A Simple Method for Inducing Class Taxonomies in Knowledge Graphs

Marcin Pietrasik and Marek Reformat

Department of Electrical and Computer Engineering, University of Alberta

1. Introduction

Knowledge graphs are data storage structures that rely on principles from graph theory to represent information. Specifically, facts are stored as triples which bring together two entities through a relation. In a graphical context, these entities are analogous to nodes, and the relations between them are analogous to edges. Figure 1 illustrates a small knowledge graph which relates Leonardo da Vinci to his birth city, Vinci.

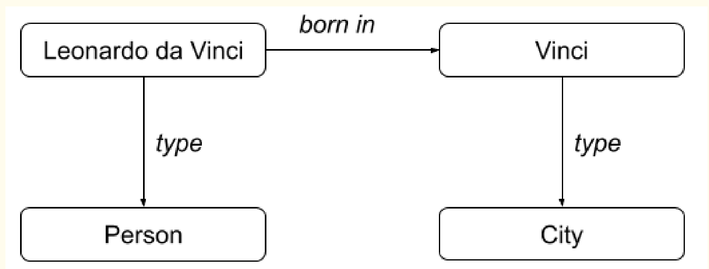

Figure 1:Toy example of knowledge graph relating Leonardo da Vinci to his birth city.

Knowledge graphs find uses in personal, academic, and commercial domains and are ubiquitous in the research fields of the Semantic Web, artificial intelligence, and computer science broadly. Furthermore, private companies are known to use proprietary knowledge graphs as a component of their data stores. Google, for instance, uses a knowledge graph to enhance their search engine results by providing infoboxes which summarize facts about a user's query [1]. This widespread use of knowledge graphs motivates us to investigate the practical issues that come up when they are deployed in real-world scenarios.

2. Problem

Ontologies are often used in conjunction with knowledge graphs to provide an axiomatic foundation on which knowledge graphs are built. In this view, an ontology may be seen as a rule book that provides semantics to a knowledge graph and governs how the information contained within it can be reasoned with. One of the core components of an ontology is the class taxonomy: a set of subsumption axioms between the type classes that may exists in the knowledge graph. When put together, the subsumption axioms form a hierarchy of classes where general concepts appear at the top and their subconcepts appear as their descendants. Figure 2 provides an example of a class taxonomy.

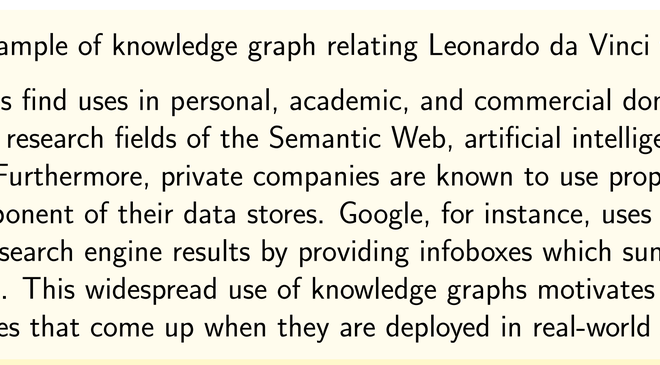

Figure 2:Toy example of class taxonomy.

One of the challenges that arise when working with large knowledge graphs is that of class taxonomy construction. Manual construction is time consuming and requires curators knowledgeable in the area. On the other hand, automated methods are not able to induce class taxonomies of the quality necessary to reliably apply to complex knowledge graphs. Furthermore, they oftentimes rely on external information which may itself be manually curated or may only be applicable to knowledge graphs in a particular domain. With this in mind, the impetus for automatically inducing class taxonomies of high quality from large-scale knowledge graphs becomes apparent.

3. Preliminaries

A knowledge graph, $\mathcal{K}$, is repository of information structured as a collection of triples where each triple relates the subject, s, to the object, o, through a relation, r. More formally, $\mathcal{K} = \{\langle s, r, o\rangle \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}\}$ where $\langle s, r, o\rangle$ is a triple, $\mathcal{E}$ is the set of entities in $\mathcal{K}$, and $\mathcal{R}$ is the set of relations in $\mathcal{K}$. $\mathcal{K}$ can therefore be viewed as a directed graph with nodes representing entities and edges representing relations.
We can think of relation-object pairs, $\langle r, o\rangle$, as tags that describe the subject. In this view, each entity that takes on the role of subject, s_i, is annotated by tags, $t_j \in \mathcal{A}_i$, where $\mathcal{A}_i$ is the set of tags that annotate s_i. We call these entities documents, $d_i \in \mathcal{D}$, such that the set of all documents is a subset of all entities, $\mathcal{D} \subseteq \mathcal{E}$. Tags are defined as relation-objects pairs, $t := \langle r, o\rangle$, and belong to the set of all tags, the vocabulary, denoted as $\mathcal{V}$, such that $t_j \in \mathcal{V}$. In this view, the knowledge graph $\mathcal{K}$ may be represented as the set of document-tag tuples $\mathcal{K} = \{\langle d, t\rangle \in \mathcal{D} \times \mathcal{V}\}$, where $\langle d, t\rangle$ is the tuple that relates document d with tag t. This tuple structure provides the input for our class taxonomy induction method.

4. Solution

Before describing the taxonomy induction procedure for our method, we define measures which are calculated on the knowledge graph as required input for our algorithm.

