

# Enhancing Wikidata with other Knowledge Graphs

Vivek Iyer<sup>1</sup>

IIIT Hyderabad, Gachibowli, India  
vivek.iyer@research.iiit.ac.in

## 1 Introduction

The Wikidata knowledge graph<sup>1</sup>[?] is a collaborative knowledge base built by the Wikimedia foundation and is one of the largest KGs available to the public in terms of breadth, with over 75 million entities. However, entities in Wikidata are very sparsely connected and only have a very limited number of attributes and relations. The Wikidata KG thus needs to be enriched, in order to create a stable, denser and more detailed KG. There are two common ways of doing this, merging with an existing Knowledge Graph, or enriching Wikidata with fresh entities and relationships extracted from Wikipedia articles. In this project, we focus on the former and use a variety of popular Knowledge Graphs as well as domain specific ontologies to enrich the Wikidata entry for the "pizza" entity.

## 2 Literature Survey

The lack of completeness of Wikidata necessitates research into Wikidata enrichment, or more broadly, knowledge base enrichment. [8] proposes enrichment of structured knowledge bases, in particular DBPedia, by semantifying web extracted facts. They map subject and object terms of these facts to instances, and relational phrases to object properties and use Markov Clustering to group identical semantic relationships with different surface forms together. Knowledge base enrichment with counting quantifiers [10] has been implemented.

However, there has been little work done with regards to enrichment of specific knowledge bases such as DBPedia or Wikidata. The DBPedia dataset has been used as background knowledge [6] beside text to extract new ontological relations. The results are evaluated using a customized experimental design called the "Pseudo Gold Standard based Ontology Evaluation" that compares the results obtained by a human expert against those obtained automatically. The enrichment of DBPedia using statistical methods in order to detect inconsistencies to extract knowledge bases of higher quality has also been proposed [13].

Most importantly, there has also been some work in attempting to merge DBPedia and Wikidata knowledge graphs by incorporating Wikidata in the DBPedia ecosystem in the form of DBw [9]. As part of our project, we seek to achieve

---

<sup>1</sup> <https://www.wikidata.org/>

the opposite, i.e. enriching entities in Wikidata with relations from DBPedia in order to make the former more densely connected. Given Wikidata (75.6 million entities) contains significantly more entities than DBPedia (6 million entities), this effort is worthwhile since it would result in the creation of a much larger knowledge graph than was possible using DBw.

### 3 Problem Statement

The goal of this project is to a) enhance the Wikidata Knowledge Graph by combining information from other Knowledge Graphs, and then b) develop a validation workbench for final validation of extracted relations before merging with Wikidata. Former, the focus would be on enriching Wikidata with generic multi-purpose Knowledge Graphs like BabelNet[11], Google Knowledge Graph[2], DBPedia[5], Freebase[1] and YAGO [4], as well as domain specific sources like the Pizza Ontology [3]. Combining these Knowledge Graphs allows the harvesting of information from varied knowledge sources such as WordNet (YAGO and BabelNet), manual contributions (Freebase and the Pizza Ontology), Wikipedia (DBPedia) and the miscellaneous sources used by the Google KG including Wikipedia, Wikidata and the CIA World Factbook.

As part of this project, I will be focusing on enriching the "Pizza" entity<sup>2</sup> as a proof of concept. This entity was chosen because a) the corresponding Wikidata entry is sparse and contains few number of relations b) the domain specific source is readily available as the Pizza ontology[3], a widely cited ontology in the research community published by Stanford University.

### 4 Pipeline

As shown in the figure, the **input** to the Wikidata enrichment model consists of entity(s) extracted from Wikidata that need to be enriched. As part of our project, for the time being I focus on enriching only the "Pizza" Wikidata entity and then move on to other entities as future work. The procedures that comprise the Wikidata enrichment pipeline can be classified into the following stages:

#### 4.1 Stage 1: RDF Triple extraction from KGs

**Input:** Entity(s) to be enriched

**Output:** Raw RDF triples extracted from various KGs

The steps followed as part of this stage are as follows:

- RDF triples from Freebase, DBPedia, BabelNet, YAGO and Google KG are extracted in this stage. These raw triples would be of the form (subject, relation, object) where the subject is the Wikidata Pizza entity.

