new sub-classes to a taxonomy for a set of entities. Therefore, they find the best-describing class using the context of the entities. In particular, they use the categories of the entities as well as the immediate context around the entities on the page. Another approach that uses entity categories as context is TableNet [7]. They leverage the context to find schematically similar or related tables for a given table in Wikipedia.

In our experiments with Wikipedia, we use section headers as page context and types in the knowledge graph as entity context. However, the definition of context in our approach is kept very generic on purpose. By doing that, we are able to incorporate additional context sources like section text or entity categories to improve extraction results. This, however, also comes with an increase in rule complexity and, consequently, run time.

2.3 Rule-based Knowledge Graph Completion

Rule-based knowledge graph completion approaches typically generate rules either on instance-level (rules that add new facts for individual instances) or on schema-level (rules that add additional schematic constraints).

AMIE+ [8] and AnyBURL [18] are instance-level rule learners inspired by integer linear programming (ILP). The former uses top-down, the latter bottom-up rule learning to generate rules in the fashion of $born(X, A) \wedge capital(A, Y) \implies citizen(X, Y)$.

DL-Learner [13] is an ILP-based approach on schema-level which finds description logic patterns for a set of instances. A related approach uses statistical schema induction [30] to derive additional schema constraints (e.g. range restrictions for predicates).

The above mentioned approaches are merely *link prediction* approaches, i.e. they predict new relations between entities already contained in the knowledge graph. The same holds for the omnipresent knowledge graph embedding approaches [32]. Such approaches are very productive when enough training data is available and they provide exact results especially when both positive and negative examples are given. In the setting of this paper, we are working with (more or less) noisy external data.

With regard to instance- versus schema-level, our approach can be regarded as a hybrid approach that generates rules for sets of entities, which are in turn used to generate facts on an instance-level. In this respect, our approach is similar to C-DF [33] which uses Wikipedia categories as an external data source to derive the characteristics of categories. To that end, they derive lexical patterns from category names and contained entities.

In this paper, we apply rule learning to co-occurring entities in Wikipedia. While existing approaches have only considered explicit co-occurrence, i.e., categories or list pages, we go beyond the state of the art by considering *arbitrary* listings in Wikipedia, as the one shown in Fig. 1.

3 INFORMATION EXTRACTION FROM CO-OCCURRENCES

In this paper, we consider a data corpus D from which co-occurring entities can be extracted (e.g., listings in Wikipedia or a collection of spreadsheets). Furthermore, we assume that a knowledge graph which contains a subset of those entities can be extended with information learned about the co-occurring entities.

3.1 Task Formulation

The Knowledge Graph $\mathcal{K}$ is a set of assertions about its entities in the form of triples $\{(s, p, o) | s \in \mathcal{E}, p \in \mathcal{P}, o \in \mathcal{E} \cup \mathcal{T} \cup \mathcal{L}\}$ defined over sets of entities $\mathcal{E}$, predicates $\mathcal{P}$, types $\mathcal{T}$, and literals $\mathcal{L}$. We refer to statements about the types of an entity (i.e., p = rdf:type, $o \in \mathcal{T}$) as type assertions ($TA \subset K$), and to statements about relations between two entities (i.e., $o \in \mathcal{E}$) as relation assertions ($RA \subset K$). With $\mathcal{K}^* \supseteq \mathcal{K}$, we refer to the idealized complete version of $\mathcal{K}$. With regard to the OWA this means that a fact is incorrect if it is not contained in $\mathcal{K}^*$.[6]

The data corpus D contains a set of listings Φ, where each listing $\phi \in \Phi$ contains a number of subject entities SE_ϕ. Our task is to identify statements that hold for all subject entities SE_ϕ in a listing ϕ. We distinguish taxonomic and relational information that is expressed in $\mathcal{K}$.

The taxonomic information is a set of types that is shared by all SE of a listing:

$$\mathcal{T}_\phi = \{t | t \in \mathcal{T}, \forall s \in SE_\phi : (s, \texttt{rdf:type}, t) \in \mathcal{K}^*\}, \tag{3}$$

and the relational information is a set of relations to other entities which is shared by all SE of a listing:[7]

$$\mathcal{R}_\phi = \{(p, o) | p \in \mathcal{P} \cup \mathcal{P}^{-1}, o \in \mathcal{E}, \forall s \in SE_\phi : (s, p, o) \in \mathcal{K}^*\}. \tag{4}$$

From these characteristics of listings, we can derive all the *additional* type assertions

$$TA^+ = \bigcup_{\phi \in \Phi} \{(s, \texttt{rdf:type}, t) | s \in SE_\phi, t \in \mathcal{T}_\phi\} \setminus TA \tag{5}$$

and *additional* relation assertions

$$RA^+ = \bigcup_{\phi \in \Phi} \{(s, p, o) | s \in SE_\phi, (p, o) \in \mathcal{R}_\phi\} \setminus RA \tag{6}$$

that are encoded in Φ and missing in $\mathcal{K}$. Furthermore, TA^+ and RA^+ can contain additional entities that are not yet contained in $\mathcal{K}$, as there is no restriction for subject entities of Φ to be part of $\mathcal{K}$.

For the sake of readability, we will only describe the case of $\mathcal{R}_\phi$ for the remainder of this section as $\mathcal{T}_\phi$ is – notation-wise – a special case of $\mathcal{R}_\phi$ with $p = \texttt{rdf:type}$ and $o \in \mathcal{T}$.

3.2 Learning Descriptive Rules for Listings

Due to the incompleteness of $\mathcal{K}$, it is not possible to derive the exact set of relations $\mathcal{R}_\phi$ for every listing in Φ. Hence, our goal is to derive an approximate version $\hat{\mathcal{R}}_\phi$ by using ϕ and the knowledge about SE_ϕ in $\mathcal{K}$.

Similar to the rule learner AMIE+ [8], we use the partial completeness assumption (PCA) to generate negative evidence. The PCA implies that if $(s, p, o) \in \mathcal{K}$ then $\forall o' : (s, p, o') \in \mathcal{K}^* \implies (s, p, o') \in \mathcal{K}$. In order words, if $\mathcal{K}$ makes some assertions with a predicate p for a subject s, then we assume that $\mathcal{K}$ contains every p-related information about s.

