la

What are the Important Properties of an Entity?

Comparing Users and Knowledge Graph Point of View

Ahmad Assaf[1], Ghislain A. Atemezing[1], Raphaël Troncy[1] and Elena Cabrio[1,2]

[1] EURECOM, Sophia Antipolis, France. <firstName.lastName@eurecom.fr>
[2] INRIA, Sophia Antipolis, France. <elena.cabrio@inria.fr>

Abstract. Entities play a key role in knowledge bases in general and in the Web of Data in particular. Entities are generally described with a lot of properties, this is the case for DBpedia. It is, however, difficult to assess which ones are more "important" than others for particular tasks such as visualizing the key facts of an entity or filtering out the ones which will yield better instance matching. In this paper, we perform a reverse engineering of the Google Knowledge graph panel to find out what are the most "important" properties for an entity according to Google. We compare these results with a survey we conducted on 152 users. We finally show how we can represent and explicit this knowledge using the Fresnel vocabulary.

Keywords: Entities, Google Knowledge Graph, visualization, knowledge extraction

1 Introduction

In many knowledge bases, entities are described with numerous properties. However, not all properties have the same importance. Some properties are considered as keys for performing instance matching tasks while other properties are generally chosen for quickly providing a summary of the key facts attached to an entity. Our motivation is to provide a method enabling to select what properties should be used when depicting the summary of an entity, for example in a multimedia question answering system such as QakisMedia[3] or in a second screen application providing more information about a particular TV program[4].

Our approach consists in: (i) reverse engineering the Google Knowledge Panel by extracting the properties that Google considers as sufficiently important to show (Section 2), and (ii) analyzing users' preferences by conducting a user survey and comparing the results (Section 3). We finally show how we can explicitly represent this knowledge of preferred properties to attach to an entity using the Fresnel vocabulary before concluding (Section 4).

[3] `http://qakis.org/`

[4] `http://www.linkedtv.eu/demos/linkednews/`

2 Reverse Engineering the Google KG Panel

Web scraping is a technique for extracting data from Web pages. We aim at capturing the properties depicted in the Google Knowledge Panel (GKP) that are injected in search result pages [1]. We have developed a Node.js application that queries all DBpedia concepts that have at least one instance which is `owl:sameAs` with a Freebase resource in order to increase the probability that the search engine result page (SERP) for this resource will contain a GKP. Moreover, we filter out generic concepts by excluding those who are direct subclasses of `owl:Thing` since they will trigger ambiguous queries. We obtained a list of 352 concepts[5].

For each of these concepts, we retrieve n instances. In our experiment, n was equal to 100 random instances. For each of these instances, we issue a search query to Google containing the instance label. Google does not serve the GKP for all user agents and we had to mimic a browser behavior by setting the $User-Agent$ to a particular browser. We use CSS selectors to check the existence of and to extract data from a GKP. An exemple of a query selector is *._om* (all elements with class name *_om*) which returns the property DOM element(s) for the concept described in the GKP. From our experiments, we found out that we do not always get a GKP in a SERP. If this happens, we try to disambiguate the instance by issuing a new query with the concept type attached. However, if no GKP was found again, we capture that for manual inspection later on. Listing 1 gives the high level algorithm for extracting the GKP. The full implementation can be found at `https://github.com/ahmadassaf/KBE`.

Algorithm 1 Google Knowledge Panel reverse engineering Algorithm

```
INITIALIZE $equivalentClasses(DBpedia, Freebase)$ AS $vectorClasses$
Upload $vectorClasses$ for querying processing
Set $n$ AS number-of-instances-to-query
for each $conceptType \in vectorClasses$ do
    SELECT $n$ instances
    $listInstances \leftarrow$ SELECT-SPARQL($conceptType$, $n$)
    for each $instance \in listInstances$ do
        CALL http://www.google.com/search?q=$instance$
        if $knowledgePanel$ exists then
            SCRAP GOOGLE KNOWLEDGE PANEL
        else
            CALL http://www.google.com/search?q=$instance + conceptType$
            SCRAP GOOGLE KNOWLEDGE PANEL
        end if
        $gkpProperties \leftarrow$ GetData(DOM, EXIST(GKP))
    end for
    COMPUTE ocurrences for each $prop \in gkpProperties$
end for
return $gkpProperties$
```

[5] SPARQL query: `http://goo.gl/EYuGm1`

3 Evaluation

We conducted a user survey in order to compare what users think should be the important properties to display for a particular entity and what the GKP shows.

