Predicting Human Associations with Graph Patterns Learned from Linked Data

Jörn Hees[1,2], Rouven Bauer[1,2], Joachim Folz[1,2], Damian Borth[1,2], and Andreas Dengel[1,2]

[1] Computer Science Department, University of Kaiserslautern, Germany
[2] Knowledge Management Department, DFKI GmbH, Kaiserslautern, Germany
{joern.hees,rouven.bauer,joachim.folz,damian.borth,andreas.dengel}@dfki.de

Abstract The datasets provided by the Linked Data community currently form the world's largest, freely available, decentralised and interlinked knowledge bases. However, to be able to benefit from this knowledge in a specific use-case, one typically needs to understand the modelling of the knowledge and formulate appropriate SPARQL queries.
In order to ease this process, we developed an evolutionary algorithm that learns such SPARQL queries (graph patterns) for pairwise relations between source and target entities. Given a training list of source-target-pairs, our algorithm learns a predictive model, which given a new source entity predicts target entities analogously to the training examples.
In this demo paper we present a high level overview over our graph pattern learner and show its application to simulate human associations (e.g., "fish - water"). In the demo users can choose a semantic entity (e.g., `dbr:Fish`) as stimulus and let the learned model predict human-like responses (e.g., `dbr:Water`).

1 Introduction

In recent years, many large, machine accessible and interlinked RDF datasets have emerged from the Semantic Web [1] and its Linked Data [2] movement. The datasets are prominently depicted as the LOD Cloud[3] and form the currently largest openly available representation of machine accessible knowledge. Due to its encyclopaedic nature DBpedia[4] [3] has become one of the most interlinked and central datasets of the LOD Cloud.

Despite the availability of all this knowledge, actually using it typically requires non-trivial up-front work: SPARQL queries need to be formulated to extract relevant knowledge for the given use-case.

Hence, we developed a graph pattern learning algorithm [4] that can help to learn such SPARQL queries. While several other systems exist that learn SPARQL queries (e.g., AutoSPARQL [7], kretr [8]), they typically focus on learning a single query for a simple list of entities. Our algorithm differs from these

[3] `http://lod-cloud.net/`
[4] `http://dbpedia.org`

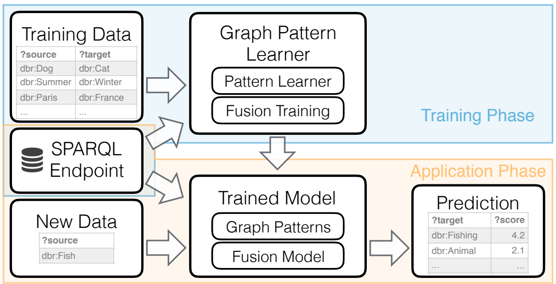


Figure 1: Graph Pattern Learner System Overview

in two main aspects: (i) it learns an ensemble model that can (and will) consist of many queries and (ii) it doesn't try to learn queries that reproduce a given list of entities, but it learns queries that represent a relation $\mathcal{R}$ between entity source-target-pairs $(s, t) \in \mathcal{R}$. By learning queries for a relation between pairs of entities, the graph pattern learner can generate a predictive model, that given a new source entity s' can predict targets $\{t' | (s', t') \in \mathcal{R}\}$.

In this demo paper we show one such predictive trained model that has been generated by our graph pattern learner as detailed in the following sections.

2 Graph Pattern Learner System Overview

Figure 1 shows a high level system overview of the graph pattern learner. The system is divided into training and application phase and resembles a classic machine learning outline. The *training phase* mainly consists of the *graph pattern learner* that is given its training data in form of source-target-pairs and a SPARQL endpoint from which to learn patterns.

For this demo, the *training data* consisted of 655 pairs of "semantic associations"[5] [5], corresponding to ≈ 22500 human associations from the Edinburgh Associative Thesaurus (EAT) [6]. The *SPARQL endpoint* for this demo is a local LOD mirror loaded with $\approx 8G$ triples (amongst others: DBpedia 2015-10, Wikidata, Freebase, BabelNet, DBLP, YAGO Labels)[6].