Following from the PCA, we use the *count* of entities with a specific predicate-object combination in a set of entities E

$$count(E, p, o) = |\{s | s \in E, \exists o : (s, p, o) \in \mathcal{K}\}| \tag{7}$$

and the *count* of entities having predicate p with an arbitrary object

$$count(E, p) = |\{s | s \in E, \exists o' : (s, p, o') \in \mathcal{K}\}| \tag{8}$$

to compute a maximum-likelihood-based frequency of a specific predicate-object combination occurring in E:

$$freq(E, p, o) = \frac{count(E, p, o)}{count(E, p)}. \tag{9}$$

From Eq. 9 we first derive a naive approximation of a listing's relations by including all relations with a frequency above a defined threshold τ_{freq}:

$$\hat{\mathcal{R}}_\phi^{freq} = \{(p, o) | (p, o) \in \mathcal{R}, freq(SE_\phi, p, o) > \tau_{freq}\}. \tag{10}$$

[6] $\mathcal{K}^*$ is merely a theoretical construct, since a complete knowledge graph of all entities in the world cannot exist.

[7] Here, the entities in SE_ϕ may occur both in the subject as well as in the object position. But for a more concise notation, we use only (p,o)-tuples and introduce the set of inverse predicates $\mathcal{P}^{-1}$ to express that SE may also occur in object position. This is, however, only a notation and the inverse predicates do not have to exist in the schema.

Table 1: Exemplary context (ζ), type frequency (T^F), and relation frequency (R^F) vectors for a set of listings extracted from D. While ζ is extracted directly from D, T^F and R^F are retrieved via distant supervision from $\mathcal{K}$.

Listing	ζ	T^F	R^F
ϕ_1	(1 0 1 ... 1)	(0.2 0.9 0.0 ... 0.1)	(0.9 0.1 0.0 ... 0.1)
ϕ_2	(0 1 1 ... 0)	(0.0 0.2 0.0 ... 0.9)	(0.0 0.0 0.0 ... 0.2)
ϕ_3	(0 0 0 ... 0)	(0.7 0.7 0.0 ... 0.0)	(0.0 0.0 0.0 ... 0.4)
		...	
ϕ_{n-1}	(1 0 0 ... 1)	(0.8 0.9 0.0 ... 0.0)	(0.0 0.9 0.0 ... 0.0)
ϕ_n	(1 0 0 ... 1)	(0.7 1.0 0.0 ... 0.3)	(0.0 0.0 0.8 ... 0.0)

As argued in Sec. 1.1, we improve this naive frequency-based approximation by learning more general patterns that describe the characteristics of listings using their context.

Hypothesis 1. The context ζ_ϕ of a listing ϕ in D contains such information about $\mathcal{R}_\phi$ that it can be used to find subsets of Φ with similar $\mathcal{R}$.

Let Table 1 contain the information about all listings in D. A listing ϕ is defined by its context ζ_ϕ (which can in theory contain any information about ϕ, from the title of its section to an actual image of the listing), the type frequencies $(t_1, t_2, .., t_x) \in T^F_\phi$, and the relation frequencies $(r_1, r_2, .., r_y) \in R^F_\phi$. Listings ϕ_1, ϕ_{n-1}, and ϕ_n have overlapping context vectors. t_2 has a consistently high frequency over all three listings. It is thus a potential type characteristic for this kind of listing context. Furthermore, r_1 has a high frequency in ϕ_1, r_2 in ϕ_{n-1}, and r_3 in ϕ_n – if the three relations share the same predicate, they may all express a similar relation to an entity in their context (e.g. to the subject of the page).

In a concrete scenario, the context vector *(1 0 0 ... 1)* might indicate that the listing is located on the page of a musician under the section *Solo albums*. t_2 holds the frequency of the type *Album* in this listing and r_1 to r_3 describe the frequencies of the relations (*artist*, Gilby Clarke), (*artist*, Axl Rose), and (*artist*, Slash).

We formulate the task of discovering frequent co-occurrences of context elements and taxonomic and relational patterns as an association rule mining task over all listings in D. Association rules, as introduced by Agrawal et al. [1], are simple implication patterns originally developed for large and sparse datasets like transaction databases of supermarket chains. To discover items that are frequently bought together, rules of the form $X \implies Y$ are produced, with X and Y being itemsets. In the knowledge graph context, they have been used, e.g., for enriching the schema of a knowledge graph [23, 30].

For our scenario, we need a mapping from a context vector $\zeta \in Z$ to a predicate-object tuple. Hence, we define a rule r, its antecedent r_a, and its consequent r_c as follows:

$$r : r_a \in Z \implies r_c \in (\mathcal{P} \cup \mathcal{P}^{-1}) \times (\mathcal{T} \cup \mathcal{E} \cup \mathcal{X}). \quad (11)$$

As a rule should be able to imply relations to entities that vary with the context of a listing (e.g. to *Gilby Clarke* as the page's subject in Fig. 1), we introduce $\mathcal{X}$ as the set of placeholders for context entities (instead of *Gilby Clarke*, the object of the rule's consequent would be `<PageEntity>`).

We say a rule antecedent r_a matches a listing context ζ_ϕ (short: $r_a \simeq \zeta_\phi$) if the vector of ζ_ϕ is 1 when the vector of r_a is 1. In essence, ζ_ϕ must comprise r_a. Accordingly, we need to find a set of rules R, so that for every listing ϕ the set of approximate listing relations

$$\hat{\mathcal{R}}^{rule}_\phi = \bigcup_{r \in R} \{r_c | r_a \simeq \zeta_\phi\} \quad (12)$$

resembles the true relations $\mathcal{R}_\phi$ as closely as possible.

Considering all the listings in Fig. 1, their $\hat{\mathcal{R}}^{rule}_\phi$ should, among others, contain the rules[8, 9]

$$topSection(\text{"Discography"}) \implies (type, \texttt{MusicalWork}) \quad (13)$$

and

$$topSection(\text{"Discography"}) \implies (artist, \textit{<PageEntity>}). \quad (14)$$

It is important to note that these rules can be derived from listings with differing context vectors. All listings only have to have in common that their top section has the title *Discography* and that the contained entities are of the type `MusicalWork` with the page entity as artist. Still, the individual listings may, for example, occur in sections with different titles.

