Context Matrix Methods for Property and Structure Ontology Completion in Wikidata

Jonathan A. Gómez
School of Data Science
University of Virginia
Charlottesville, United States
jag2j@virginia.edu

Thomas Hartka
School of Medicine
University of Virginia
Charlottesville, United States
trh6u@virginia.edu

Binyong Liang
School of Data Science
University of Virginia
Charlottesville, United States
bl9m@virginia.edu

Gavin Wiehl
School of Data Science
University of Virginia
Charlottesville, United States
gtw4vx@virginia.edu

***Abstract*—Wikidata is a crowd-sourced knowledge base built by the creators of Wikipedia that applies the principles of neutrality and verifiability to data. In its more than eight years of existence, it has grown enormously, although disproportionately. Some areas are well curated and maintained, while many parts of the knowledge base are incomplete or use inconsistent classifications. Therefore, tools are needed that can use the instantiated data to infer and report structural gaps and suggest ways to address these gaps. We propose a context matrix to automatically suggest potential values for properties. This method can be extended to evaluating the ontology represented by knowledge base. In particular, it could be used to propose types and classes, supporting the discovery of ontological relationships that lend conceptual identification to the content entities. To work with the large, unlabelled data set, we first employ a pipeline to shrink the data to a minimal representation without information loss. We then process the data to build a recommendation model using property frequencies. We explore the results of these models in the context of suggesting type classifications in Wikidata and discuss potential extended applications. As a result of this work, we demonstrate approaches to contextualizing recently-added content in the knowledge base as well as proposing new connections for existing content. Finally, these methods could be applied to other knowledge graphs to develop similar completions for the entities contained therein.**

***Index Terms*—Wikidata, semantic web, machine learning**

I. Introduction

A. Background

The Wikimedia Foundation designed Wikidata to support shared data management across the Wikipedia family of sites [1]. Over time, the project evolved into a collaboratively-edited multilingual knowledge graph that provided central storage for structured data. The ongoing growth of Wikidata relies heavily on community contribution. However, the nature of unsupervised or minimally supervised crowdsourcing rendered Wikidata with problems such as inaccurate entries, vague relations between entries, incomplete properties, ambiguous conceptual domains, and redundant data.

A number of tools attempt to address issues of completeness and structural organization, such as the Relative Completeness Indicator (ReCoin) and the Wikidata Concept Tree Generator [2], [3]. ReCoin can suggest missing properties yet only does so when the author has already specified a well-defined type for the entity. The Wikidata Concept Tree Generator only shows existing links and cannot suggest new relationships. The present work investigates constructing a tool to suggest missing properties based on structurally-comparable entries. We intended to build a tool that can be employed to provide recommendations for entities based on similar properties and is agnostic to whether an author has defined types for entities. Importantly, this can be used for newly-authored entities and evaluate the appropriateness of existing entities.

Thanks to Wiki Education for sponsoring this work.

B. Objectives

The goal of this work is to develop a recommendation system that could predict the most likely value for a particular property of an entity. One potential application is to provide suggestions during editing of Wikidata. For example, a user might start a new entry and enter 'Place of birth' {P19}, 'Date of birth' {P569}, and 'Spouse' {P26}. This system would then hypothetically predict a likely value for 'Instance of' {P31} to be 'human' {Q5}.

In addition to facilitating the authorship of new entries in Wikidata, such a system could also be used for anomaly detection. A particular property of existing entities could be predicted, then entities could be flagged for review in cases where the actual value differed significantly from the predicted values. This type of anomaly detection could be useful in reviewing sets of entities within Wikidata to identify areas for improvement.

II. Literature review

Wikidata was originally proposed in 2012 by Wikimedia Deutschland with the goal of building a new collaborative platform for structured data to be used within the greater Wikimedia universe, such as Wikipedia [1]. Wikidata aims to provide well maintained, high quality data, and includes the ability to add references to data to help keep authorial sources transparent. Editing in Wikidata is open to the worldwide community, and its contents can be edited and read in every language. Wikidata, alongside other knowledge bases such as DBpedia and YAGO [4], [5], has become a significant member of the community of sites containing linked open data. Like many of these sites, Wikidata congregates data linked by ontological relationships, is inspired by the design principles

of the Semantic Web, and seeks to integrate and reason about data through data relationship links [6].