---

<sup>2</sup> <https://www.wikidata.org/wiki/Q177>

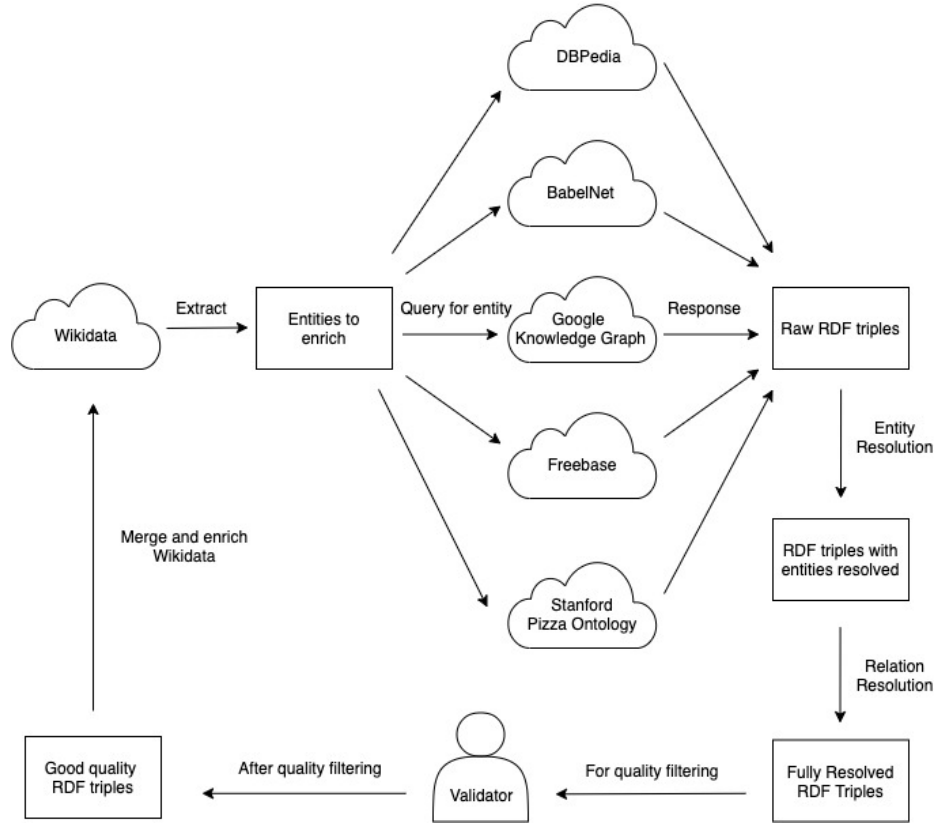


Fig. 1: Pipeline of the Wikidata Enrichment algorithm

- Non-English results were filtered out of the response
- The filtered (property, entity) tuples were written to a file

But in these extracted RDF triples, the same entity having different forms are counted as separate entities. For example, Sachin Tendulkar and S.R Tendulkar refer to the same entity but have different linguistic representations. Hence we proceed to Stage 2.

#### 4.2 Stage 2. Entity Resolution:

**Input:** Raw RDF triples

**Output:** RDF triples with resolved entities

As noted in the previous section, the data obtained from stage 1 contains concepts referring to the same entity but having different representations. We would like to disambiguate these concepts and map them to the corresponding Wikidata entity, regardless of representation.