3.3 Quality Metrics for Rules

In original association rule mining, two metrics are typically considered to judge the quality of a rule $X \implies Y$: the support of the rule antecedent (how often does X occur in the dataset), and the confidence of the rule (how often does $X \cup Y$ occur in relation to X).

Transferring the support metric to our task, we count the absolute frequency of a particular context occurring in Φ. Let $\Phi_{r_a} = \{\phi | \phi \in \Phi, r_a \simeq \zeta_\phi\}$, then we define the support of the rule antecedent r_a as

$$supp(r_a) = |\Phi_{r_a}|. \quad (15)$$

Due to the incompleteness of $\mathcal{K}$, the values of Y are in our case no definitive items but maximum-likelihood estimates of types and relations. With respect to these estimates, a good rule has to fulfill two criteria: it has to be correct (i.e. frequent with respect to all SE of the covered listings) and it has to be consistent (i.e. consistently correct over all the covered listings).

We define the correctness, or confidence, of a rule as the frequency of the rule consequent over all SE of a rule's covered listings:

$$conf(r) = \frac{\sum_{\phi \in \Phi_{r_a}} count(SE_\phi, p_{r_c}, o_{r_c})}{\sum_{\phi \in \Phi_{r_a}} count(SE_\phi, p_{r_c})}, \quad (16)$$

and we define the consistency of a rule using the mean absolute deviation of an individual listing's confidence to the overall confidence of the rule:

$$cons(r) = 1 - \frac{\sum_{\phi \in \Phi_{r_a}} |freq(SE_\phi, p_{r_c}, o_{r_c}) - conf(r)|}{supp(r_a)}. \quad (17)$$

While a high confidence ensures that the overall assertions generated by the rule are correct, a high consistency ensures that few listings with many SE do not outvote the remaining covered listings.

[8] Note that Eqs. 1 and 2 are the axiom equivalents of Eqs. 13 and 14. For better readability, we use the description logics notation of Eqs. 1 and 2 from here on.

[9] Instead of a binary vector, we use a more expressive notation for the listing context in our examples. The notations are trivially convertible by one-hot-encoding.

To select an appropriate set of rules R from all the candidate rules R^* in the search space, we have to pick reasonable thresholds for the minimum support (τ_{supp}), the minimum confidence (τ_{conf}), and the minimum consistency (τ_{cons}). By applying these thresholds, we find our final set of descriptive rules R:

$$\{r|r \in R^*, supp(r_a) > \tau_{supp} \land conf(r) > \tau_{conf} \land cons(r) > \tau_{cons}\}. \tag{18}$$

Typically, the choice of these thresholds is strongly influenced by the nature of the dataset D and the extraction goal (correctness versus coverage).

4 EXPLOITING CO-OCCURRENCES IN WIKIPEDIA

Wikipedia is a rich source of listings, both in dedicated list pages as well as in sections of article pages. Hence, we use it as a data corpus for our experiments. In Sec. 6, we discuss other appropriate corpora for our approach.

Due to its structured and encyclopedic nature, Wikipedia is a perfect application scenario for our approach. We can exploit the structure by building very expressive context vectors. Obviously, this positively influences the quality of extraction results. Still, the definition of the context vector is kept abstract on purpose to make the approach applicable to other kinds of web resource as well. However, an empirical evaluation of the practicability or performance of the approach for resources outside of the encyclopedic domain is out of scope of this paper.

4.1 Approach Overview

Fig. 2 gives an overview of our extraction approach. The input of the approach is a dump of Wikipedia as well as an associated knowledge graph. In the *Subject Entity Discovery* phase, listings and their context are extracted from the Wikipedia dump and subject entities are identified (Sec. 4.3). Subsequently, the existing information in the knowledge graph is used to mine descriptive rules from the extracted listings (Sec. 4.4). Finally, the rules are applied to all the listings in Wikipedia in order to extract new type and relation assertions (Sec. 4.5).

4.2 Wikipedia as a Data Corpus

We pick Wikipedia as a data corpus for our experiments as it brings several advantages:

Structure. Wikipedia is written in an entity-centric style with a focus on facts. Listings are often used to provide an overview of a set of entities that are related to the main entity. Due to the encyclopedic style and the peer-reviewing process, it has a consistent structure. Especially section titles are used consistently for specific topics. Wikipedia has its own markup language (Wiki markup), which allows a more consistent access to interesting page structures like listings and tables than plain HTML.

Entity Links. If a Wikipedia article is mentioned in another article, it is typically linked in the Wiki markup (a so called *blue link*). Furthermore, it is possible to link to an article that does not (yet) exist (a so called *red link*). As Wikipedia articles can be trivially mapped to entities in Wikipedia-based knowledge graphs like DBpedia, since they create one entity per article, we can identify many named entities in listings and their context without the help of an entity linker.

For our experiments, we use a Wikipedia dump of October 2016 which is, at the time of the experiments, the most recent dump that is compatible with both DBpedia and CaLiGraph. In this version, Wikipedia contains 6.9M articles, 2.4M of which contain listings with at least two rows.[10] In total, there are 5.1M listings with a row count median of 8, mean of 21.9, and standard deviation of 76.8. Of these listings, 1.1M are tables, and 4.0M are lists.

4.3 Subject Entity Discovery

4.3.1 *Entity Tagging.*

Apart from the already tagged entities via blue and red links, we have to make sure that any other named entity in listings and their context is identified as well. This is done in two steps:

In a first step, we expand all the blue and red links in an article. If a piece of text is linked to another article, we make sure that every occurrence of that piece of text in the article is linked to the other article. This is necessary as by convention other articles are only linked at their first occurrence in the text.[11]

In a second step, we use a named entity tagger to identify additional named entities in listings. To that end, we use a state-of-the-art entity tagger from spaCy.[12] This tagger is trained on the OntoNotes5[13] corpus, and thus not specifically trained to identify named entities in short text snippets like they occur in listings. Therefore, we specialize the tagger by providing it Wikipedia listings as additional training data with blue links as positive examples. In detail, the tagger is specialized as follows:

- We retrieve all listings in Wikipedia list pages as training data.
- We apply the plain spaCy entity tagger to the listings to get named entity tags for all mentioned entities.
- To make these tags more consistent, we use information from DBpedia about the tagged entities: We look at the distribution of named entity tags over entities with respect to their DBpedia types and take the majority vote. For example, if 80% of entities with the DBpedia type `Person` are annotated with the tag *PERSON*, we use *PERSON* as label for all these entities.
- Using these consistent named entity tags for blue-link entities, we specialize the spaCy tagger.

