ADOG - Anotating Data with Ontologies and Graphs

Daniela Oliveira and Mathieu d'Aquin

Data Science Institute, Insight Centre for Data Analytics, NUI Galway, Ireland
{firstname.lastname}@insight-centre.org

Abstract. ADOG is a system focused on leveraging the structure of a well-connected ontology graph extracted from different Knowledge Graphs to annotate structured or semi-structured data. The Semantic Web Challenge on Tabular Data to Knowledge Graph Matching provided us with the means to test the system within the more restricted scenario of annotating data with a single ontology. This competition provided important insights into the challenges we face not only in a single-ontology case but also in future multi-source scenarios.

Keywords: Knowledge Graphs · Ontologies · DBPedia

1 Presentation of the system

1.1 State, purpose, general statement

ADOG combines a series of existing technologies and algorithms in novel ways to automatically annotate structured and semi-structured files. It takes advantage of the native graph structure of ontologies to built a well-connected network on ontologies from different sources. This integration facilitates the discovery of connections between entities with distinct origins and types, but related topics. More details and a preliminary evaluation of its effectiveness are available in [1].

The Semantic Web Challenge on Tabular Data to Knowledge Graph Matching[1] provided us with a platform to test the base use-case of a single Knowledge Graph (KG) with a single underlying ontology. The challenge distinguished between three separate tasks:

– Column Type Annotation (CTA): a DBPedia ontology class was assigned for each target column.
– Cell-Entity Annotation (CEA): target cells were annonated with DBPedia entities.
– Columns-Property Annotation (CPA): properties were assigned for the relationship between two target columns.

[1] http://www.cs.ox.ac.uk/isg/challenges/sem-tab

We participated in the four Rounds of the competition, and, except in Round 1, we submitted results to all tasks. Except for the CTA task, the remaining tasks were evaluated with F1-measure and Precision. After Round 1, the CTA task adopted a weighted scoring metric. The main score metric was named Average Hierarchical Score (AH-Score) and the secondary measure was called Average Perfect Score (AP-Score)

The following metrics named Average Hierarchical Score (AH-Score) and Average Perfect Score (AP-Score) are calculated for ranking:

$$\text{AH-Score} = \frac{(1 \times |PA|) + (0.5 \times |OKA|) - (1 \times |WA|)}{|\text{Target Columns}|} \tag{1}$$

where $|PA|$ is the number of Perfect Annotations, $|OKA|$ is the number of Correct Annotations, and $|WA|$ is the number of Wrong annotations.

$$\text{AP-Score} = \frac{|PA|}{|\text{Annonated Classes}|} \tag{2}$$

1.2 Specific techniques used

ADOG takes advantage of the graph properties of the ontologies and the KG by enriching the links between nodes and, therefore, provide a new level of relatedness connections between concepts in the KG. At this stage, the system leverages the depth of the concepts in the ontology, i.e., the distance to the root node, and the shortest paths between nodes to distinguish between stronger or weaker candidate annotations.

Figure 1 shows the three steps needed to build the schema layer that includes the ontology graph, native links and discovered relatedness edges.

The first step parses the ontology files and entities of the KG. The system is designed to integrate multiple ontologies via their `owl:subclassOf` and object properties. Additionally, the ontologies are subjected to an ontology matching step to explore additional relations between ontology classes.

The ontology graph and all its links are loaded into ArangoDB[2], a multi-model NoSQL database. We chose this database due to its multi-model capabilities, which allowed us to use graph and document models to store each ontology class as a node in a graph that can be described as a document with key/value properties. This database contained all information relevant to the schema. Each node saved the relevant information of a class (uri, label, definition), and also the distance to the root node, i.e., depth. The database also includes a document collection of relevant properties of the graph such as diameter (maximum distance between two nodes in the graph), maximum Inverse Document Frequency (IDF), and maximum depth.

The relevant entity properties of the KG are indexed with ElasticSearch[3]. The only mandatory property to be indexed is a label of entity that can be matched against the data to be annotated.