The steps followed as part of this stage are as follows:

- For each entity, a query was made to Wikidata and the top 5 search results were returned.
- These 5 results were then ranked on the basis of a similarity score with the original extracted entity.
- In order to calculate this similarity score, various approaches were tried:
  - **Naive Sequence Matcher (without threshold)**
    - \* This algorithm uses the Gestalt pattern matching approach [12] for string comparison and is a form of syntactic character-level comparison that does not take semantics into consideration.
    - \* It always returns the closest hit out of the 5 results (i.e. the one with the maximum similarity score) regardless of how low the top similarity is.
    - \* This algorithm yields an **accuracy** of **83.11%**
  - **Universal Sentence Encoder (with threshold)**
    - \* Here I use the Universal Sentence Encoder [7], which is a semantic approach to string similarity.
    - \* This is preferred over traditional distributional embeddings models like word2vec because our use case involves disambiguating entities, which are more often than not phrases and these phrases are often not present in the word2vec vocabulary.
    - \* Thus instead of going for a semantic approach dependent on a finite vocabulary, the Universal Sentence Encoder is preferred since it is pretrained to convert any sentence (or phrase) to a fixed-length embedding.
    - \* Returns the closest hit iff similarity is greater than a certain threshold.
    - \* Yields an accuracy of 87.41% at an optimum threshold of 0.6
  - **Sequence Matcher (with threshold)**
    - \* Combines the ideas of the previous two algorithms, by using Gestalt pattern-matching, but with a threshold for filtering poor quality results
    - \* Returns the closest hit iff similarity is greater than a certain threshold.
    - \* Yields an accuracy of 91.52% at an optimum threshold of 0.56
    - \* Surprisingly yields better accuracy than the semantic USE approach, presumably due to the fact that USE is better suited for sentences and longer phrases, not phrases containing that refer to specific entities and consist only of a few words.
    - \* Moreover, since a huge percentage of entities result in exact matches, making semantic comparisons was found to be unnecessary
- Hits belonging to fields that had majority misses were not disambiguated, and were left in original string representation. This is because fields that resulted in majority misses (for ex. images, description etc) were likely not needed to be disambiguated in the first place.
- Similarly, terms that did not have corresponding entities in Wikidata were not disambiguated either.

This stage ensured entity disambiguation and thus guaranteed a unique form for each entity. But this still does not ensure statement uniqueness, due to various problems with relation representation:

- Different KBs have different formats for representing relations. For instance, Freebase uses internal paths, DBPedia uses external URLs and BabelNet uses compound words joined by camelCase to represent its relations.
- In addition, different relations in different KBs have different levels of granularity, making merging relations an even more complicated task. For instance, Freebase contains the broad relation "type" that could refer to any of these fine-grained relations on Wikidata: "instance of", "subclass of", "category of" etc.
- Even within a KB, relations have names that often do not indicate much about their purpose, necessitating a reading of the relation description page. For instance, "wikiPageWikiLink" for external links in DBPedia and "lnToCompound" for compound Words in BabelNet

To solve these issues with relation disambiguation, we proceed to stage 3.

### 4.3 Stage 3: Relation Disambiguation

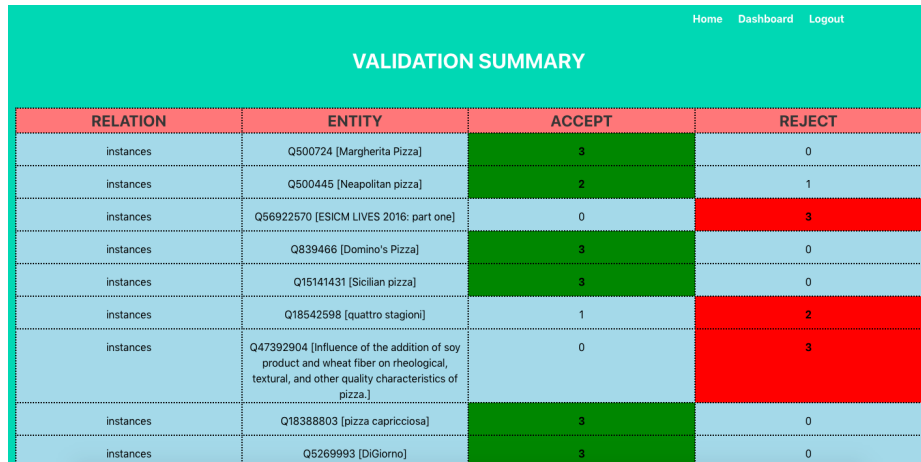
**Input:** RDF Triples with resolved entities but unresolved relations