4.3.2 *Subject Entity Classification.*

We apply the approach from [11] for the identification of subject entities in listings. In short, we use lexical, positional, and statistical features to classify entities as subject or non-subject entities (refer to Sec. 2.1 for more details). Despite being developed only for listings in list pages, the classifier is applicable to any kind of listing in Wikipedia. A disadvantage of this broader application is that the classifier is not trained in such a way that it ignores listings used for organisational or design purposes (e.g. summaries or timelines). These have to be filtered out in the subsequent stages.

[10] Wiki markup is parsed with WikiTextParser: https://github.com/5j9/wikitextparser.

[11] https://en.wikipedia.org/wiki/Wikipedia:Manual_of_Style/Linking#Duplicate_and_repeat_links

[12] https://spacy.io

[13] https://catalog.ldc.upenn.edu/LDC2013T19

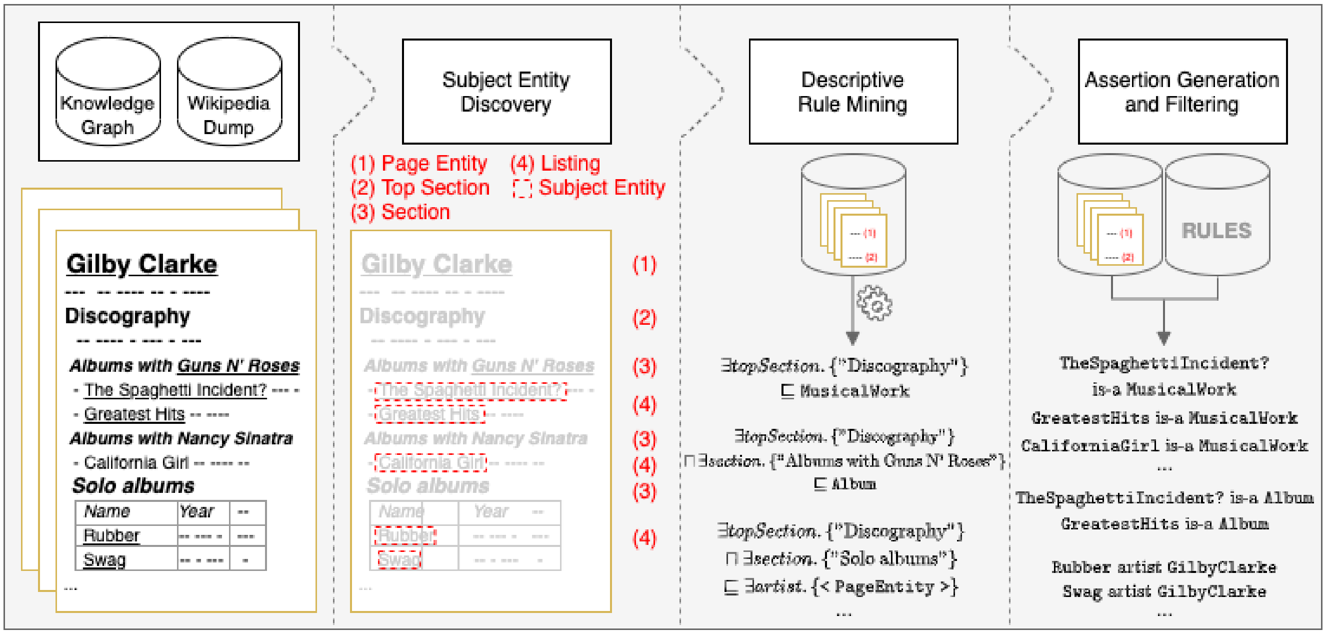

Knowledge Graph
Wikipedia Dump
Gilby Clarke
Discography
Albums with Guns N' Roses
The Spaghetti Incident?
Greatest Hits
Albums with Nancy Sinatra
California Girl
Solo albums
Name
Year
Rubber
Swag
...
Subject Entity Discovery
(1) Page Entity
(2) Top Section
(3) Section
(4) Listing
Subject Entity
(1)
(2)
(3)
(4)
Descriptive Rule Mining
∃topSection.{"Discography"} ⊑ MusicalWork
∃topSection.{"Discography"} ⊓ ∃section.{"Albums with Guns N' Roses"} ⊑ Album
∃topSection.{"Discography"} ⊓ ∃section.{"Solo albums"} ⊑ ∃artist.{< PageEntity >}
...
Assertion Generation and Filtering
RULES
TheSpaghettiIncident? is-a MusicalWork
GreatestHits is-a MusicalWork
CaliforniaGirl is-a MusicalWork
...
TheSpaghettiIncident? is-a Album
GreatestHits is-a Album
Rubber artist GilbyClarke
Swag artist GilbyClarke
...

Figure 2: An overview of the approach with exemplary outputs of the individual phases.

4.3.3 Results. After expanding all the blue and red links on the pages, the dataset contains 5.1M listings with 60.1M entity mentions. 51.6M additional entity mentions are identified by the named entity tagger.

Of all the entity mentions, we classify 25.8M as subject entities. Those occur in 2.5M listings of 1.3M pages. This results in a mean of 10.5 and median of 4 subject entities per listing with a standard deviation of 49.8.

4.4 Descriptive Rule Mining

4.4.1 Describing Listings. The search space for rule candidates is defined by the listing context. Thus, we choose the context in such a way that it is expressive enough to be an appropriate indicator for $\mathcal{T}_\phi$ and $\mathcal{R}_\phi$, and concise enough to explore the complete search space without any additional heuristics.

We exploit the fact that Wikipedia pages of a certain type (e.g., musicians) mostly follow naming conventions for the sections of their articles (e.g., albums and songs are listed under the top section *Discography*). Further, we exploit that the objects of the SE's relations are usually either the entity of the page, or an entity mentioned in a section title. We call these typical places for objects the relation *targets*. In Fig. 1, *Gilby Clarke* is an example of a *PageEntity* target, and *Guns N' Roses* as well as *Nancy Sinatra* are examples for *SectionEntity* targets. As a result, we use the type of the page entity, the top section title, and the section title as listing context.