The data present in the knowledge base can be inconsistent in several ways because the information in Wikidata depends on public editors. Different dimensions in which to assess data quality in Wikidata include accuracy, objectivity, reputation, consistency, and completeness [7]. However, there is little in the way of objective measures in which to quantify each of these dimensions. Often times auditing the semantic graph along these dimensions involves considering the opinions of topic experts [8]. However, some tools have been developed in order to assess the quality of the linked data.

Previous work has been published in developing methods and tools to analyze the relationships between linked data entities in a knowledge base. The Relative Completeness Indicator (ReCoin) [2], is a tool developed for Wikidata that infers property-wise completeness based on the properties that are present in entities of a similar type. ReCoin allows for the measurement of information present in an entity relative to other related entities in the graph. Semantic graph completeness has also been examined through the use of matrix factorizations such as singular value decomposition, non-negative matrix factorization techniques, and Bayesian approaches such as latent Dirichlet allocation [9]. Further research has provided means for scalable statistical learning techniques in sparse semantic graphs [10].

Our work is based on the intuition behind ReCoin, in that we examine property-wise similarities to find patterns that relate back to semantic structure. However, in our work we aim to suggest an entity property values in a broader set of conditions, rather than measure relative completeness where entities are already typed by the author.

We evaluate our results by comparing predictions to existing values. Since this can be framed as an information retrieval problem ("given an entity and a property, return the best matches for value for the property"). Research into information retrieval therefore provides useful metrics. Recall measures the percentage of valid results obtained in a prediction. Precision measures the density of valid results in the set of results. Average precision takes the mean of the precision for each valid results, and for this reason can take advantage of the order of the suggestions. This last metric also roughly relates to the information density, in terms of the area under the recall-precision curve [11].

III. Data

A. Obtaining the data

We obtained our data from the Wikidata download archive in Resource Description Framework (RDF) truthy format. The file contained all active entries, with assertions in a triple structure: entity, property, value. The truthy designation denoted that the file contained only these assertions and lacked supporting information such as author-contributed citations for the assertions. Entities and properties were indicated with 'Q' and 'P' designations, respectively (e.g. 'Belgium' {Q31}, 'official language' {P37}). The archive used for this project was captured on Oct 2, 2020 and contained over 5.4 billion assertions. It was approximately 650 gigabytes (uncompressed) on disk.

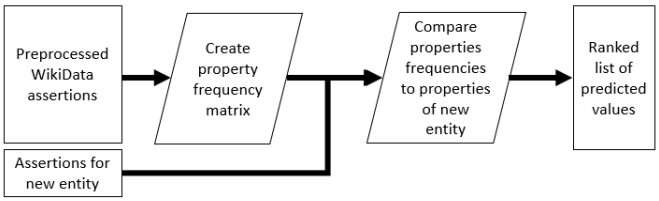

Fig. 1. Overview of the process to predicting values of a particular property for a target entity.

B. Processing the data

We used several pre-processing steps to reduce the size of the data into a format suitable for processing. We removed non-English names and descriptions for convenient interpretation of results. Next, we removed prefixes from entity and property designations. For example, '<http://www.wikidata.org/entity/Q31>' was trimmed to '31'. We then filtered down to assertions that represented edges between entities, effectively dropping edges that connect entities to literal values such as fixed numeric values. We reasoned that edges between entities would be most relevant to the overall structure of the knowledge graph of Wikidata.

IV. Methods

A. Model architecture

Our recommendation system is designed to predict values of a particular target property. It determines the frequency of properties associated with all entities that have a particular value for the target property (Fig. 1). We use these frequencies to predict the most likely value of the target property for a new entity. The appropriateness of a particular value is determined by measuring the similarity of the properties of the new entity with the property frequencies for each candidate value. The value with the highest similarity is considered to be the most related. This approach is based on methods for determining the similarity between text documents used the relative frequency of words [12].

