What is Special about Bethlehem, Pennsylvania?
Identifying Unexpected Facts about DBpedia Entities

Benjamin Schäfer, Petar Ristoski, and Heiko Paulheim

University of Mannheim, Germany
Research Group Data and Web Science
{benni,petar,heiko}@dwslab.de

Abstract. Most Linked Data browsers list *all* facts about an entity in an equal manner. In this paper, we present a prototype for identifying *unexpected* facts about entities, i.e., those facts that deviate from the expectations. To that end, we use an attribute-wise method for anomaly detection, which is also capable of providing qualitative explanations for the anomalies found. By comparing an entity at hand to a reference set of similar entities, we can provide information on how the entity at hand differs from the typical patterns found for similar entities, and display those unexpected facts together with a short explanation.

Keywords: DBpedia, Data Exploration, Anomaly Detection

1 Introduction

Many linked data browsers display lists of facts about an entity at hand without a particular notion of order or importance [2]. While some approaches exist for ranking the existing information [1, 3], the top ranked facts for an entity are often the trivial ones (e.g., *Bethlehem, Pennsylvania* is a *City*).

A slightly different problem is the search for *unexpected* or *surprising* facts. Rather than ranking facts by their importance, ranking by *unexpectedness* requires a notion of the usual state of an entity. To that end, an entity needs to be compared to a *reference set* of similar entities, and the typical patterns underlying the entities in that set have to be identified. Then, unexpected facts can be identified as those facts of an entity which strongly deviate from the patterns.

In this demonstration, we introduce a prototype for finding unexpected facts in DBpedia.[1] Starting from selecting a DBpedia entity, the user can define the reference set and is then presented a number of unexpected facts. The first results with selected entities look promising.

2 Prototype

The basic workflow of the tool comprises four steps, as depicted in Fig. 1. In the first step, the user selects a DBpedia entity to analyze. This step is supported

[1] Available online at http://topfacts.informatik.uni-mannheim.de/

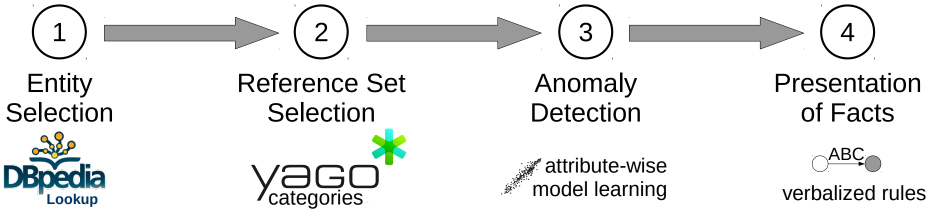


Fig. 1: Schematic depiction of the tool workflow.

by DBpedia Lookup and its autocomplete function.[2] For example, the entity `dbpedia:Bethlehem,_Pennsylvania` is selected.

Once the entity is selected, the user has to select a *reference set* of entities to compare to. To that end, all YAGO types, which form a much richer hierarchy than the DBpedia ontology types [9], are retrieved.[3] All types that have between 20 and 1,000 entities can be used as a reference set.[4] In our example, the user may, e.g., select the class `yago:CitiesInPennsylvania`.

In the next step, the reference set is retrieved. For each entity, an attribute vector is created, using attributes such as datatype properties and direct types. Based on those feature vectors, an individual anomaly score for each attribute is computed using the *ALSO* approach [5]. This approach learns a predictive model for each attribute from all other attributes. Then, it computes the anomaly score for each attribute value based on the deviation between the actual value and the value predicted by the model, and the predictive strength of the model. For building the models, we use the rule variant of M5' [6].

Finally, for all attributes that have a high anomaly score, the finding is output as the models' justification for expecting a different value, ordered by the respective anomaly score. An example output is shown in Fig. 2. Following the *details on demand* paradigm [8], the single statements which are involved in the justification are shown upon request.

Since the output would also show quite a few statements that are not unexpected facts, but mere errors in DBpedia, we filter out those rules referring to statements which are inconsistent with the DBpedia ontology.

For implementing the prototype, we use RapidMiner server[5] with the Linked Open Data extension [7].

3 Example Findings

In this section, we show some interesting example findings for different resources and reference sets.

[2] `http://lookup.dbpedia.org`

[3] DBpedia delivers YAGO types as well, so no separate linkage to YAGO is required.

[4] The numbers have been chosen for having a reference set that is big enough for discovering some meaningful patterns, and at the same time small enough to be processed in real time.

[5] http://www.rapidminer.com

Bethlehem, PA compared to Cities in Pennsylvania: Bethlehem is one of the oldest places in Pennsylvania, being founded in 1741. Furthermore, most cities in Pennsylvania are not a founding place of any organization, but Bethlehem is the founding place of *Lehigh University Press*.

Pennsylvania compared to States of the US: For Pennsylvania, we find that it has some uncommon characteristics for the US founding states[6]: it is unusually large (119,283 square kilometers, with only *New York* being larger), has an unusually large maximum elevation (*Mount Davis* with 979m), and an unusually low area covered with water (2.7%).

Black Swan compared to Ballet Films: It is unusual, e.g., that Black Swan is an Academy Award winning ballet film. Futhermore, ballet films are usually not thrillers.

Trent Reznor compared to American Heavy Metal Singers: Unlike other heavy metal singers, Reznor is also a piano player and has written various film scores.

Joanne K. Rowling compared to British Billionaires: Rowling is the only female among the British billionaires, and one of the rare supporters of the Labour party.

4 Conclusion and Outlook

In this paper, we have introduced a prototype which identifies unexpected facts about DBpedia entities. We compare an entity to a reference set of similar entities, and identify those facts which deviate from the patterns that are typical for the reference set.