[2] https://www.arangodb.com
[3] https://www.elastic.co/products/elasticsearch

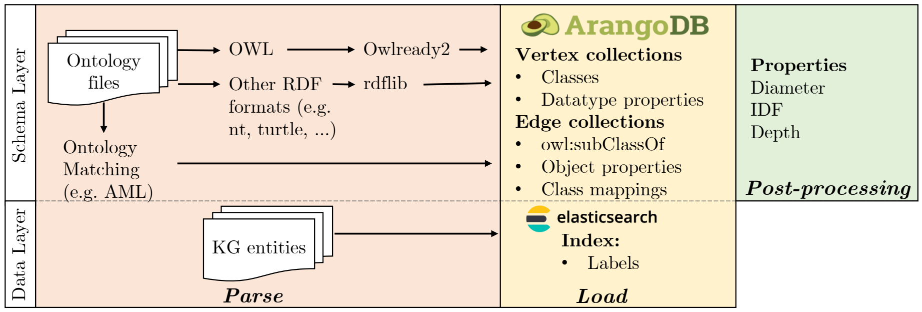

Fig. 1. Building steps of the schema and data layers.

After the build stage is complete, the matching process can start by matching the data against the ElasticSearch index. When several matches are returned from the matching process, additional measures are employed to score the relevance of each match to the query, considering the context of the data to annotate. The three main steps are calculating the similarity and frequency of properties measures, and final score weighting.

Similarity This measure finds the string similarity between query words and the matched terms. Both strings are normalised, punctuation is removed, and word inside brackets are ignored. The similarity measure uses Levenshtein Distance (LD) to calculate the similarity between $s1$ and $s2$ as follows:

$$sim = 1 - \left(\frac{LD(s1, s2)}{max(length(s1), length(s2))} \right) \tag{3}$$

Frequency of Properties If any extra properties, besides the labels, were indexed from the source KG, this step calculates and normalises their frequencies for each match. For example, in DBPedia, these properties can be the categories, types or even other entities linked to the matched entity via an object property.

Final Score The final score of each candidate will be weighted considering the previous steps, plus the normalised ElasticSearch score for each search performed. These weights are variable and can be adjusted to fit any model, giving more or less weight to similarity, search scores, or property frequencies.

1.3 Adaptations made for the evaluation

The main aim of ADOG is integrating ontologies and KGs from different sources, however, it is still possible to use the system with a single ontology and KG. In the case of the present challenge, the system is using DBPedia as the Knowledge Graph and the DBPedia ontology as the schema. We adopted different approaches for Round 1 and Round 2, detailed in the following sections.

Round 1 In Round 1, we focused on the CTA task, and therefore, the build stage was central to the methodology. Since the challenge includes only one ontology, we matched it against itself to find possible missing relations that do not have to denote equivalence but can only indicate a degree of relatedness between the two concepts. For example, in the DBPedia ontology (dbo) the class `http://dbpedia.org/ontology/MovieDirector` is not directly connected with the class `http://dbpedia.org/ontology/Film`. Instead the class `dbo:Film` has the property `http://dbpedia.org/ontology/director` connecting `dbo:Film` to `dbo:Person`. Partially matching `dbo:MovieDirector` with `dbo:Film` created a direct mapping between these two classes that could more easily help identify related matches for both classes.

Since we focused mostly matching the columns to ontology classes, the data layer was kept shallow, only indexing the labels of each resource from each language available in the DBPedia data dumps. As scoring properties, we used ontology type frequency and pair-wise shortest path computation between candidate ontology classes.

Round 2/3/4 In the remaining Rounds, we worked on improving the results of the CEA task, and therefore, the focus was on the Data Layer. The Schema Layer was not changed, while the Data Layer was updated to include more information from DBPedia resources that facilitates the choice of right match between query word and matched label. In addition to the previous properties, resource categories were indexed and the IDF of the categories and types was added to weight the frequencies with tf-idf.

1.4 Link to the system and parameters file

The code used for completing the challenge is available in https://github.com/danielapoliveira/iswc-annotation-challenge. Instructions to run are also contained in the repository.