**Output:** RDF Triples with resolved entities and relations

- As mentioned in the previous stage, there are various issues with relation representation, that necessitate relation disambiguation. However, automated relation disambiguation was found to be infeasible.
- As mentioned earlier, it is necessary to understand the granularity of the relation first before mapping accordingly. However it would be outside the scope of this project, if not outright impossible, to predict granularity from a single phrase alone.
- The problem with relation names not being sufficient enough to convey meaning can be solved by obtaining relation descriptions and then using a similarity comparison on these descriptions. However developing a general purpose scraper for automated extract of description from relation description pages with vastly different formats would, once again, be a very challenging task.
- An alternative to scraping descriptions, therefore, would be merging relations that are mapped to a "lot" (more than a fixed threshold) of common entities. However experimentally, it was observed that most of the information extracted from different KGs corresponds to new relations and new entities, making automated merging a hard (and erroneous) task.

To combat these issues, manual relation resolution was implemented as follows:

- As a preprocessing step, relations were queried for possible matches in Wikidata, and the most similar matches were returned as suggestions to a human validator.
- The human validator manually queries Wikidata for each relation, reads the corresponding description pages and selects the most appropriate one based on granularity and the relation description.
- While automated relation resolution was found to be infeasible because of the issues mentioned previously, manual resolution of relations can be done in a feasible, efficient and scalable manner.
- The **efficiency** is due to the dataset containing a relatively very small number of unique relations as compared to unique entities. For instance, in my dataset, there were **19 relations** for **500 entities**.
- Extending this approach to multiple domains is **scalable** because most of these are **domain-independent**, so there is a very finite subset of relations that need to be manually annotated.
- At present, for one entity, manual relation annotation was trivial but for multiple entities, the task could be crowdsourced.



RELATION	ENTITY	ACCEPT	REJECT
instances	Q500724 [Margherita Pizza]	3	0
instances	Q500445 [Neapolitan pizza]	2	1
instances	Q56922570 [ESICM LIVES 2016: part one]	0	3
instances	Q839466 [Domino's Pizza]	3	0
instances	Q15141431 [Sicilian pizza]	3	0
instances	Q18542598 [quattro stagioni]	1	2
instances	Q47392904 [Influence of the addition of soy product and wheat fiber on rheological, textural, and other quality characteristics of pizza.]	0	3
instances	Q18388803 [pizza capricciosa]	3	0
instances	Q5269993 [DiGiorno]	3	0

Fig. 2: Response Summary in the Validation Workbench

#### 4.4 Stage 4: Crowdsourced Validation

This stage involves using a human validator to validate the quality of all the extracted triples. This is done to ensure that only good quality information is added to Wikidata so that it doesn't get polluted.

A **validation workbench** was developed for this purpose and assists the validation process as follows:

- The validation workbench provides support for two types of users: normal users, and admin users.

Home Dashboard Logout

### VALIDATION FORM

RELATION	ENTITY	ACCEPT	REJECT
instances	Q500724 [Margherita Pizza]	<input type="radio"/>	<input type="radio"/>
instances	Q500445 [Neapolitan pizza]	<input type="radio"/>	<input type="radio"/>
instances	Q56922570 [ESICM LIVES 2016: part one]	<input type="radio"/>	<input type="radio"/>
instances	Q839466 [Domino's Pizza]	<input type="radio"/>	<input type="radio"/>
instances	Q15141431 [Sicilian pizza]	<input type="radio"/>	<input type="radio"/>
instances	Q18542598 [quattro stagioni]	<input type="radio"/>	<input type="radio"/>
instances	Q47392904 [Influence of the addition of soy product and wheat fiber on rheological, textural, and other quality characteristics of pizza.]	<input type="radio"/>	<input type="radio"/>
instances	Q18388803 [pizza capricciosa]	<input type="radio"/>	<input type="radio"/>
instances	Q5269993 [DiGiorno]	<input type="radio"/>	<input type="radio"/>