B. Property frequency matrix derivation

Determining the property frequency matrix is a key step in this method. It captures the proportion of each property specified for entities that share a common value for the target property. This matrix is unique for each target property. Every possible value is a separate row of the frequency matrix while the columns represent the different properties.

The property matrix encodes data about the relative presence of properties across groups of entities. For example, the target property 'instance of' {P31} for a particular entity might have the value 'human' {Q5}, 'city' {Q515}, or 'language' {Q34770} (Fig. 2). Entities that are 'instance of' {P31} with the value 'human' {Q5} frequently have the properties 'place

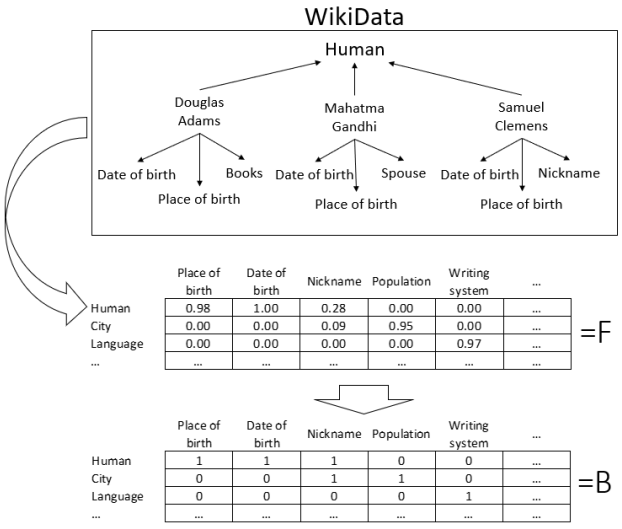

Fig. 2. Example of derivation of frequency matrix (F) and binary matrix (B) for 'instance of' property.

	Place of birth	Date of birth	Nickname	Population	Writing system	...	
Mel Brooks	1	1	0	0	0	...	=e

Fig. 3. Example of the vector representation of an entity's properties.

of birth' {P19} and 'date of birth' {P569}. In contrast, entities that are 'instance of' {P31} with the value 'city' {Q515} often have the property 'population' {P1082} and may have a 'nickname' {P1449}.

In order to calculate the property frequency matrix (**F**), the entities were grouped based on the value of the target property. Then, the number of entities with a certain property were divided by the total number of entities in the group. We also created a Boolean property matrix (**B**), where all non-zero frequencies were converted to one. However, in practice, we used a Jaccard similarity measure, discussed later, instead of working with the transformed matrix.

C. Similarity measurement

We then created a vector **e** for each entity for which we wanted to predict a value . Each entity vector contained a sequence over each possible property, with a one indicating the entity was linked through the property and a zero indicating that it was not. The order of properties is the same as that of the frequency matrix (Fig. 3).

We then determined the similarity of **e** to each row in **F**. This similarity was measured using several methods that were separately evaluated. These methods included cosine similarity, Manhattan distance, Jaccard index, inner product, and a penalized inner product. The penalized inner product was a custom similarity score we developed for this task. It provided a form that adds two penalty terms to the inner product. These penalty terms allowed us to decrease the score when properties of the entity or the target value were not present in both.

Given an entity **e** and a possible value represented by a row in the frequency matrix $\mathbf{F}_i$, the penalized inner product was calculated as follows:

$$\text{Similarity}_{\text{penalized}} = \sum x_j,$$

where,

$$x_j = \begin{cases} e_j * F_{ij} & \text{if } e_j, F_{ij} > 0 \\ -e_j & \text{if } e_j > 0, F_{ij} = 0 \\ -F_{ij} & \text{if } e_j = 0, F_{ij} > 0\ . \end{cases} \quad (1)$$

The values associated with rows of **F** with the highest similarity to **e** were considered to be the most likely predictions.

D. Weighting with prior frequency

The similarity scores were analyzed with and without weighting the results using a prior frequency. The weighted score was determined by multiplying the similarity score by the log of the number of times a certain value occurred in the training set. Weighting was used to increase the scores for commonly encountered values.

