Department of Information Engineering and Computer Science

Bachelor’s Degree in
Computer Science

FINAL DISSERTATION

soweego
solid catalogs and weekee go together

Supervisors
Andrea Passerini
Marco Fossati

Student
Massimo Frasson
mat. 180868

Academic Year 2017/2018
# Contents

1 Introduction .................................................. 5

2 Related work .................................................. 7

3 Preliminary analysis ........................................... 9
   3.1 Retrieving the dump ........................................ 9
      3.1.1 Problem ............................................ 9
      3.1.2 Solutions ........................................... 9
   3.2 Handling the format ....................................... 10
      3.2.1 Problem ............................................ 10
      3.2.2 Solution ........................................... 10
   3.3 Mapping to Wikidata ....................................... 10
      3.3.1 Solution ........................................... 10
      3.3.2 Side note: available entity metadata ................. 10
   3.4 Conclusion ............................................... 10

4 System architecture ........................................... 11
   4.1 Importer .................................................. 11
   4.2 Linker .................................................... 12
      4.2.1 Perfect name match .................................. 12
      4.2.2 Perfect name with birth and death dates match ...... 12
      4.2.3 Cross-database URL match ............................ 12
      4.2.4 Normalized full names match ......................... 13
   4.3 Ingestor .................................................. 13
   4.4 Validator .................................................. 13

5 Results ......................................................... 15

6 Conclusion ..................................................... 17

Bibliografia ..................................................... 19
Summary

Wikipedia and Wikidata are open source collaborative projects with the goal of collecting and sharing general knowledge. While the former is intended for humans, the latter caters for machine-readable data. Both are not meant to contain original research.\footnote{https://en.wikipedia.org/wiki/Wikipedia:No_original_research} Original research means that no reliable or published material exists referring to reported facts, allegations or ideas. Such design decision entails that Wikipedia and Wikidata content should be supported by references to external sources. Nevertheless, Wikidata suffers from a lack of references. The soeweego project aims at mitigating the issue by linking Wikidata to a set of trustworthy target catalogs, a task that can be cast as a record linkage problem. As a result, references and new content may be mined from these catalogs after the linking.

Record linkage\footnote{https://en.wikipedia.org/wiki/Record_linkage#Probabilistic_record_linkage} must cope with several challenges: inaccurate data, heterogeneous data precision, and ambiguity, just to name the key ones. These challenges can be addressed by comparing other data attributes, such as dates and locations. However, there is no guarantee that Wikidata and the target will share any attribute. soeweego addresses the issue by working on a common set of attributes, but for the sake of generalization it supports target-specific ones.

This thesis details a set of deterministic techniques for record linkage: perfect full name match; perfect name with birth and death dates match; perfect cross-catalog link match; tokenized cross catalog link match; normalized full names match. Specifically, we designed them to work on the common set of attributes, and we evaluate the results obtained picking the MusicBrainz as target database. We observe that URLs to external sources and dates attributes provide a relatively high precision. Hence, the output produced by these techniques has been added to Wikidata.

Our work on the aforementioned techniques represents the fundamental building blocks and knowledge to scale up from deterministic algorithms to probabilistic ones. We summarize our contributions to the soeweego project as follows.

- Analysis of the long tail of candidate target catalogs;
- development of the facility to import the MusicBrainz dump in the system;
- implementation of the baseline techniques for MusicBrainz;
- performance evaluation;
- software packaging in a portable environment;
- intervention to all technical discussions.

1 Introduction

In 2018, the English Wikipedia has been visited 207.83 billions of times, a 4.73% increase year by year.\footnote{https://stats.wikimedia.org/v2/#/en.wikipedia.org/reading/total-page-views/normal|bar|1-Year|~total} This shows that Wikipedia is one of the key tools for enhancing people knowledge, due to the easily accessible free information. However, the unstructured nature of its content does not enable straightforward machine processing. Wikidata\footnote{https://www.wikidata.org/} acts as central storage for structured data of its Wikimedia sister projects, including Wikipedia, Wikivoyage, Wikisource, and others.