- The number of documents annotated by tag t_a is denoted as D_{t_a}.
- The number of documents annotated by both tags t_a and t_b is denoted as D_{t_a,t_b}. We note that this measure is symmetrical, i.e. $\mathrm{D}_{t_a,t_b} = \mathrm{D}_{t_b,t_a}$.
- The generality of tag t_a, G_{t_a}, measures how general the concept described by the tag is and how high it belongs in the taxonomy. The generality is defined as:

$$\mathrm{G}_{t_a} = \sum_{t_b \in \mathcal{V}_{-t_a}} \frac{\mathrm{D}_{t_a,t_b}}{\mathrm{D}_{t_b}} \tag{1}$$

Where $\mathcal{V}_{-t_a}$ is the set of all tags excluding tag t_a.
Having calculated the aforementioned measures, we proceed by sorting tags in the order of descending generality and store them as $\mathcal{V}_{sorted}$. The first element of this list, $\mathcal{V}_{sorted}[0]$, is semantically the most general of all tags and becomes the root tag of the taxonomy.
The taxonomy, $\mathcal{T}$, is represented as a set of subsumption axioms between parent and child tags. Formally, each subsumption between parent tag, t_{parent}, and child tag, t_{child}, is represented by $\{t_{parent} \rightarrow t_{child}\}$ such that $\{t_{parent} \rightarrow t_{child}\} \in \mathcal{T}$. The taxonomy is therefore initialized with the root tag as $\mathcal{T} = \{\{\emptyset \rightarrow \mathcal{V}_{sorted}[0]\}\}$ where $\emptyset$ represents a null value, i.e. no parent.
Following initialization, the remaining tags are added to the taxonomy in terms of descending generality by calculating the similarity between the tag being added, t_b, and all the tags already in the taxonomy, $\mathcal{T}*$. The tag $t_a \in \mathcal{T}*$ that has the highest similarity with tag t_b becomes the parent of t_b and $\{t_a \rightarrow t_b\}$ is added to $\mathcal{T}$. The similarity between tags t_a and t_b, $\mathrm{S}_{t_a \rightarrow t_b}$, measures the degree to which tag t_b is the direct descendant of tag t_a. It is calculated as the degree to which tag t_b is compatible with tag t_a and all the ancestors of t_a:

$$\mathrm{S}_{t_a \rightarrow t_b} = \sum_{t_c \in \mathcal{P}_{t_a}} \alpha^{l_a - l_c} \frac{\mathrm{D}_{t_b,t_c}}{\mathrm{D}_{t_b}} \tag{2}$$

Where $\mathcal{P}_{t_a}$ is the path in the taxonomy from the root tag $\mathcal{V}_{sorted}[0]$ to tag t_a. l_a and l_c denote the levels in the hierarchy of tags t_a and t_c, respectively. The levels are counted from the root tag starting at zero. Thus, the level of $\mathcal{V}_{sorted}[0]$, denoted as $l_{\mathcal{V}_{sorted}[0]}$, is equal to zero, the levels of its children are equal to one, and so on. The decay factor, α, is a hyperparameter that controls the effect ancestors of tag t_a have on its similarity when calculating $\mathrm{S}_{t_a \rightarrow t_b}$. By setting the value of α such that $0 < \alpha < 1$, we ensure that the effect is lower the more distant an ancestor tag is. The cases were $\alpha = 0$ and $\alpha = 1$ correspond to ancestors having no effect and equal effect on the similarity, respectively.

5. Results

We applied our method to three real-world datasets: Life [2], DBpedia [3], and Wordnet [4]. Once the subsumption axioms for each dataset were generated, we compared our induced taxonomy to the gold standard taxonomy for each dataset. We used the confusion matrix to derive the harmonic mean between precision and recall, the F_1 score, as our evaluation metric. Table 1 summarizes these results.

	Life			DBpedia			WordNet		
Method	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1
Heymann and	–	–	–	.7944	.8021	.7982	.6027	.5814	.5918
Garcia-Molina	–	–	–	**±.01483**	**±.01500**	**.01490**	**±.0116**	**±.0112**	**±.0114**
Schmitz	.8936	.7966	.8423	.8063	.7962	.8013	.8140	.7756	.7943
	±0	**±0**	**±0**	**±0**	**±0**	**±0**	**±0**	**±0**	**±0**
Paulheim and	–	–	–	.1040	.2190	.1410	–	–	–
Fümkranz	–	–	–	–	–	–	–	–	–
Ristoski	–	–	–	.5940	.4650	.5210	–	–	–
et al.	–	–	–	–	–	–	–	–	–
Völker and	–	–	–	.9920	.9970	.9950	–	–	–
Niepert	–	–	–	–	–	–	–	–	–
Our method	.8740	.8513	.8625	.8781	.8867	.8824	.7275	.7018	.7144
	±.0041	**±.0040**	**±.0040**	**±.0051**	**±.0052**	**±.0052**	**±.0070**	**±.0068**	**±.0069**

Table 1:Method results (mean ± standard deviation) on the aforementioned datasets.

Although a thorough comparison between our model and existing methods for class taxonomy induction is omitted here for brevity, we note that our model outperforms or is competitive with these methods.

6. References

[1] Amit Singhal.
Introducing the knowledge graph: things, not strings, 2012.

[2] Yuri Roskov et al.
Species 2000 itis catalogue of life, 2019 annual checklist.
2019.

[3] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al.
Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia.
Semantic Web, 6(2):167–195, 2015.

[4] George A Miller.
Wordnet: a lexical database for english.
Communications of the ACM, 38(11):39–41, 1995.