User survey. We set up a survey[6] on February 25th, 2014 and for three weeks in order to collect the preferences of users in term of the properties they would like to be shown for a particular entity. We select one representative entity for nine classes: `TennisPlayer`, `Museum`, `Politician`, `Company`, `Country`, `City`, `Film`, `SoccerClub` and `Book`. 152 participants have provided answers, 72% from academia, 20% coming from the industry and 8% having not declared their affiliation. 94% of the respondents have heard about the Semantic Web while 35% were not familiar with specific visualization tools. The detailed results[7] show the ranking of the top properties for each entity. We only keep the properties having received at least 10% votes for comparing with the properties depicted in a KGP. Hence, users do not seem to be interested in the `INSEE code` identifying a French city while they expect to see the `population` or the `points of interest` of this city.

Comparison with the Knowledge Graphs. The results of the Google Knowledge Panel (GKP) extraction[8] clearly show a long tail distribution of the properties depicted by Google, with a top N properties (N being 4, 5 or 6 depending on the entity) counting for 98% of the properties shown for this type. We compare those properties with the ones revealed by the user study. Table 1 shows the agreement between the users and the choices made by Google in the GKP for the 9 classes. The highest agreement concerns the type `Museum` (66.97%) while the lowest one is for the `TennisPlayer` (20%) concept.

Classes	TennisPlayer	Museum	Politician	Company	Country	City	Film	SoccerClub	Book
Agr.	20%	66.97%	50%	40%	60%	60%	60%	50%	60%

Table 1. Agreement on properties between the users and the Knowledge Graph Panel

With this set of 9 concepts, we are covering 301, 189 DBpedia entities that have an existence in Freebase, and for each of them, we can now empirically define the most important properties when there is an agreement between one of the biggest knowledge base (Google) and users preferences.

Modeling the preferred properties with Fresnel. Fresnel[9] is a presentation vocabulary for displaying RDF data. It specifies *what* information contained in

[6] The survey is at `http://eSurv.org?u=entityviz`

[7] `https://github.com/ahmadassaf/KBE/blob/master/results/agreement-gkp-users.xls`

[8] `https://github.com/ahmadassaf/KBE/blob/master/results/survey.json`

[9] `http://www.w3.org/2005/04/fresnel-info/`

an RDF graph should be presented with the core concept `fresnel:Lens` [2]. We use the Fresnel and PROV-O ontologies[10] to explicitly represent what properties should be depicted when displaying an entity.

```
:tennisPlayerGKPDefaultLens rdf:type fresnel:Lens ;
fresnel:purpose fresnel:defaultLens ;
fresnel:classLensDomain dbpedia-owl:TennisPlayer ;
fresnel:group :tennisPlayerGroup ;
fresnel:showProperties (dbpedia-owl:abstract dbpedia-owl:birthDate
 dbpedia-owl:birthPlace dbpprop:height dbpprop:weight
 dbpprop:turnedpro dbpprop:siblings) ;
prov:wasDerivedFrom
  <http://www.google.com/insidesearch/features/search/knowledge.html> .
```

4 Conclusion and Future Work

We have shown that it is possible to reveal what are the "important" properties of entities in a large knowledge base by reverse engineering the choices made by Google when creating knowledge graph panels and by comparing with users preferences obtained from a user survey. Our motivation is to represent this choice explicitly, using the Fresnel vocabulary, so that any application could just read this configuration file for deciding which properties of an entity is worth to visualize. We are aware that this knowledge is highly dynamic, the Google Knowledge Graph panel differing from countries and varying along the time. We have provided the code that enables to perform new calculation at run time and we aim to study the temporal evolution of what are important properties on a longer period. This knowledge which has been captured will be made available shortly in a SPARQL endpoint. We are also investigating the use of Mechanical Turk to perform a larger survey for the complete set of DBpedia classes.

Acknowledgments. This work has been partially supported by the ANR Datalift (ANR-10-CORD-009) and UCN (ANR-11-LABX-0031-01) projects.

References

1. M. Bergman. Deconstructing the Google Knowledge Graph. `http://www.mkbergman.com/1009/deconstructing-the-google-knowledge-graph`.
2. E. Pietriga, C. Bizer, D. Karger, and R. Lee. Fresnel: A Browser-Independent Presentation Vocabulary for RDF. In *5th International Semantic Web Conference (ISWC'06)*, pages 158–171, 2006.
3. B. Shneiderman. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In *IEEE Symposium on Visual Languages*, pages 336–343, 1996.
4. T. Steiner and S. Mirea. SEKI@home or Crowdsourcing an Open Knowledge Graph. In *1st International Workshop on Knowledge Extraction & Consolidation from Social Media (KECSM'12)*, Boston, USA, 2012.

[10] `http://www.w3.org/TR/prov-o/`