Additionally, we use the type of entities that are mentioned in section titles. This enables the learning of more abstract rules, e.g., to distinguish between albums listed in a section describing a band:

$$\exists pageEntityType.\{\texttt{Person}\} \sqcap \exists topSection.\{\text{"Discography"}\} \\ \sqcap \exists sectionEntityType.\{\texttt{Band}\} \sqsubseteq \texttt{Album},$$

and songs listed in a section describing an album:

$$\exists pageEntityType.\{\texttt{Person}\} \sqcap \exists topSection.\{\text{"Discography"}\} \\ \sqcap \exists sectionEntityType.\{\texttt{Album}\} \sqsubseteq \texttt{Song}.$$

4.4.2 Threshold Selection. We want to pick the thresholds in such a way that we tolerate some errors and missing information in $\mathcal{K}$, but do not allow many over-generalized rules that create incorrect assertions. Our idea for a sensible threshold selection is based on two assumptions:

Assumption 1. Being based on a maximum-likelihood estimation, rule confidence and consistency roughly order rules by the degree of prior knowledge we have about them.

Assumption 2. Assertions generated by over-generalized rules contain substantially more random noise than assertions generated by good rules.

Assumption 1 implies that the number of over-generalized rules increases with the decrease of confidence and consistency. As a consequence, assumption 2 implies that the amount of random noise increases with decrease of confidence and consistency.

To measure the increase of noise in generated assertions, we implicitly rely on existing knowledge in $\mathcal{K}$ by using the named entity tags of subject entities as a proxy. This works as follows: For a subject entity e that is contained in $\mathcal{K}$, we have its type information $\mathcal{T}_e$ from $\mathcal{K}$ and we have its named entity tag ψ_e from our named entity tagger. Going over all SE of listings in Φ, we compute the probability of an entity with type t having the tag ψ by counting how often they co-occur:

$$tagprob(t,\psi) = \frac{|\{e|\exists\phi \in \Phi : e \in SE_\phi \land t \in \mathcal{T}_e \land \psi = \psi_e\}|}{|\{e|\exists\phi \in \Phi : e \in SE_\phi \land t \in \mathcal{T}_e\}|}. \quad (19)$$

For example, for the DBpedia type `Album`, we find the tag probabilities
WORK_OF_ART: 0.49, *ORG*: 0.14, *PRODUCT*: 0.13, *PERSON*: 0.07, showing that album titles are rather difficult to recognize. For the type `Person` and the tag *PERSON*, on the other hand, we find a probability of 0.86.

We can then compute the tag-based probability for a set of assertions *A* by averaging over the tag probability that is produced by the individual assertions. To compute this metric, we compare the tag of the assertion's subject entity with some kind of type information about it. This type information is either the asserted type (in case of a type assertion), or the domain of the predicate[14] (in case of a relation assertion):

$$tagfit(A) = \begin{cases} \frac{\sum_{(s,p,o)\in A} tagprob(o, \psi_s)}{|A|} & \text{if } p = \texttt{rdf:type} \\ \frac{\sum_{(s,p,o)\in A} tagprob(domain_p, \psi_s)}{|A|} & \text{otherwise.} \end{cases} \tag{20}$$

While we do not expect the named entity tags to be perfect, our approach is based on the idea that the tags are consistent to a large extent. By comparing the $tagfit$ of assertions produced by rules with varying levels of confidence and consistency, we expect to see a clear decline as soon as too many noisy assertions are added.

4.4.3 Results. Fig. 3 shows the $tagfit$ for type and relation assertions generated with varying levels of rule confidence and consistency. Our selection of thresholds is indicated by blue bars, i.e. we set the thresholds to the points where the $tagfit$ has its steepest drop. The thresholds are picked conservatively to select only high-quality rules by selecting points before an accelerated decrease of cumulative $tagfit$. But more coverage-oriented selections are also possible. In Fig. 3d, for example, a threshold of *0.75* is also a valid option.

An analysis of rules with different levels of confidence and consistency has shown that a minimum support for types is not necessary. For relations, a support threshold of 2 is helpful to discard over-generalized rules. Further, we found that it is acceptable to pick the thresholds independently from each other, as the turning points for a given metric don't vary significantly when varying the remaining metrics.

Applying these thresholds, we find an overall number of 5,294,921 type rules with 369,139 distinct contexts and 244,642 distinct types. Further, we find 3,028 relation rules with 2,602 distinct contexts and 516 distinct relations. 949 of the relation rules have the page entity as target, and 2,079 have a section entity as target.

Among those rules are straightforward ones like

$$\exists pageEntityType.\{\texttt{Person}\} \sqcap \exists topSection.\{\text{"Acting filmography"}\} \sqsubseteq \exists actor.\{<PageEntity>\},$$

and more specific ones like

$$\exists pageEntityType.\{\texttt{Location}\} \sqcap \exists topSection.\{\text{"Media"}\} \sqcap \exists section.\{\text{"Newspapers"}\} \sqsubseteq \texttt{Periodical\_literature}.$$

4.5 Assertion Generation and Filtering

4.5.1 Assertion Generation. We apply the rules selected in the previous section to the complete dataset of listings to generate type and relation assertions. Subsequently, we remove any duplicate assertions and assertions that already exist in $\mathcal{K}$.

4.5.2 Tag-based Filtering. To get rid of errors introduced during the extraction process (e.g. due to incorrectly extracted subject entities or incorrect rules), we employ a final filtering step for the generated assertions: every assertion producing a $tagprob \leq \frac{1}{3}$ is discarded. The rationale behind the threshold is as follows: Types have typically one and sometimes two corresponding named entity tags (e.g. the tag *PERSON* for the DBpedia type `Person`, or the tags *ORG* and *FAC* for the type `School`). As tag probabilities are relative frequencies, we make sure that, with a threshold of $\frac{1}{3}$, at most two tags are accepted for any given type.

For the tag probabilities of type `Album` from Sec. 4.4.2, the only valid tag is *WORK_OF_ART*. As a consequence, any assertions of the form $(s, rdf{:}type, \texttt{Album})$ with s having a tag other than *WORK_OF_ART* are discarded.

