Domain-agnostic Construction of Domain-Specific Ontologies

Rosario Uceda-Sosa, Nandana Mihindukulasooriya
IBM Research AI
Yorktown Heights, NY, USA
rosariou@us.ibm.com, nandana.m@ibm.com

Atul Kumar
IBM Research AI
IBM India Research Lab, India
kumar.atul@in.ibm.com

Abstract

In this poster, we present a novel approach to construct domain-specific ontologies with minimal user intervention from open data sources. We leverage techniques such as clustering and graph embeddings and show their usefulness in information retrieval tasks, thus reinforcing the idea that knowledge graphs and DL can be complementary technologies.

1 Introduction

Knowledge graphs (KGs) have become the representational paradigm of choice in general AI tasks, and in most cases they leverage existing open knowledge graphs such as Wikidata [1] or DBPedia [2]. However, it is not clear whether the success of these general KGs extrapolates to specialized, technical domains where a ready-made ontology doesn’t exist and a generic large graph such as Wikidata, may not be efficient for processing and may introduce ambiguities in entity resolution and linking tasks.

In this poster, we discuss how to create a systematic, domain-agnostic pipeline to build non trivial domain-specific ontologies from open knowledge sources (Wikidata, DBPedia, Wikipedia) from a reference ontology with minimal user intervention. We show how these domain-specific ontologies have a symbiotic relationship with Deep Learning (DL). DL (as well as Semantic Web) technologies help to improve ontology coverage with respect to a baseline forest population and, at the same time, these ontologies greatly increase the accuracy of DL models in tasks like term ranking.

We illustrate our findings in the IT Operations domain with a new industrial-sized dataset of Lenovo Tips troubleshooting documents.

2 Related Work

When we decided to create an application ontology for the IT Operations domain, we looked to ontologies in related technical areas [3], [4], [5], [6], however none of these refer to a generic pipeline to quickly build specialized knowledge graphs. While efforts to induce an ontology from text with more or less supervision have been around for ten years [7], [8], [9], [10], validating the quality of the resulting ontology for Semantic Web (i.e., query) applications is hard. We are also aware of efforts to create simple Wikidata inheritance hierarchies [11] to leverage LOD in order to populate ontologies [12], or to classify Knowledge Organization systems [13]. None of these approaches allow us to create a new, rich ontology on a specialized domain that extends well curated resources like Wikidata.

Reference ontologies have been identified as a viable tool to simplify and speed up the construction of application ontologies in a variety of domains. Up to now, most reference ontologies do not capture the existing upper ontologies of popular LOD graphs [14], [15], [16], which makes it difficult to leverage them. Our reference ontology, the Universal Upper Ontology (U2O), links to Wikidata’s upper ontology concepts, like people, places, temporal entities, institutions, products,

services, information and documents. It also links to linguistic representations ([17], [18], [19]) of statements and verbal propositions. This means U2O can represent both the entities extracted from the LOD cloud as well as statements from text corpora using Natural Language Processing (NLP) and information extraction techniques.

Representing RDF knowledge graphs in a vector space has gained a lot of attention over the past few years [20] and these are influenced by the generic graph embedding algorithms such as DeepWalk [21] and Deep Graph Kernels [22]. In this work, we are using the pre-trained graph embeddings for Wikidata that are provided by [23].

3 Construction of a domain-specific ontology

Wikipedia and Wikidata (60 million entities) are comprehensive, curated open knowledge sources with entities and relations about many domains. We’ve built a domain-agnostic service which extracts related entities and extends them with minimal user curation. We describe the steps of this pipeline below.

**U2O, the Universal Upper Ontology.** The Universal Upper Ontology (U2O), captures domain-independent entities and properties which overlap with those of Wikidata upper ontology concepts. In particular, the U2O vocabulary includes generic entities (instances of u2o:DomainEntity), like Person, Company, Event, Standard, Geographic Location, and equates them with those in Wikidata. The second type of entities in U2O are linguistic concepts (see 2), like statements and verb propositions so text corpora instances can be linked to LOD resources.

General-purpose relations (or properties) are also part of the U2O vocabulary and are organized in hierarchies. In particular, the asymmetric, transitive relation u2o:subConceptOf is defined as the fixpoint of the union of instanceOf (wdt:P31) and subClassOf (wdt:P279) in Wikidata.

All entities in an application ontology (ITOPS in our example) are formally defined as instances of u2o:DomainEntity and related by the u2o:subConcept relation. Not differentiating between classes and instances gives us the flexibility of adding new instances which may become concepts on their own right in further extensions. This way, we’re not committing to a fixed, unmovable T-Box.

**Partial Forest Population.** We automatically extract a connected, self-contained Wikidata subgraph and translate its model [15] into an RDF/OWL ontology that extends the U2O reference ontology above from a seed list of concepts. In the case of the IT Operations (ITOPS) ontology, 50 concepts, worth 5 hours of human curation are used (see 1). The population service allows for specific concepts to NOT be populated. For example, ITOPS does not include instances of videogames or RFC (Wikidata ID Q212971), which is a subconcept of Internet Standard (wd:Q290378).

Formally, given a concept set \( U = \{ C_1, \ldots, C_n \} \) we extract a graph \( G_U = \{ V, E \} \) of vertices and edges, where \( E \) are sub-properties of Wikidata Property (wdt:Q18616576) and \( V \) is made up of Wikidata items \( I_t \) such that \( I_t \) (wdt:P279 wdt:P31) \( C_j \) in \( U \), using a SPARQL clause. All items...
in $V$ are defined as instances (rdf:type) of u2o:DomainEntity and the u2o:subConceptOf creates the IS-A hierarchy.