For an entity **e**, a possible value $\mathbf{F}_i$, and an occurrence count n_v, the weighted similarity score was

$$\text{Similarity}_{\text{wgt}}(\mathbf{e}, \mathbf{F}_i) = \text{Similarity}(\mathbf{e}, \mathbf{F}_i) \times \log_{10}(n_i). \quad (2)$$

where the term n_i was calculated as the number of distinct matching entities in the data.

E. Evaluation of predictions

To evaluate results, we framed the problem domain of identifying useful predictions as sharing features with that of returning relevant results in information retrieval.

We looked at entities that had one or more existing values for the target property, which were in no specific order, and compared these to the ranked predictions for the entities. Calculations around forms of precision accommodated comparison between ranked and unranked results. We therefore used precision to evaluate the performance of our model. See [11], [13], [14] for treatments upon which the following is based.

Precision over unranked sets is computed as

$$\text{Precision} = \frac{|\{\text{retrieved}\} \cap \{\text{relevant}\}|}{|\{\text{retrieved}\}|}. \quad (3)$$

It is also possible to use precision with ranked data. First, one calculates positional precision,

$$\text{Precision(x)} = \text{Precision over first x results.} \quad (4)$$

Then the average positional precision is

$$\sum P(x)\Delta r(x), \quad (5)$$

where $\Delta r(x)$ is the change in recall at ranked position x. Finally, the mean average precision is the mean over the collection of ranked results.

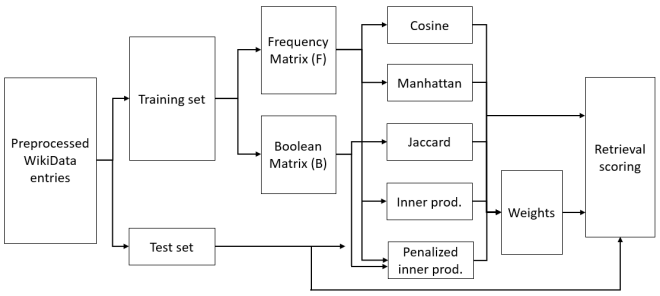

Fig. 4. Process used to test our models for predicting values. This process was repeated for each target property. The retrieval scoring was performed on the unweighted and weighted predictions from each similarity method.

We first calculated the average precision of each test entity. We chose to examine only the top ten predictions under the assumption that users of predictions would rarely look beyond the top ten results. If no predictions in the first ten predictions were correct, the average precision was recorded as zero. We then calculated the mean average precision of all test entities to determine the overall performance. This process was repeated for each different method for determining similarity.

F. Testing methodology

In order to test our models, we split our data into testing and training data sets (Fig. 4). The testing data set was composed of all assertions contained in Wikidata associated with 1,000 randomly selected entities. We chose 'instance of' {P31} for the target property. This property was selected because it is desirable that it be present for entities, it can be difficult for novice contributors to determine, and it can be used to evaluate the overall ontology of Wikidata. The similarity of the rows of the matrices were then determined using the five similarity measures These predictions were then evaluated using our information retrieval metric, mean average precision.

We then repeated this experiment, but required each entity in the test set to have at least ten properties. We performed this experiment because many of the entities in our test set were found to have few properties, which makes prediction difficult since our method relies on matching properties. We again compared the results of different similarity methods using the mean average precision.

We then performed a sensitivity analysis using our best performing similarity score. This analysis measured how sensitive our predictions were to the number of properties of an entity. The average was recorded for each number of properties.

Finally, we identified ten entities in Wikidata that were missing 'instance of' {P31}, referred to as orphan entities. These entities were selected based on having at least a minimal set of properties beside 'instance of' {P31}. We manually assigned a category to each orphan entity, then predicted the most likely value using our best performing method to determine similarity. The manual and predicted values were reviewed for appropriateness.