4.5.3 Results. Tab. 2 shows the number of generated type and relation assertions before and after the tag-based filtering. The number of inferred types are listed separately for DBpedia and CaLiGraph. For relations, we show two kinds: The entry *Relations* lists the number of extracted assertions from rules. As DBpedia and CaLiGraph share the same set of predicates, these assertions are applicable to both graphs. Furthermore, as *Relations (via CaLiGraph)*, we list the number of relations that can be inferred from the extracted CaLiGraph types via restrictions in the CaLiGraph ontology. CaLiGraph contains more than 300k of such restrictions that imply a relation based on a certain type. For example, the ontology contains the value restriction

$$\texttt{Pop\_rock\_song} \sqsubseteq \exists genre.\{\text{Pop music}\}.$$

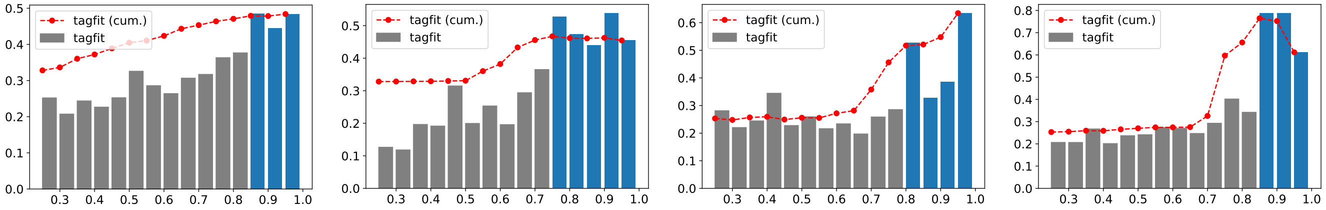
As we extract the type `Pop_rock_song` for the Beach Boys song *At My Window*, we infer the fact (At My Window, *genre*, Pop music).

For CaLiGraph, we find assertions for 3.5M distinct subject entities with 3M of them not contained in the graph. For DBpedia, we find assertions for 3.1M distinct subject entities with 2.9M of them not contained. The unknown subject entities are, however, not disambiguated yet. Having only small text snippets in listings as information about these entities, a disambiguation with general-purpose disambiguation approaches [39] is not practical. We thus leave this as an own research topic for future work. For an estimation of the actual number of novel entities, we rely on previous work [11], where we analyzed the overlap for red links in list pages. In that paper, we estimate an overlap factor of 1.07 which would – when applied to our scenario – reduce the number of actual novel entities to roughly 2.8M for CaLiGraph and 2.7M for DBpedia. In relation to the current size of those graphs, this would be an increase of up to 38% and 54%, respectively [9].

5 EVALUATION

In our performance evaluation, we judge the quality of generated assertions from our rule-based approach. As a baseline, we additionally evaluate assertions generated by the frequency-based

[14] We use the domain of the predicate p as defined in $\mathcal{K}$. In case of $p \in \mathcal{P}^{-1}$, we use the range of the original predicate.



(a) Type confidence **(b) Type consistency** **(c) Relation confidence** **(d) Relation consistency**

Figure 3: *tagfit* of assertions generated from rules in a specified confidence or consistency interval. Bars show scores for a given interval (e.g. *(0.75,0.80]*), lines show cumulative scores (e.g. *(0.75,1.00]*). Blue bars indicate the selected threshold.

Table 2: Number of generated assertions after removing existing assertions (Raw), and after applying tag-based filtering (Filtered).

Assertion Type	Raw	Filtered
Types (DBpedia)	11,459,047	7,721,039
Types (CaLiGraph)	47,249,624	29,128,677
Relations	732,820	542,018
Relations (via CaLiGraph)	1,381,075	796,910

Table 3: Correctness of manually evaluated assertions.

Assertion Type	#Dataset	#Samples	Correct [%]
Types (DBpedia)			
frequency-based	6,680,565	414	91.55 ± 2.68
rule-based	7,721,039	507	93.69 ± 2.12
Types (CaLiGraph)			
frequency-based	26,676,191	2,000	89.40 ± 1.23
rule-based	29,128,677	2,000	91.95 ± 1.19
Relations			
frequency-based	392,673	1,000	93.80 ± 1.49
rule-based	542,018	1,000	95.90 ± 1.23

approach (see Eq. 10). For the latter, we use a threshold comparable to our rule-based approach (i.e., we set τ_{freq} to τ_{conf} and disregard listings with less than three subject entities).

5.1 Evaluation Procedure

The evaluated assertions are created with a stratified random sampling strategy. The assertions are thus distributed proportionally over all page types (like `Person` or `Place`) and sampled randomly within these.

The labeling of the assertions is performed by the authors with the procedure as follows: For a given assertion, first the page of the listing is inspected, then – if necessary and available – the page of the subject entity. If a decision cannot be made based on this information, a search engine is used to evaluate the assertion. Samples of the rule-based and frequency-based approaches are evaluated together and in random order to ensure objectivity.

Tab. 3 shows the results of the performance evaluation. In total, we evaluated 2,000 examples per approach for types and 1,000 examples per approach for relations. The taxonomy of CaLiGraph comprises the one of DBpedia. Thus, we evaluated the full sample for CaLiGraph types and report the numbers for both graphs, which is the reason why the sample size for DBpedia is lower. For relations, we only evaluate the ones that are generated directly from rules and not the ones inferred from CaLiGraph types, as the correctness of the inferred relations directly depends on the correctness of CaLiGraph types.

5.2 Type and Relation Extraction

The evaluation results in Tab. 3 show that the information extracted from listings in Wikipedia is of an overall high quality. The rule-based approach yields a larger number of assertions with a higher correctness for both types and relations.

For both approaches, the correctness of the extracted assertions is substantially higher for DBpedia. The reason for that lies in the differing granularity of knowledge graph taxonomies. DBpedia has 764 different types while CaLiGraph has 755,441 with most of them being more specific extensions of DBpedia types. For example, DBpedia might describe a person as `Athlete`, while CaLiGraph describes it as `Olympic_field_hockey_player_of_South_Korea`. The average depth of predicted types is 2.06 for the former and 3.32 for the latter.