Fig. 3: Accept/reject form in the Validation Workbench

- The admin user uploads validation files for the normal users to validate.
- Normal users get a list of files to validate on their dashboard.
- Each file is parsed to display an accept/reject form (Figure 3). After validating all relations, the user clicks on "Submit" and the accept/reject decisions are written back to the database.
- The admin can view a summary of responses (Figure 2) and see how many people accepted/rejected a particular relation.
- The final accept/reject decision is done based on majority voting.
- To ensure quality, the form includes some basic accept/reject questions that must be answered correctly in order for the response to be considered.
- With these quality control measures, this workbench can be used for crowd-sourced validation of relations, before finally merging with Wikidata entity.

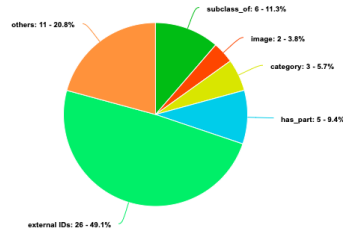


Fig. 4: Distribution of relations in pizza entity before enrichment

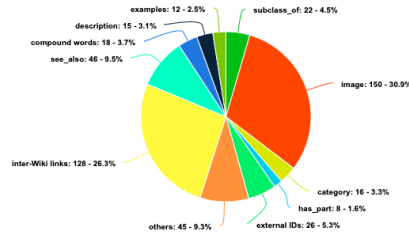


Fig. 5: Distribution of relations in pizza entity after enrichment

## 5 Observations

After completion of all four stages, the resultant data was analyzed and compared with the original data, resulting in the following observations:

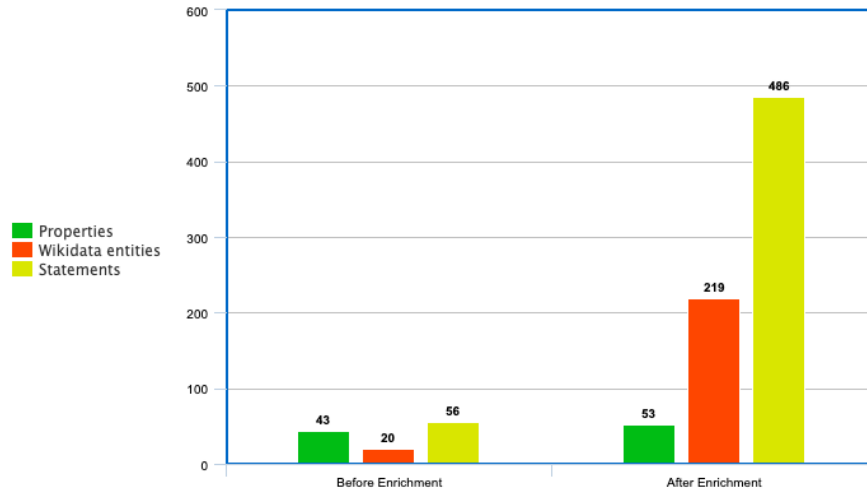


Fig. 6: Number of properties, connected entities and statements in the Wikidata pizza entity before and after enrichment