2 Results

Table 1 shows the overall results for Rounds 1, 2, 3, and 4. Even though Round had distinct datasets and, in some cases, even different evaluation metrics, the side-by-side comparison gives an overall view of the performance of our system in this challenge.

2.1 Round 1

In Round 1 we focused on the CTA task, but also submitted a results for the CEA task. We did not submit to the CPA task in this Round. Most of the performance enhancements were focused on improving the column type annotations, therefore, the system performed better in CTA, then CEA.

Table 1 shows that for CTA the F1-Score obtained was 0.908, with 0.915 precision, while in CEA we obtained a F1-Score of 0.657, with 0.673 precision.

Round	CTA		CEA		CPA	
	AH-Score	AP-Score	Precision	Recall	Precision	Recall
1	0.908	0.915	0.657	0.673	-	-
2	0.713	0.673	0.742	0.745	0.459	0.708
3	1.409	0.238	0.912	0.913	0.558	0.763
4	1.538	0.296	0.835	0.838	0.750	0.767

Table 1. Results for the three tasks in every Round. For Round 1, in the CTA task, the metrics used were Precision and Recall, instead of AH-Score and AP-Score, respectively.

2.2 Round 2/3/4

In Round 2, 3, and 4, we mostly focused on the CEA task but submitted to all tasks. The CTA task had different scoring and, therefore, is not comparable to the results of Round 1.

In these Rounds, both CTA and CPA were obtained from the CEA results since the methods we used allowed us to directly extract all the necessary information from CEA's results. For the CTA results no changes were necessary, while for CPA a few changes were added to extract the correct relation between the elements matched by the CEA algorithm. The CEA task had improved results in with the new ground truth and refined methods.

3 General comments

3.1 Comments on the results

ADOG is still in early stages of research and development and we took advantage of this challenge as a concrete testbed for research into the single-ontology use-case. Despite being focused on the multiple source scenario, the system still achieved a reasonable performance without many modifications to its core function. However, throughout the competition we were faced with a few challenges. In its current state, the system is very sensitive to scoring and weight changes, i.e., even small changes can have a big impact or changes that benefit a type of data, hinder other types.

3.2 Discussions on the way to improve the proposed system

The main research in the future will be focused on the multi-source system. However, a more robust scoring system is necessary before adding an extra step of complexity. Adding extra KGs and schema could lead to a performance improvement since the graph capabilities of the approach could be further explored. In the future, we also intend to focus more on the property annotation task since that is also one of the overall goals of our system.

3.3 Comments on the challenge procedure

We believe that a system should be organised so that the results submitted and shown on the leaderboards can be double-checked for accuracy. Also, to avoid overfitting to the ground truth, we suggest that the systems are tested against another test set generated by the same methods. Finally, we believe that a more standard and robust method of generating the ground truth is necessary since issues around inconsistencies, different encodings, and several instances of incorrect ground truth data can generate frustration for participants, making the competition less appealing.

3.4 Comments on the challenge measures

We consider that the measures used for CTA in Round 2 are not appropriate to accurately evaluate the performance of an algorithm. A participant that obtains all perfect results without modifying their system should not be forced to add every parent of the right match just to fit the challenge. Instead, we would suggest a different weight measure, where the class assigned by the algorithms is weighted based on their distance from the perfect match. For example, if the exact match was `dbo:MovieDirector` and a result is submitted with `dbo:Person`, this match should get a score of 0.5 instead of 1. If the exact match is found, then the score for the match is 1. In this way, a single match would not have multiple answers, and the total scores are bound from 0 to 1.

4 Conclusions

Overall the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching provided an engaging platform for developing and testing our system. The system expanded its functionalities due to the demands of this challenge and participating provided important insights into the hurdles we are faced when dealing with data annotation based on KGs.

Acknowledgements

This work has been partly funded by Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289_P2, Insight Centre for Data Analytics.

References

1. Oliveira, D., Sahay, R., d'Aquin, M.: Leveraging Ontologies for Knowledge Graph Schemas. In: Knowledge Graph Building Worshop. p. 12 (2019), https://openreview.net/pdf?id=B1xnsmvaUE