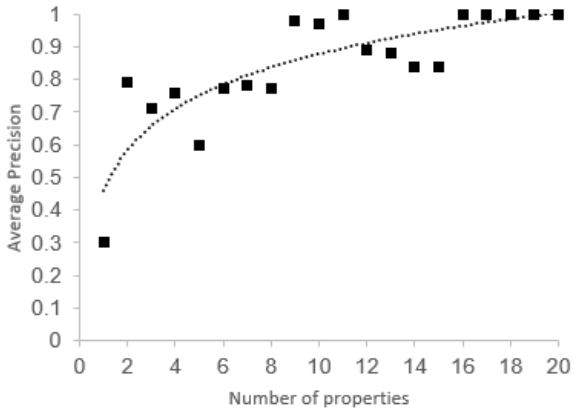

Fig. 5. Mean average precision for predictions based on the number of properties for each entity. The dotted line represents a logarithmic trendline.

V. Results

The training set was found to include 65,920 distinct values for 'instance of' {P31}. The entities that were 'instance of' {P31} with these values had 1,340 unique properties. This resulted in a frequency matrix (**F**) that was 62,920 x 1,340. The 1,000 entities in the test set had 198 unique properties.

The results for each similarity score are shown in Table I. In all cases, the mean average precision was much higher when the predictions were weighted using value frequencies. Cosine similarity and inner product had the best performance, followed by the penalized inner product. The Manhattan similarity method was notable because it essentially did not successfully match any entities correctly. The mean average precision was notably higher in the test set where all entities had at least ten different properties.

TABLE I
Mean average precision for different similarity measures from the test set of 1,000 randomly selected entities.

	One or more properties		Ten or more properties	
Measure	Unweighted	Weighted	Unweighted	Weighted
Cosine	0.064	0.758	0.081	0.957
Manhattan	0.000	0.000	0.000	0.000
Jaccard	0.001	0.009	0.002	0.020
Inner Product	0.017	0.713	0.025	0.951
Penalized	0.052	0.654	0.090	0.946

The average precision increased as the number of properties of test entities increased (Fig 5). The mean found to be very low for entities with only one property and improved significant for entities with two properties. The mean precision was nearly 1 for entities with 15 or more properties.

Our manual categorization of the ten orphan entities showed similarity to the predicted values. In one case, our model predicted that this entity would not have a type specified.

VI. Discussion / Interpretation

Some statistics are drawn from [15].

TABLE II
HUMAN CATEGORIZATION AND PREDICTED VALUES OF 'INSTANCE OF' P31 FOR TEN ORPHAN ENTITIES IN WIKIDATA

Entity number	Human categorization	Model predicted value
27508297	Piece of modern art	Painting
18401540	Threatrical play	Film
31914955	Historical location	House
4206658	Injectable drug	Protein family
18974314	Gene	Embryonic stem cell
3753587	Concert series	Album
39789687	Scientific article	Scholarly article
19536215	Male name	Male given name
37346993	Scientific article	Scholarly article
12323376	Public library system	[no type]

In this paper, we demonstrated our context matrix approach is able to predict the actual 'instance of' {P31} values for entries in the Wikidata. We found this approach to be very dependent on the number of other properties specified for an entity, with marked improvement when there were at least two properties. This approach could be used to develop a property value prediction tool to facilitate addition of new entities and to evaluate the existing ontology.

A. Hierarchical representation of Wikidata

The key engine of our approach is the construction of the Property frequency matrix. For any entity, our goal is to predict its value for a particular property. If we envision an entity as a node on a hierarchical tree, we are interested in finding its parental node, i.e., the type. Such a hierarchical tree could be built using links through the property 'instance of' {P31}, or any property that links entities in principle. The advantage of 'instance of' {P31} is that 98.6% of all entities have this property specified. In comparison, other candidate link properties 'subclass of' {P279} and 'part of' {P361} are only specified for 2.8% and 2.6% of all entities respectively. It should be noted that "instance of", "subclass of", and "part of" are not mutually exclusive.