While the asserted types are very diverse (the most predicted type is `Agent` with 7.5%), asserted relations are dominated by the predicate *genus* with 69.8% followed by *isPartOf* (4.4%) and *artist* (3.2%). This divergence cannot be explained with a different coverage: In DBpedia, 72% of entities with type `Species` have a *genus*, and 69% of entities with type `MusicalWork` have an *artist*. But we identify two other influencing factors: Wikipedia has very specific guidelines for editing species, especially with regard to standardization and formatting rules.[15] In addition to that, the *genus* relation is functional and hence trivially fulfilling the PCA. As our approach is strongly relying on this assumption and it potentially inhibits the mining of practical rules for non-functional predicates (like, for example, for *artist*), we plan on investigating this relationship further.

The inferred relations from CaLiGraph types are not evaluated explicitly. However, based on the correctness of restrictions in CaLiGraph that is reported to be 95.6% [10] and from the correctness

[15] https://species.wikimedia.org/wiki/Help:General_Wikispecies

Table 4: Error types partitioned by cause. The occurrence values are given as their relative frequency (per 100) in the samples evaluated in Tab. 3.

Error type	Type	Relation
(1) Entity parsed incorrectly	2.6	0.2
(2) Wrong subject entity identified	1.4	1.6
(3) Rule applied incorrectly	3.7	2.3
(4) Semantics of listing too complex	0.3	0.0

of type assertions, we estimate the correctness of the resulting relation assertions to be around 85.5% for the frequency-based and around 87.9% for the rule-based approach.

5.3 Novel Entity Discovery

For CaLiGraph, the frequency-based approach finds assertions for 2.5M distinct subject entities (2.1M of them novel). While the rule-based approach finds 9% more assertions, its assertions are distributed over 40% more entities (and over 43% more novel entities). This demonstrates the capabilities of the rule-based approach to apply contextual patterns to environments where information about actual entities is sparse.

Further, we analyzed the portion of evaluated samples that applies to novel entities and found that the correctness of these statements is slightly better (between 0.1% and 0.6%) than the overall correctness. Including CaLiGraph types, we find an average of 9.03 assertions per novel entity, with a median of 7. This is, again, due to the very fine-grained type system of CaLiGraph. For example, for the rapper *Dizzle Don*, which is a novel entity, we find 8 types (from `Agent` over `Musician` to `American_rapper`) and 4 relations: (*occupation*, Singing), (*occupation*, Rapping), (*birthPlace*, United States), and (*genre*, Hip hop music).

5.4 Error Analysis

With Tab. 4, we provide an analysis of error type frequencies for the rule-based approach on the basis of the evaluated sample. (1) is caused by the entity linker, mostly due to incorrect entity borders. For example, the tagger identifies only a part of an album title. (2) is caused by errors of the subject entity identification approach, e.g. when the approach identifies the wrong column of a table as the one that holds subject entities. (3) can have multiple reasons, but most often the applied rule is over-generalized (e.g. implying `Football_player` when the listing is actually about athletes in general) or applied to the wrong listing (i.e., the context described by the rule is not expressive enough). Finally, (4) happens, for example, when a table holds the specifications of a camera as this cannot be expressed with the given set of predicates in DBpedia or CaLiGraph.

Overall, most of the errors are produced by incorrectly applied rules. This is, however, unavoidable to a certain extent as knowledge graphs are not error-free and the data corpus is not perfect. A substantial portion of errors is also caused by incorrectly parsed or identified subject entities. Reducing these errors can also have a positive impact on the generated rules as correct information about entities is a requirement for correct rules.

6 DISCUSSION AND OUTLOOK

In this work, we demonstrate the potential of exploiting co-occurring similar entities for information extraction, and especially for the discovery of novel entities. We show that it is possible to mine expressive descriptive rules for listings in Wikipedia which can be used to extract information about millions of novel entities.

To improve our approach, we are investigating more sophisticated filtering approaches for the generated assertions to reduce the margin from raw to filtered assertions (see Tab. 2). Furthermore, we are experimenting with more expressive rules (e.g. by including additional context like substring patterns or section text) to improve our Wikipedia-based approach.

At the moment, we extract entities from single pages. While entity disambiguation on single pages is quite simple (on a single Wikipedia page, it is unlikely that the same surface form refers to different entities), the disambiguation of entities across pages is a much more challenging problem. Here, entity matching across pages is required, which should, ideally, combine signals from the source pages as well as constraints from the underlying ontology.

Furthermore, we work towards applying our approach to additional data corpora. Since the only language-dependent ingredient of our approach is the named entity tagging, and the entity tagger we use in our experiments has models for various languages,[16] our approach can also be extended to various language editions of Wikipedia.

Besides Wikipedia, we want to apply the approach to wikis in the Fandom[17] universe containing more than 380k wikis on various domains (among them many interesting wikis for our approach, like for example WikiLists[18]). For background knowledge, we plan to rely on existing knowledge graphs in this domain like DBkWik [12] or TiFi [3]. In the longer term, we want to extend the applicability of the approach towards arbitrary web pages, using microdata and RDFa annotations [20] as hooks for background knowledge.

REFERENCES

[1] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. 1993. Mining association rules between sets of items in large databases. In *1993 ACM SIGMOD international conference on Management of data*. 207–216.

[2] Matteo Cannaviccio, Lorenzo Ariemma, Denilson Barbosa, and Paolo Merialdo. 2018. Leveraging Wikipedia table schemas for knowledge graph augmentation. In *21st International Workshop on the Web and Databases*. 1–6.

[3] Cuong Xuan Chu, Simon Razniewski, and Gerhard Weikum. 2019. TiFi: Taxonomy Induction for Fictional Domains. In *The World Wide Web Conference*. 2673–2679.

[4] Luciano Del Corro and Rainer Gemulla. 2013. Clausie: clause-based open information extraction. In *The World Wide Web Conference*. 355–366.

[5] Xin Luna Dong, Evgeniy Gabrilovich, Kevin Murphy, Van Dang, Wilko Horn, Camillo Lugaresi, Shaohua Sun, and Wei Zhang. 2015. Knowledge-Based Trust: Estimating the Trustworthiness of Web Sources. *VLDB Endowment* 8, 9 (2015), 938–949.