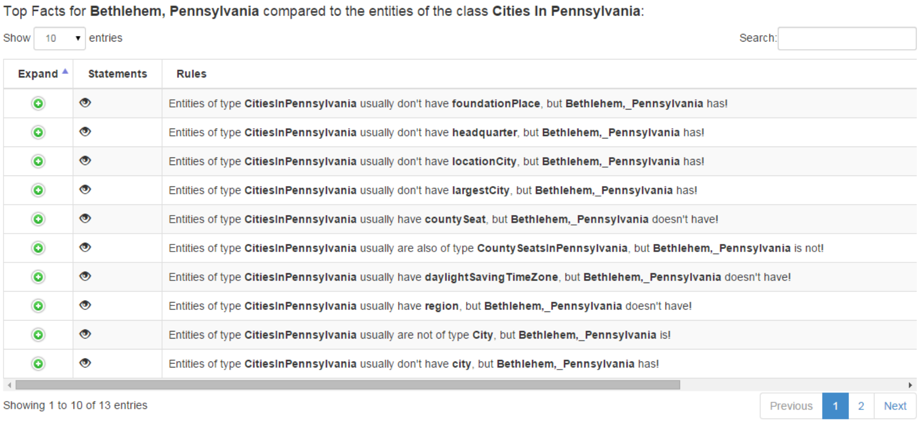
While first anecdotal findings are promising, a full user evaluation, also contrasting different presentation variants, still has to be conducted. Such an evaluation should, ideally, try to define and capture the human notion of *unexpectedness*, which, however, is not trivial.

In our prototype, we have so far used direct types, numeric datatype attributes, and relations as features. Other features, such as relations to individuals or qualified relations [4], might even lead to more findings, but come at the cost of a dimensionality explosion, and, hence, problems with realtime processing. Thus, some mechanism for on-the-fly feature selection would be required. Furthermore, the impact of the choice of different rule learning algorithms and heuristics would be interesting to explore.

For defining the reference set, we have only used YAGO categories so far. It would be interesting to also allow more sophisticated restrictions, e.g., compare a city to other cities in the same range of inhabitants.

In summary, the demo shows a novel way of interacting with Linked Data and identifying facts which are interesting to the user.

[6] Although this was not the contrast set we chose, many of the rules found refer to the founding states.



Top Facts for **Bethlehem, Pennsylvania** compared to the entities of the class **Cities In Pennsylvania**:

Expand	Statements	Rules
⊕	👁	Entities of type **CitiesInPennsylvania** usually don't have **foundationPlace**, but **Bethlehem,_Pennsylvania** has!
⊕	👁	Entities of type **CitiesInPennsylvania** usually don't have **headquarter**, but **Bethlehem,_Pennsylvania** has!
⊕	👁	Entities of type **CitiesInPennsylvania** usually don't have **locationCity**, but **Bethlehem,_Pennsylvania** has!
⊕	👁	Entities of type **CitiesInPennsylvania** usually don't have **largestCity**, but **Bethlehem,_Pennsylvania** has!
⊕	👁	Entities of type **CitiesInPennsylvania** usually have **countySeat**, but **Bethlehem,_Pennsylvania** doesn't have!
⊕	👁	Entities of type **CitiesInPennsylvania** usually are also of type **CountySeatsInPennsylvania**, but **Bethlehem,_Pennsylvania** is not!
⊕	👁	Entities of type **CitiesInPennsylvania** usually have **daylightSavingTimeZone**, but **Bethlehem,_Pennsylvania** doesn't have!
⊕	👁	Entities of type **CitiesInPennsylvania** usually have **region**, but **Bethlehem,_Pennsylvania** doesn't have!
⊕	👁	Entities of type **CitiesInPennsylvania** usually are not of type **City**, but **Bethlehem,_Pennsylvania** is!
⊕	👁	Entities of type **CitiesInPennsylvania** usually don't have **city**, but **Bethlehem,_Pennsylvania** has!

Showing 1 to 10 of 13 entries

Fig. 2: Example explanations provided by the tool.

Acknowledgements

The work presented in this paper has been partly funded by the German Research Foundation (DFG) under grant number PA 2373/1-1 (Mine@LOD).

References

1. Gong Cheng, Thanh Tran, and Yuzhong Qu. Relin: relatedness and informativeness-based centrality for entity summarization. In *Proceedings of the 10th International Semantic Web Conference (ISWC2011)*, pages 114–129, 2011.
2. Aba-Sah Dadzie and Matthew Rowe. Approaches to visualising linked data: A survey. *Semantic Web*, 2(2):89–124, 2011.
3. Li Ding, Rong Pan, Tim Finin, Anupam Joshi, Yun Peng, and Pranam Kolari. Finding and ranking knowledge on the semantic web. In *Proceedings of the 4th International Semantic Web Conference (ISWC2005)*, pages 156–170, 2005.
4. Heiko Paulheim and Johannes Fürnkranz. Unsupervised Generation of Data Mining Features from Linked Open Data. In *International Conference on Web Intelligence, Mining, and Semantics (WIMS'12)*, 2012.
5. Heiko Paulheim and Robert Meusel. A Decomposition of the Outlier Detection Problem into a Set of Supervised Learning Problems. *Machine Learning*, (2-3):509–531, 2015.
6. John Quinlan. Learning with continuous classes. In *5th Australian Joint Conference on Artificial Intelligence*, volume 92, pages 343–348. Singapore, 1992.
7. Petar Ristoski, Christian Bizer, and Heiko Paulheim. Mining the Web of Linked Data with RapidMiner. *Journal of Web Semantics*, 2015.
8. Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *IEEE Symposium on Visual Languages*, pages 336–343, 1996.
9. Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *16th international conference on World Wide Web*, pages 697–706, 2007.