- The validation process revealed the accuracy of the Wikidata enrichment algorithm to be 90.02%, i.e. 90% of extracted relations were valid to merge with Wikidata.
- Figures 4 and 5 show the distribution of relations in Wikidata before and after enrichment.
- Before enrichment, external IDs such as JSTOR ID, Quora topic ID, NYT topic ID etc. constituted approx. 50% of statements before enrichment. These statements do not provide much local information about the given entity. With more meaningful relations added during enrichment, the percentage of external IDs has gone down to 5.3%.
- The distribution (and number) of images has shot up drastically, from 3.8% to 30.9%. This can be attributed to BabelNet being a rich source of images.
- These figures show that not only have the existing relations become more dense, the number and diversity of relations has also increased.
- Figure 6 shows the number of properties, connected entities and statements in Wikidata pizza entity before and after enrichment. While the number of properties has only increased by 23%, the number of statements has increased drastically by 767.85%. This is due to a large increase in the number of entities but only a small increase in the number of relations.
- This indicates that it is not that Wikidata does not have many properties, but that Wikidata is not complete enough.
- The figure also shows that the original Wikidata Pizza entity was linked to 20 entities, but the enriched entity is now linked to 219 entities, indicating an increase in completeness by 995%.



## 6 Conclusion and Future Work

In this project, a Proof of Concept was successfully presented for merging Wikidata with other Knowledge Bases by focusing on one specific entity. The corresponding pipeline was fully implemented, from extraction of triples from these KBs, to entity and relation disambiguation and also developing a workbench for facilitating crowdsourced validation.

As part of future work, the following goals remain to be accomplished:

- The work done thus far has been for one specific domain, but as part of future work, we would like to extend it to multiple domains. This can be easily achieved as the approach adopted has been domain-independent.
- However, when trying to extend to multiple domains, crowdsourcing relation resolution and statement validation becomes a bit more challenging. Apart from the current quality checks introduced in the validation work bench, other, more reliable quality control mechanisms (such as using social media profiles to judge validator credibility) can be adopted.
- Since the scope has now shifted to multiple entities and we have a larger dataset, relation resolution can also be partially automated by mapping relations with common entities. These mappings can then be used as suggestions to aid the validators in charge of relation resolution.
- Quality control can also be partially automated by using Active Learning-based algorithms to query the validator for only some data points, which would make the quality control process much more scalable and efficient.

## References

1. Freebase. <https://developers.google.com/freebase>, accessed: 2020-11-20
2. Google knowledge graph. <https://developers.google.com/knowledge-graph>, accessed: 2020-11-20
3. Stanford pizza ontology. <https://protege.stanford.edu/ontologies/pizza/pizza.owl>, accessed: 2020-11-20
4. Yago. <https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/>, accessed: 2020-11-20
5. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: Dbpedia: A nucleus for a web of open data. In: The semantic web, pp. 722–735. Springer (2007)
6. Booshehri, M., Luksch, P.: An ontology enrichment approach by using dbpedia. In: Proceedings of the 5th International Conference on Web Intelligence, Mining and Semantics. pp. 1–11 (2015)
7. Cer, D., Yang, Y., Kong, S.y., Hua, N., Limtiaco, N., John, R.S., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., et al.: Universal sentence encoder. arXiv preprint arXiv:1803.11175 (2018)
8. Dutta, A., Meilicke, C., Stuckenschmidt, H.: Enriching structured knowledge with open information. In: Proceedings of the 24th international conference on world wide web. pp. 267–277 (2015)

9. Ismayilov, A., Kontokostas, D., Auer, S., Lehmann, J., Hellmann, S., et al.: Wiki-data through the eyes of dbpedia. *Semantic Web* **9**(4), 493–503 (2018)
10. Mirza, P., Razniewski, S., Darari, F., Weikum, G.: Enriching knowledge bases with counting quantifiers. In: *International Semantic Web Conference*. pp. 179–197. Springer (2018)
11. Navigli, R., Ponzetto, S.P.: Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence* **193**, 217–250 (2012)
12. Ratcliff, J.W., Metzener, D.E.: Pattern-matching-the gestalt approach. *Dr Dobbs Journal* **13**(7), 46 (1988)
13. Töpper, G., Knuth, M., Sack, H.: Dbpedia ontology enrichment for inconsistency detection. In: *Proceedings of the 8th International Conference on Semantic Systems*. pp. 33–40 (2012)