Data quality is a crucial factor for the trustworthiness of Wikipedia and Wikidata content. In fact, Wikipedia provides mature citation guidelines to enforce high quality standard.\footnote{https://en.wikipedia.org/wiki/Wikipedia:Citing_sources} Likewise, Wikidata allows to store references to (ideally authoritative) sources along with its data.

Nevertheless, less than a quarter of the Wikidata knowledge base (KB) statements currently has a reference to non-wiki sources, and roughly an half of them is totally unreferenced.\footnote{https://docs.google.com/presentation/d/1XX-yzt98fg1AfFxHoixO11XCluwrS6foU1xjdZT9YI/edit?usp=sharing, slides 15 to 19} The problem can be alleviated in several ways: we could encourage the community to focus on the referencing task. Another option could be the alignment of Wikidata entities to a set of external databases, which can be automated by software. This is a particularly interesting option because it provides a repeatable process.

We define the problem as follows: given a Wikidata entity, find a suitable match in the target database. This is not a trivial task, with homonyms being the first challenge. For instance, suppose we would like to know more about the singer John Smith. We type the name into the Wikidata search box: multiple results appear, and it takes a while to find the person we are looking for. Unfortunately, Wikidata does not hold much information about him and our search moves to another database, like MusicBrainz\footnote{https://musicbrainz.org}. We must repeat the same procedure as with Wikidata, and after some digging, we manage to find John Smith the singer. This is a match: the MusicBrainz entity can be linked to Wikidata. soweego aims at solving the issue at a large scale, via disambiguation of a source Wikidata entity to a target database entry. Figure 1.1 depicts the solution.

Wikidata entities are designed to store this kind of links. In particular, entities like John Smith are called \textit{items} and each of them is composed of an \textit{identifier}, a \textit{fingerprint}, and a set of \textit{statements}.\footnote{https://www.mediawiki.org/wiki/Wikibase/DataModel/Primer} Each statement is broken down into \textit{claims} (i.e., property, value pairs) and optional \textit{references} (i.e., the sources of the factual information). For instance, John Smith could have the claim \textit{(given name, John)}, together with its reference \textit{(stated in, Duckburg Municipality archive)}. The identifier links to external sources are expressed as claims too. The John Smith match between Wikidata and MusicBrainz would be e.g., expressed as a claim over the Wikidata item with identifier \texttt{Q666} (\texttt{MusicBrainz artist ID, 77a1c579-3532-491c-86bd-595ddd4780cc}), where the latter value corresponds to the MusicBrainz identifier.

Officially, soweego has the following goals:\footnote{https://meta.wikimedia.org/wiki/Grants:Project/Hjfocs/soweego#Project_goals}

1. to ensure live maintenance of identifiers for people in Wikidata, via link validation;

2. to develop a set of linking techniques that align people in Wikidata to corresponding identifiers in external catalogs;
Figure 1.1: *soweego* creates a connection (read performs disambiguation) between entries of the source database (Wikidata) and a target database. Image by Hjfoe, CC BY-SA 4.0

3. to ingest links into Wikidata, either through a bot (confident links), or mediated by curation (non-confident links);

4. to achieve exhaustive coverage (ideally 100%) of identifiers over 4 large-scale trusted catalogs;

5. to deliver a self-sustainable application that can be easily operated by the community after the end of the project.

The remainder of this thesis is structured as follows. In section 2 we report a brief review of the state of the art. The preliminary analysis of the candidate targets is detailed in section 3. Section 4 describes the project architecture with a focus on the matching strategies, which we evaluate against MusicBrainz in section 5. We draw our conclusions in section 6.
2 Related work

The alignment of Wikidata to third-party structured databases may tackle the lack of references, but it is a complex task. Although a given label can be the same between Wikidata and an external database, there could be ambiguous homonyms. To overcome the issue, we need to exploit other attributes in addition to labels. Choosing them is a challenge itself, since Wikidata and the target database have probably different attribute sets. It is not even assumable that the attributes will be the same among all the entities in the same KB, like Wikidata.