[6] Michael Färber, Achim Rettinger, and Boulos El Asmar. 2016. On emerging entity detection. In *European Knowledge Acquisition Workshop*. Springer, 223–238.

[7] Besnik Fetahu, Avishek Anand, and Maria Koutraki. 2019. TableNet: An approach for determining fine-grained relations for Wikipedia tables. In *The World Wide Web Conference*. 2736–2742.

[8] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M Suchanek. 2015. Fast rule mining in ontological knowledge bases with AMIE+. *The VLDB Journal* 24, 6 (2015), 707–730.

[9] Nicolas Heist, Sven Hertling, Daniel Ringler, and Heiko Paulheim. 2020. Knowledge Graphs on the Web–an Overview. *Studies on the Semantic Web* 47 (2020), 3–22.

[16] https://spacy.io/models

[17] https://www.fandom.com/

[18] https://list.fandom.com/wiki/Main_Page

[10] Nicolas Heist and Heiko Paulheim. 2019. Uncovering the Semantics of Wikipedia Categories. In *International Semantic Web Conference*. Springer, 219–236.

[11] Nicolas Heist and Heiko Paulheim. 2020. Entity Extraction from Wikipedia List Pages. In *Extended Semantic Web Conference*. Springer, 327–342.

[12] Sven Hertling and Heiko Paulheim. 2020. DBkWik: extracting and integrating knowledge from thousands of wikis. *Knowledge and Information Systems* 62, 6 (2020), 2169–2190.

[13] Jens Lehmann. 2009. DL-Learner: learning concepts in description logics. *The Journal of Machine Learning Research* 10 (2009), 2639–2642.

[14] Jens Lehmann et al. 2015. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* 6, 2 (2015), 167–195.

[15] Oliver Lehmberg and Christian Bizer. 2017. Stitching web tables for improving matching quality. *VLDB Endowment* 10, 11 (2017), 1502–1513.

[16] Guiliang Liu, Xu Li, Jiakang Wang, Mingming Sun, and Ping Li. 2020. Extracting Knowledge from Web Text with Monte Carlo Tree Search. In *The Web Conference 2020*. 2585–2591.

[17] Erin Macdonald and Denilson Barbosa. 2020. Neural Relation Extraction on Wikipedia Tables for Augmenting Knowledge Graphs. In *29th ACM International Conference on Information & Knowledge Management*. 2133–2136.

[18] Christian Meilicke, Melisachew Wudage Chekol, Daniel Ruffinelli, and Heiner Stuckenschmidt. 2019. Anytime Bottom-Up Rule Learning for Knowledge Graph Completion.. In *28th International Joint Conference on Artificial Intelligence*. 3137–3143.

[19] Pablo N Mendes et al. 2011. DBpedia spotlight: shedding light on the web of documents. In *7th international conference on semantic systems*. 1–8.

[20] Robert Meusel, Petar Petrovski, and Christian Bizer. 2014. The webdatacommons microdata, rdfa and microformat dataset series. In *International Semantic Web Conference*. Springer, 277–292.

[21] Yaser Oulabi and Christian Bizer. 2019. Using weak supervision to identify long-tail entities for knowledge base completion. In *International Conference on Semantic Systems*. Springer, 83–98.

[22] Heiko Paulheim. 2017. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web* 8, 3 (2017), 489–508.

[23] Heiko Paulheim and Johannes Fümkranz. 2012. Unsupervised generation of data mining features from linked open data. In *2nd international conference on web intelligence, mining and semantics*. 1–12.

[24] Heiko Paulheim and Simone Paolo Ponzetto. 2013. Extending DBpedia with Wikipedia List Pages. In *1st International Workshop on NLP and DBpedia*, Vol. 1064. CEUR Workshop Proceedings, 85–90.

[25] Dominique Ritze, Oliver Lehmberg, Yaser Oulabi, and Christian Bizer. 2016. Profiling the potential of web tables for augmenting cross-domain knowledge bases. In *The World Wide Web Conference*. 251–261.

[26] Ahmad Sakor, Kuldeep Singh, Anery Patel, and Maria-Esther Vidal. 2020. Falcon 2.0: An entity and relation linking tool over Wikidata. In *29th ACM International Conference on Information & Knowledge Management*. 3141–3148.

[27] Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. 2018. Supervised open information extraction. In *2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 885–895.

[28] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *The World Wide Web Conference*. 697–706.

[29] Marieke Van Erp, Pablo Mendes, Heiko Paulheim, Filip Ilievski, Julien Plu, Giuseppe Rizzo, and Jörg Waitelonis. 2016. Evaluating entity linking: An analysis of current benchmark datasets and a roadmap for doing a better job. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. 4373–4379.

[30] Johanna Völker and Mathias Niepert. 2011. Statistical schema induction. In *Extended Semantic Web Conference*. Springer, 124–138.

[31] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85.

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

[33] Bo Xu, Chenhao Xie, Yi Zhang, Yanghua Xiao, Haixun Wang, and Wei Wang. 2016. Learning defining features for categories. In *25th International Joint Conference on Artificial Intelligence*. 3924–3930.

[34] Mohamed Yakout, Kris Ganjam, Kaushik Chakrabarti, and Surajit Chaudhuri. 2012. InfoGather: entity augmentation and attribute discovery by holistic matching with web tables. In *2012 ACM SIGMOD International Conference on Management of Data*. 97–108.

[35] Shuo Zhang et al. 2020. Novel Entity Discovery from Web Tables. In *The Web Conference 2020*. 1298–1308.

[36] Shuo Zhang and Krisztian Balog. 2020. Web Table Extraction, Retrieval, and Augmentation: A Survey. *ACM Transactions on Intelligent Systems and Technology (TIST)* 11, 2 (2020), 1–35.

[37] Shuo Zhang, Krisztian Balog, and Jamie Callan. 2020. Generating Categories for Sets of Entities. In *29th ACM International Conference on Information & Knowledge Management*. 1833–1842.

[38] Ziqi Zhang. 2014. Towards efficient and effective semantic table interpretation. In *International Semantic Web Conference*. Springer, 487–502.

[39] Ganggao Zhu and Carlos A Iglesias. 2018. Exploiting semantic similarity for named entity disambiguation in knowledge graphs. *Expert Systems with Applications* 101 (2018), 8–24.