B. Entity property vector

The formation of the property vector for any entity is relatively straightforward. For any entity, we simply coded 1 for all known properties, 0 for all nonexistent properties. There are instances where an entity has multiple entries for a single property. We did not capture this multiplicity, and it could be explored in future models. Another variation of this approach is to use a penalty, such as -1, for nonexistent properties. This 1/-1 approach will put a high premium on completeness of an entity. Our penalized inner product incorporated this type of penalty, however it was not found to improve performance.

C. Bayesian weighting

Our initial measurements of similarity obtained very low mean average precision. On manual review of the results, we found very rare values were often the highest ranked prediction. Previously, we had observed a very non-uniform distribution of values for 'instance of' {P31} values. For example, 40% of entities had the value 'scholarly article'. One way to incorporate this type of domain knowledge is to include the type distribution as our prior knowledge to weight our predictions, a form of Bayesian approach. The assumption is that new entities will follow a similar entity distribution as the present structure of Wikidata. We considered if distributional bias was already been inherently incorporated into our frequency matrix. However, entries in our frequency matrix have been normalized by rows; hence, it appeared that our frequency matrix lacked this information. The use of the Bayesian prior substantially improved the performance of our models, leading us to believe that it is an important element for such predictions.

D. Model applications

There are at least two applications that we can foresee for the similarity scores. For entities that are already in the Wikidata, similarity scores can serve as a "sanity check". For example, similarity scores between 'Belgium' {Q31} and 'Germany' {Q183} will be close to unity, whereas score between 'Belgium' {Q31} and 'Douglas Adams' {Q42} will be low. These vector representations may be coded to a two dimensional space with approaches such as t-SNE for visualization. The similarity scores may be more beneficial for new entries. In addition to a list of recommended values, the similarity scores will locate comparable entries that are already in Wikidata. For any Wikidata editors, examples from comparable entries will provide valuable guidance and foster uniformity of future entries. In this particular application, our similarity score will be able to provide similar functionality as the established tool ReCoin.

E. Sensitivity to number of properties

Our randomly selected test set achieved fairly modest performance (highest mean average precision: 0.758). A detailed inspection of the results revealed a significant problem in this data. In Wikidata, 11% of entities have only a single unique property, and 61% of entities have three or fewer properties. Our random selection of 1,000 test entities was dominated by entities with a small number of entries. Prediction for these entries is very much like random selection from a huge pool of equally probable outcomes. This led us to create another test set limited to entities with at least 10 properties. Not surprisingly, we found much improved performance with this new data set (highest mean average precision: 0.957). Our sensitivity analysis showed that having at least two properties is critical for prediction.

F. Prediction for orphans

According to Wikimedia, there were 5.5% of entities in Wikidata that were not linked by either 'instance of' {P31} or 'subclass of' {P279} as of Feb 2020. We decided it would be interesting to apply our approach to recommend types for these so-called "orphan" entities. For this analysis, we also decided to exclude entities with 'part of' {P361}, since it can also be considered as a hierarchical property. We discovered a total of 0.87 million such orphans (i.e., without values for {P31},

{P279}, and {P361}) in our downloaded dataset. We selected ten entities from the orphans with at least ten properties. After manual inspection, we categorized each these orphan entities based on the data available in Wikidata. Subsequently, we applied our algorithm to these entities.

The highest scored type is listed in Table II. These results are very encouraging; for example, two scientific papers and a male name were predicted correctly. Three art-related items, a modern art piece, a play, and a concert were very well matched with their predictions. Two health science items were also matched roughly with their types. The item {Q31914955}, predicted as a type of house, was an entry that seemed to refer to a group of historic buildings, although we had a difficult time trying to understand the exact subject of the entry. The last item in our list is the public library system of Copenhagen. This is an abstract entity and our method predicted it should have no type. From these results, it appears that for items that Wikidata already has a large sample size, such as human, scientific articles, and films, our algorithm can produce very reasonable predictions. On the other hand, our algorithm has difficulty making predictions for abstract entities.