SocialLink is a system that aligns KB entries of people and organizations to the corresponding social media profiles,[3] and shares our problem. Its approach is to pick a minimal subset of attributes: name, difference between person and organization and temporal information that tells if the entity is alive or existent. Similarly, we choose full name, birth and death dates, as well as a set of URLs related to the entity. Unlike SocialLink, we allow the addition of target-specific attributes.

The exceeding attributes can improve the linking process, but they can also be exploited in a KB population task. In fact, mapping the semantics of these attributes against the Wikidata ontology would result in the addition of referenced statements. These statements cannot replace the work done by StrepHit[1] or the one described in [4], but we still view it as a contribution. In contrast to us, StrepHit focuses on unstructured data, typically free-text documents from Web sources.

[5] exploits an enhanced representation of the social media content, compared to [2]. Despite the improvement, we argue that the approach will not be helpful in soweego, since we cannot assume the availability of any social media data.

The alignment task deals with a lot of queries on those attributes, so working with the target database APIs could be an issue: not only API usage is restricted, but Web requests also bring latency in code execution. Similarly to [2], we work with an internal database, but we populate it through the target dumps, instead of the social medium feed.
3 Preliminary analysis

The very first task of this project is to select the target databases.\(^1\) We see two directions here: either we focus on a few big and well known targets as per the project proposal, or we can try to find a technique to link a lot of small ones from the long tail, as suggested by ChristianKl.\(^2\)

We used SQID\(^3\) as a starting point to get a list of people databases that are already used in Wikidata, sorted in descending order of usage.\(^4\) This is useful to split the candidates into big and small fishes, namely the head and the (long) tail of the result list respectively.

Quoting ChristianKl, it would be ideal to create a configurable tool that enables users to add links to new databases in a reasonable time-frame (standing for no code writing). Consequently, we carried out the following investigation: we considered as small fishes all the entries in SQID with an external ID data type, used for class human (Q5)\(^5\), and with less than 15 uses in statements. It results that some critical issues need to be solved to follow this direction, as described below.

The analysis of a small fish can be broken down into a set of steps. This is also useful to translate the process into software and to make each step flexible enough for dealing with the heterogeneity of the long tail. The steps have been implemented into a piece of software by the author of this thesis.\(^6\)

3.1 Retrieving the dump

The naïve technique to link two databases can be resumed as follows: for each entity in the former, look it up into the latter. Since databases contain a lot of data, we need the dumps to avoid APIs restrictions and slow computation due to connection latency. As we focus on people, it is also necessary to obtain the appropriate dump for each small fish we consider.

3.1.1 Problem

In the real world, such a trivial step raises a first critical issue: not all the database websites give us the chance to download the dump.

3.1.2 Solutions

A non-technical solution would be to contact the database administrators and discuss dump releases for Wikidata. This task is out of scope for our work.

On the other hand, autonomously building the dump would scale up much better. Given a valid URI for each entity, we can re-create the dump. However, this is not trivial to generalize: sometimes it is impossible to retrieve the list of entities, sometimes the URIs are merely HTML pages that require Web scraping. For instance, at the time of writing this thesis, Welsh Rugby Union men’s player ID (P3826),\(^7\) Berlinische Galerie artist ID (P4580),\(^8\) FAI ID (P4556),\(^9\) need scraping for both the list of entities and each entity; Debrett’s People of Today ID (P2255),\(^10\) AGORHA event identifier (P2345),\(^11\) do not seem to expose any list of people.

\(^1\)https://meta.wikimedia.org/wiki/Grants:Project/Hjfcos/soweego#Work_package
\(^2\)https://meta.wikimedia.org/wiki/Grants_talk:Project/Hjfcos/soweego#Target_databases_scalability
\(^3\)https://tools.wmflabs.org/sqid/#/browse?type=properties
\(^4\)Select datatype set to ExternalId. Used for class set to human Q5
\(^5\)https://www.wikidata.org/wiki/Q5
\(^6\)https://github.com/MaxFrax/Evaluation
\(^7\)https://www.wikidata.org/wiki/Property:P3826
\(^8\)https://www.wikidata.org/wiki/Property:P4580
\(^9\)https://www.wikidata.org/wiki/Property:P4556
\(^10\)https://www.wikidata.org/wiki/Property:P2255
\(^11\)https://www.wikidata.org/wiki/Property:P2345
### 3.2 Handling the format
The long tail is roughly broken down as follows:

- XML;
- JSON;
- RDF;
- HTML pages with styling and whatever a Web page can contain.