Furthermore, if there’s a relation between an item in $V$ and another item in Wikidata not in $V$ the target item is substituted by a string with its label value, to avoid dangling references. In the ITOPS case study, this step results in a self-contained ontology of 32,800 entities (2,000 entities has subconcepts of their own).

**Wikipedia category-based ontology augmentation.** Most Wikipedia articles are manually labeled with tags corresponding to categories by human contributors. As this is done by Wikipedia editors (which is a much larger group than Wikidata editors), such tags generally contain information not in Wikidata. We obtain categories of interest from the seed knowledge graph above and their associated Wikipedia articles. It is worth noting that these categories are also organized in a hierarchy.

Once the categories of interest were identified, we acquired all entities associated with those categories and calculate several metrics that indicate the heterogeneity of these categories and their overlap with the seed ontology. Finally, we train a binary classifier to categorize whether a given category corresponding to a domain specific concept is in the correct level of granularity, i.e., all its members can be included in the IT ontology using data from the previous step as training data. This step identified 9700 new entities. After manually curation of low scored entities, resulting in 8250 entities were added to the existing ontology.

**Graph embeddings-based ontology augmentation** This step extends the ontology in a similar way to the previous approach but using graph embeddings, that is, encoding a subsymbolic representation of the graph in a vector space in which distances between vectors associated to each node correspond to the occurrences of graph edges. The Wikidata RDF representation can be treated as a large directed graph of entities and encoded as graph embeddings. In this poster, we have used pre-trained Wikidata graph embeddings from PyTorch-BigGraph [23].

The goal of this step is to identify the relevant entities that belong to the domain ontology which were missed in the first two steps. We do this in two phases. In the expansion phase we take the output of the previous step as the input and expand it by taking the $k$ nearest neighbours of each domain-specific entity and adding them to an intermediate graph. In the second, pruning phase the intermediate graph is clustered to $N$ clusters using the k-means clustering algorithm. Then each cluster is analyzed to calculate the percentage of entities from the seed ontology. If that percentage is above a predefined threshold, the new entities are included in the graph, otherwise they are ignored. This step produces 1,871 new entities with clusters that have more than 75% old entities.

**Linking to DBPedia definitions** We finally link definitions and infobox information from DBPedia to add add large, focused text fragments to be used in propositionalizations and embeddings.
Table 1: Methods for term ranking compared using the average precision with respect to gold standard terms.

<table>
<thead>
<tr>
<th>Approach</th>
<th>Avg. Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>TTF</td>
<td>0.029</td>
</tr>
<tr>
<td>ATTF</td>
<td>0.003</td>
</tr>
<tr>
<td>TTF-IDF</td>
<td>0.032</td>
</tr>
<tr>
<td>RIDF</td>
<td>0.035</td>
</tr>
<tr>
<td>CValue</td>
<td>0.039</td>
</tr>
<tr>
<td>Chi-Square</td>
<td>0.020</td>
</tr>
<tr>
<td>RAKE</td>
<td>0.019</td>
</tr>
<tr>
<td>Basic</td>
<td>0.037</td>
</tr>
<tr>
<td>ComboBasic</td>
<td>0.036</td>
</tr>
<tr>
<td>CValue + ITOPS</td>
<td>0.121</td>
</tr>
</tbody>
</table>

Figure 3: Term ranking results using ITOPS (in red), the graph shows how many gold standard terms were found at each cut-off N.

4 Evaluation

As the base dataset for testing, we are using 4,000 troubleshooting documents from Lenovo\footnote{https://support.lenovo.com/us/en/solutions/ht503909}, we are working on publishing the subset that was used in the experiments. A tool developed in a separate project was used to extract terms associated with the titles. This tool used the open source library spaCy\footnote{https://spacy.io} and IBM’s Watson Natural Language Understanding (NLU)\footnote{https://www.ibm.com} to extract and filter terms based on domain specific manually curated dictionaries. For example, the title “No display on external monitor - ideapad and ideacentre”, becomes “display”, “external monitor”, “ideapad”, “ideacentre”. An average of 4.2 terms per title are produced.

This output helps us assess the coverage of the ITOPS ontology. A strict search of name, rdfs:label and aliases of terms, reveals that 86% of titles match one or more entities to ITOPS, while 56% of the titles match two or more and 38% match three or more. Furthermore, 59% of all terms in all titles match an ITOPS entity. In the example above, we match “display” and “ideapad”. “External monitor” is not matched, even though ITOPS has the concept "monitor". Ideacentre is an isolated node in Wikidata and categorized in Wikipedia as ‘Lenovo’. Given that this category is heterogeneous, it hasn’t been vetted by our algorithm. This example shows some of the current limitations of our implementation.

Despite the incomplete coverage, an ontology like ITOPS is useful in several IR tasks, such as terminology ranking. A second (unrelated) project focused on inducing concepts and instances from the corpus above, has leveraged ITOPS as a way to improve dramatically the quality of their term ranking as shown in Table 1 and Figure 3. In this experiment, the task is to rank the terms automatically extracted from the aforementioned dataset and a manually curated list of 480 domain terms were used as the gold standard. Several state of the art methods have been used to rank these terms but the use of the ITOPS ontology improves over these.

5 Conclusions and Future Work

Our preliminary results indicate that resources like Wikidata, Wikipedia and DBPedia can be used to create meaningful, specialized ontologies with minimal user intervention. These are rich enough to provide a good seed knowledge for real-life text corpora and improve the performance of ML/DL tasks, like term ranking.

However, open source data is not enough to provide a full specialized ontology. We are currently working on the ingestion of terms from glossaries, product catalogs and other domain-specific resources where both ontology alignment and integration are needed.

More work is also needed to evaluate the usefulness of a domain-specific ontology with respect to Wikidata, Wikipedia and DBPedia taken as a whole, especially in Deep Learning applications.
References