VII. Conclusions

This paper demonstrated an approach to predicting the value for properties in Wikidata based on the other properties of an entity. Among the methods tested, we found that cosine similarity had the best performance for matching entities to property values. The mean average precision increased as the number of properties used to describe the entity increased, and we found that weighting the predictions based on the frequency of occurrence was critical to obtaining accurate results. Using this method, we were able to predict values for several entities with unknown values present in Wikidata. Our approach could be used as to augment the creation of new entities or extended to provide methods to evaluate the existing Wikidata ontology.

VIII. Acknowledgments

We would like to thank our faculty advisor, Dr. Rafael Alvarado and the University of Virginia School of Data Science for their guidance throughout our research. We also thank Wiki Education for their support of this project, with a special thanks to Will Kent, the Wikidata Program Manager at Wiki Education.

References

[1] D. Vrandecic, "Wikidata: a new platform for collaborative data collection," *Proceedings of the 21st International Conference on World Wide Web*, 2012.

[2] V. Balaraman, S. Razniewski, and W. Nutt, "Recoin: Relative Completeness in Wikidata," in *Companion of the The Web Conference 2018 on The Web Conference 2018 - WWW '18*. Lyon, France: ACM Press, 2018, pp. 1787–1792, http://www.simonrazniewski.com/wp-content/uploads/2018_Wiki-workshop.pdf. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3184558.3191641

[3] M. Munnecke. Wikidata concept tree generator. [Online]. Available: https://observablehq.com/@repmax/wikidata-concept-tree-generator

[4] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer, "DBpedia – a large-scale, multilingual knowledge base extracted from wikipedia," vol. 6, no. 2, pp. 167–195, publisher: IOS Press. [Online]. Available: https://content.iospress.com/articles/semantic-web/sw134

[5] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: a core of semantic knowledge," in *Proceedings of the 16th international conference on World Wide Web*, ser. WWW '07. Association for Computing Machinery, pp. 697–706. [Online]. Available: https://doi.org/10.1145/1242572.1242667

[6] "Data - W3C." [Online]. Available: https://www.w3.org/standards/semanticweb/data.html

[7] "Wikidata:Requests for comment/Data quality framework for Wikidata - Wikidata." [Online]. Available: https://www.wikidata.org/wiki/Wikidata:Requests_for_comment/Data_quality_framework_for_Wikidata

[8] D. Abián, F. Guerra, J. Martínez-Romanos, and R. Trillo-Lado, "Wikidata and dbpedia: A comparative study," in *Semantic Keyword-Based Search on Structured Data Sources*, J. Szymański and Y. Velegrakis, Eds. Cham: Springer International Publishing, 2018, pp. 142–154.

[9] Y. Huang, V. Tresp, M. Bundschus, A. Rettinger, and H.-P. Kriegel, "Multivariate Prediction for Learning on the Semantic Web," in *Inductive Logic Programming*, ser. Lecture Notes in Computer Science, P. Frasconi and F. A. Lisi, Eds. Berlin, Heidelberg: Springer, 2011, pp. 92–104.

[10] Y. Huang, V. Tresp, M. Nickel, A. Rettinger, and H.-P. Kriegel, "A Scalable Approach for Statistical Learning in Semantic Graphs," *Semantic Web Journal*, vol. 5, Jan. 2013.

[11] "Evaluation measures (information retrieval)," Jan. 2021, page Version ID: 1000641711. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Evaluation_measures_(information_retrieval)&oldid=1000641711

[12] G. G. Chowdhury, "Natural language processing," vol. 37, no. 1, pp. 51–89, _eprint: https://asistdl.onlinelibrary.wiley.com/doi/pdf/10.1002/aris.1440370103. [Online]. Available: https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/aris.1440370103

[13] S. Robertson, "A new interpretation of average precision," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '08*. ACM Press, p. 689. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1390334.1390453

[14] C. Manning, P. Raghavan, and H. Schütze, "Evaluation in information retrieval," in *Introduction to Information Retrieval*, 2009. [Online]. Available: https://nlp.stanford.edu/IR-book/

[15] "Wikidata:Statistics - Wikidata." [Online]. Available: https://www.wikidata.org/wiki/Wikidata:Statistics