#### 3.2.1 Problem
Formats are heterogeneous. We focus on open data and RDF, as dealing with custom APIs is out of scope for this investigation. We also hope that the open data trend of recent years would help us. However, a manual scan of the small fishes yielded poor results. There were 56 targets in our long tail, out of 16 randomly picked candidates, only YCBA agent ID\(^{12}\) was in RDF, and has thousands of uses in statements at the time of writing this thesis.

#### 3.2.2 Solution
To define a way (by scripting for instance) to translate each input format into a standard project-wide one. This could be achieved during the next step, namely ontology mapping between a given small fish and Wikidata.

### 3.3 Mapping to Wikidata
Linking Wikidata items to target entities requires a mapping between both schemas.

#### 3.3.1 Solution
The mapping can be manually defined by the community: a piece of software will then apply it. To implement this step, we also need the common data format described above.

#### 3.3.2 Side note: available entity metadata
Small fishes may contain entity metadata which are likely to be useful for automatic matching. The entity linking process may dramatically improve if the system is able to mine extra property mappings. This is obvious when metadata are in different languages, but in general we cannot be sure that two different databases hold the same set of properties, if they have some in common.

### 3.4 Conclusion
It is out of scope for the project to perform entity linking over the whole set of small fishes. On the other hand, it may make sense to build a system that lets the community plug in new small fishes with relative ease. Nevertheless, this would require a reshape of the original proposal, which comes with its own risks:

- it is probably not a safe investment of resources;
- eventual results would not be in the short term, as they would require a lot of work to create a flexible system for everybody’s needs;
- it is likely that the team is not facing eventual extra problems in this phase.

Most importantly, a system to plug new small fishes already exists: Mix’n’match\(^{13}\), which is specifically designed for the task.\(^{14}\)

---

\(^{12}\)https://www.wikidata.org/wiki/Property:P4169

\(^{13}\)https://tools.wmflabs.org/mix-n-match/

\(^{14}\)http://magnusmanske.de/wordpress/?p=471
4 System architecture

soweego is a pipeline that performs several tasks in order to link entities between Wikidata and the target database. We drive the description by means of our work on MusicBrainz. MusicBrainz is a community-maintained open-source database of music information. Given our people use case, we focus on the subset of musical artists.

4.1 Importer

The importer module is in charge of downloading the latest target database dump and store it in an internal database table. The importer is designed to slice the dump in order to only keep the desired subset. Since soweego processes different databases, a configurable middleware takes care of the task. The middleware allows to pre-process the data and to map it against our extendable model. The core model is as follows:

- **Catalog ID**, i.e., the identifier of the entity in the target database;
- **Label**, i.e., the full name;
- **Birth Date**;
- **Birth date precision**;
- **Death Date**;
- **Death date precision**.

The extendable schema allows to leverage the peculiarities of the targets, while shared procedures can be applied thanks to the core model.

A common issue (in music databases above all) is people having multiple names, known as aliases. The schema easily handles them by treating the aliases as standalone entities. Clearly, these standalone entities duplicate all the original entity data, except the label. Since the pipeline only reads from the database, a de-normalized approach naturally fits better than a normalized one.

The import phase actually attempts to create two tables: one as described above, and one for the external links. External links are valid URLs pointing to additional resources out of the target database. The core model chosen for links is as follows:

- **Entity ID**, i.e., a foreign key to retrieve the person information;
- **URL**, i.e., the original full link available in the dump;
- **Tokens**, i.e., a tokenized version of the link.

The tokens attribute contains the link without the meaningless words of a well-formed URL, such as http. This is inspired by Natural Language Process (NLP) techniques for highlighting the semantics of a text. For instance, the tokenized link does not distinguish the same resource referenced by m.facebook.com or facebook.com.

---

1https://musicbrainz.org
2https://musicbrainz.org/doc/About
4.2 Linker
The linker module is the core engine of the whole system: it is in charge of linking Wikidata to the chosen target database. At the time of writing this thesis, it does not exploit any statistical/probabilistic approach, such as neural networks. It is instead implemented as a set of rule-based linking strategies. A machine learning approach will probably be explored in the further steps. The rule-based linking strategies can be viewed as a set of features for a probabilistic classifier.

4.2.1 Perfect name match
Checking if two human entities share the same full name is the most straightforward linking strategy, but it is prone to many errors. First, it fails at homonyms disambiguation; second, it is not tolerant against string differences. For instance, people with a second name like George Walker Bush can appear also as George Bush or George W. Bush. On top of that, the label George Bush is ambiguous because it can refer to both George Bush junior or George Bush senior. To reduce the amount of ambiguous entities, we apply this strategy to a relevant source database subset. In the case of MusicBrainz, we slice the musicians subset out of Wikidata. As observed in the example above, this heuristic does not completely solve the problem: for instance, both the Bushes were politicians.

4.2.2 Perfect name with birth and death dates match
The perfect name match strategy can be improved by involving other attributes in the process. The George Bush ambiguity described above is easily solved through the birth date attribute. However, the availability of dates can be an issue. In fact, dates can be missing or with low precision (e.g., month or day may be missing). Therefore, the dates match can only deal with the least precise shared date. Some math built on the birthday problem gives us an idea of how likely is to have homonyms with same full birth date. Nevertheless, we expect that our use case databases do not fall into this problem, since they do not cover all existing people.

4.2.3 Cross-database URL match
As widely mentioned in this thesis, multiple Web sources may represent the same entity (humans in our case). In general, there is a one-to-many relationship between an entity and its representations in the Web, such as a Twitter profile, a MusicBrainz page, a Wikidata item, etc. Indeed, we can consume the low-hanging fruits of such multiple representation at linking time: as a rule of thumb, two database entries can be linked if they share an URL pointing to the same resource. In some optimal cases, MusicBrainz can even contain Wikidata entity URLs. Wikidata does not explicitly expose external links: it is rather a combination of an external ID and a URL formatter. A valid reference to the external source can be built from these data.

    4https://www.capgemini.com/2011/09/same-name-same-birth-date-how-likely-is-it/
target database ones. However, it is not robust to URL heterogeneity: for instance, it fails when one URL starts with http and the other with https. Making a Web request to these two URLs and comparing the response would be the most suitable way to assess their equality. Still, it would bears a significant latency in code execution.

We improve the strategy robustness by taking inspiration from NLP techniques. The intuition is that there are many variable parts in a valid URL: for instance, http versus https, the presence or absence of a trailing slash /, of leading www or m, and so forth. Consequently, we tokenized the URL and removed stop words like the aforementioned ones. For instance, https://twitter.com/GeorgeHWBush would become ['twitter', 'com', 'GeorgeHWBush'].

4.2.4 Normalized full names match
As described in Section 4.2.2, the perfect name match strategy has some flaws. For instance, Vladimir Vladimirović Putin or vladimir vladimirović putin would not match, due to the different usage of capital letters. Another issue arises when comparing different language alphabets, such as Latin Vladimir Vladimirović Putin and Cyrillic Владимир Владимирович Путин. There are many normalization techniques that help reduce the impact of these problems. The first problem is easily solved by always working with lowercased strings; the second one requires transliteration\(^5\) of non-Latin alphabets into Latin via conventional conversion tables. In addition, diacritics get normalized to the ASCII character set, thus solving mismatches due e.g., to accents omission. Finally, word tokenization takes place, resulting in a set of tokens from the normalized string. Tokenization is implemented as a string split by non-word characters, specifically through the regular expression \W+.

4.3 Ingestor
The main goal of the system is to improve Wikidata content: hence, confident output should be directly added. Wikidata bots\(^6\) are non-human users allowed to perform high-volume edits. A bot account must first undergo community discussion for eventual approval, since it can damage a lot of data. Specifically, soweego adds confident output to Wikidata through a bot included in the ingestor module, while we plan to upload non-confident output to the Mix’n’match\(^7\) tool, which enables human curation of identifier matches. The first approved task\(^8\) of the soweego bot\(^9\) is the addition of links in the form of External ID\(^10\) claims,\(^11\) or references\(^12\) whenever the claim already exists.

4.4 Validator
Given a target database, the validator module retrieves already linked Wikidata entities and performs validation checks according to a set of criteria. First, dead or wrong identifiers should be removed. More specifically, the target database may not contain the entity anymore or the entity may have moved to another identifier. In these cases, the validator interacts with the ingestor to remove the invalid link. In the subsequent run, the linker module will also propose matches to those entities that were affected, thus potentially fixing eventual validation errors.

The second validation criterion relies on a set of essential attributes, namely gender, birth date, birth place, death date, and death place. Clearly, a check over this criterion can only run if both the Wikidata and the target database entities share at least a subset of these attributes. In case of mismatches, the link is marked as invalid and the validator sends it to the ingestor for deprecation.

We foresee the implementation of more fine-grained checks focusing on data consistency. For instance, a dead person cannot be married after the death date. The validator module will play a central role in the final system: first, it will serve as a tool to detect divergence between Wikidata and a target database; second, it will be responsible of the training set construction for linking strategies

---

\(^5\)https://en.wikipedia.org/wiki/Transliteration
\(^6\)https://www.wikidata.org/wiki/Wikidata:Bots/
\(^7\)https://tools.wmflabs.org/mix-n-match/
\(^8\)https://www.wikidata.org/wiki/Wikidata:Requests_for_permissions/Bot/soweego_bot
\(^9\)https://www.wikidata.org/wiki/User:Soweego_bot
\(^10\)https://www.wikidata.org/wiki/Wikidata:Glossary/en#External_identifier
\(^11\)https://www.wikidata.org/wiki/Wikidata:Glossary/en#Claim
\(^12\)https://www.wikidata.org/wiki/Wikidata:Glossary/en#Reference
based on machine learning.
5 Results

`soweego` has been tested against the 19 September 2018 MusicBrainz dump\(^1\) and a sample of Wikidata entities. The MusicBrainz artist subset contains 986,765 artists\(^2\) and 181,221 aliases. The Wikidata sample consists of 1100 entities randomly picked from 1% of all musicians with no MusicBrainz link.\(^3\) The table below shows the performance of the linking strategies outlined in Section 4.2.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Total matches</th>
<th>Checked</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perfect names</td>
<td>381</td>
<td>38 (10%)</td>
<td>84.2%</td>
</tr>
<tr>
<td>Perfect names + dates</td>
<td>32</td>
<td>32 (100%)</td>
<td>100%</td>
</tr>
<tr>
<td>Normalized names</td>
<td>227</td>
<td>24 (10.6%)</td>
<td>70.8%</td>
</tr>
<tr>
<td>Perfect Link</td>
<td>71</td>
<td>71 (100%)</td>
<td>100%</td>
</tr>
<tr>
<td>Token Link</td>
<td>102</td>
<td>102 (100%)</td>
<td>99%</td>
</tr>
</tbody>
</table>

We first observe that link-based strategies achieve the best performance. They match almost 10% of the sample with near 100% precision. Figure 5.1 illustrates why their coverage is 10%: the `perfect link matches` are a subset of the `token link matches`. We note that link-based strategies may be improved if we also consider the official home page Wikidata property, which has not been used so far.

The perfect name strategy improves recall with a 35% coverage over the Wikidata sample, at the cost of precision. We assessed a randomly picked 10% of these matches, obtaining a precision of 84%. We argue that the relatively high precision stems from the very high quality of MusicBrainz data and the small size of the checked set. We also isolated the subset of matches obtained by the perfect name strategy alone and not by other ones (cf. the leftmost set in Figure 5.2), resulting in a precision of 68%.\(^4\)

![Figure 5.1: Venn diagram of the links matches](image1)

---

\(^1\)ftp://ftp.eu.metabrainz.org/pub/musicbrainz/data/fullexport/

\(^2\)Entries in MusicBrainz tagged as artists or not tagged

\(^3\)Entities having an `occupation` property (P106) with value `musician` (Q639669) and all its direct subclasses

\(^4\)10% of the leftmost set considered, 19 valid on 28 tests
The normalized name strategy performs similarly to the perfect name one. This is possibly due to the relatively big intersection of matches between the two strategies, as shown in Figure 5.3. We notice that the normalized name strategy has a worse recall than the perfect name one. After further investigation, we realized that Wikidata multilingual labels are responsible for such performance decrease: the current implementation actually adds them to the token set of the entity and we trigger a match only if the source and target token sets are equal. Hence, Arabic, Chinese, or Russian labels that are only available in Wikidata and not in MusicBrainz create a consistent amount of false negatives.

Unsurprisingly, date-based strategies yield high precision, although we note that they are not sufficient alone. Therefore, we can view them as constraints for other strategies, such as in the perfect name and dates one.

**Figure 5.2**: Venn diagram of perfect name matches, token link matches and perfect name and dates matches

**Figure 5.3**: Venn diagram of the perfect name matches and the normalized names matches
6 Conclusion

Wikidata is a general-purpose structured knowledge base housed by the Wikimedia Foundation. Its community is becoming more and more active in data curation: in the last year, the number of edits increased by 75.26%, compared to the previous year. Despite such increased effort, Wikidata still suffers from a lack of references that support the trustworthiness of its content: currently, only less than a quarter of it has a reference to non-wiki sources and roughly a half is totally unreferenced.

In this thesis, we described the first development iteration of soweego, an automatic linking system for large catalogs that aims at filling the reference gap. Specifically, we illustrated the Wikidata - MusicBrainz use case. Our contribution boils down to a set of core building blocks, the most prominent being the linker, plus the ingestor, validator, and importer components, which lay the foundation for the final product. Despite the project is only 3 months old, with a planned duration of 1 year, we managed to add several hundreds of high-quality matches to Wikidata.

In the linker module, we explored rule-based matching strategies, which serve as a baseline system for construction and comparison of future machine learning-based strategies. Furthermore, we acquired insights on the actual data quality of Wikidata and earned expertise on the key features for improvement of the linking process. As described in Section 5, links are a very precise feature to rely upon. Although dates perform well, they are not a valid standalone feature: on the other hand, they have a positive impact on the precision if they are combined to other ones. Finally, full name matching is a risky feature, as expected, creating a lot of false positives.

The very next steps will focus on the improvement of existing strategies. First, the cross-database URL strategy introduced in Section 4.2.3 will consume official homepages of Wikidata entities, as well as the ISNI code of the MusicBrainz ones. Moreover, we will investigate how tokenization in normalized names strategy (cf. Section 4.2.4) can be better exploited.

Future work will include deeper research in the record linkage literature to improve the system, with a special attention on probabilistic approaches. Machine learning could be an interesting direction to take, as shown in [3]. The immediate goal is to obtain a confidence score for each computed match, expressed as probability. The baseline strategies could become our features in a machine learning-based solution. Moreover, inspired by [3], we plan to use the full names strategies to select a subset of entities as input for the linking phase. In this way, our system would perform faster, thanks to the lower number of entities to check. This is possible just because we verified that full names matches tend to be a super set of the other strategies matches.

---

1. https://wikimediafoundation.org/
2. https://stats.wikimedia.org/v2/#/wikidata.org/contributing/edits/normal|bar|1-Year|total
3. https://docs.google.com/presentation/d/1XX-yzT98fgLaFHKSeioxO1tXC1uxezxjdZT9TYI/edit?usp=sharing, slides 15 to 19
Bibliography