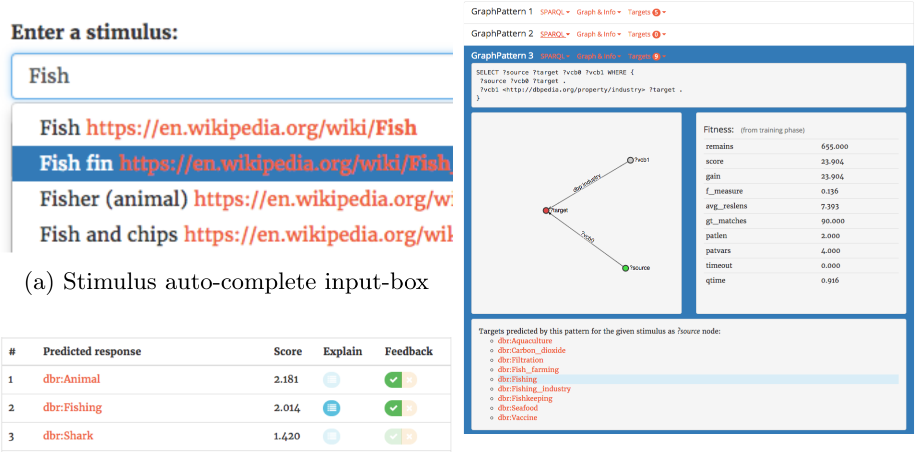
In all brevity[7], the *graph pattern learner* is an evolutionary algorithm which finds SPARQL queries that are good predictors: Each pattern consists of a SPARQL Basic Graph Pattern (BGP) which contains at least a `?source` and a `?target` variable. A pattern is good if it fulfils many of the training source-target-pairs (`ASK` queries where `?source` and `?target` are bound correspondingly) and it doesn't generate much noise (`SELECT ?target` queries where `?source` is bound only return few irrelevant values for `?target`).

Each of the patterns learned by the pattern learner is a predictor: We can execute a SPARQL `SELECT ?target` query in which we bind the `?source` variable

[5] Datasets available at http://w3id.org/associations/#datasets.

[6] For further details, see set-up instructions at: https://joernhees.de/blog/2015/11/23/setting-up-a-linked-data-mirror

[7] See [4] for details on the training and evaluation.



Enter a stimulus:

Fish

- Fish https://en.wikipedia.org/wiki/Fish
- Fish fin https://en.wikipedia.org/wiki/Fish
- Fisher (animal) https://en.wikipedia.org/w
- Fish and chips https://en.wikipedia.org/wil

(a) Stimulus auto-complete input-box

#	Predicted response	Score	Explain	Feedback
1	dbr:Animal	2.181	●	✓ ✗
2	dbr:Fishing	2.014	●	✓ ✗
3	dbr:Shark	1.420	●	✓ ✗

(b) Fused Prediction Results

GraphPattern 1 SPARQL ▾ Graph & Info ▾ Targets 5 ▾

GraphPattern 2 SPARQL ▾ Graph & Info ▾ Targets 0 ▾

GraphPattern 3 SPARQL ▾ Graph & Info ▾ Targets 9 ▾

```
SELECT ?source ?target ?vcb0 ?vcb1 WHERE {
 ?source ?vcb0 ?target .
 ?vcb1 <http://dbpedia.org/property/industry> ?target .
}
```

Fitness: (from training phase)

remains	655.000
score	23.904
gain	23.904
f_measure	0.136
avg_reslens	7.393
gt_matches	90.000
patlen	2.000
patvars	4.000
timeout	0.000
qtime	0.916

Targets predicted by this pattern for the given stimulus as *?source* node:

- dbr:Aquaculture
- dbr:Carbon_dioxide
- dbr:Filtration
- dbr:Fish_farming
- dbr:Fishing
- dbr:Fishing_industry
- dbr:Fishkeeping
- dbr:Seafood
- dbr:Vaccine

(c) Filtered graph patterns highlighting those that generated the target dbr:Fishing

Figure 2: Main Demo Components

to an arbitrary source s' (e.g. with a `VALUES (?source) {(`s'`)}` block). This results in one list of target candidates per pattern that need to be ranked to yield the set of predicted targets. The *fusion training* component uses several strategies to generate (late) fusion models for this purpose.

In order to fuse such resulting target-lists for a provided new source node, the graph pattern learner includes a *fusion training* component that generates late fusion machine learning models. The fusion models vary in complexity from basic to full-fledged machine learning models themselves. For example, we provide basic target-occurrence ranking over all queries (called "target occurrences") potentially normalised so that each pattern only has a total vote of 1 (called "precisions"). As full machine learning models, we provide amongst others KNN, SVM, Logistic Regression and RankSVM models estimating relevance based on target candidate vectors wrt. the generating queries. In the demo the user can switch between these fusion models with a simple drop-down as shown in Section 3.

After training, the set of graph patterns and fusion model form a predictive *trained model* that is used in the *application phase*. Given a new source node (e.g. `dbr:Fish`) the trained model uses all learned graph patterns to issue `SELECT ?target` queries in which the `?source` variable is bound to the new source node against the SPARQL endpoint, and fuses the individual target result lists into an overall ranked list of target predictions.

3 Demo

The main screen of our online demo[8] starts with the stimulus auto-complete input-box on top (Figure 2a) asking the user to enter a stimulus. The auto-

[8] `https://w3id.org/associations/gp_learner/demo/predict.html`

complete is realised via the Wikipedia OpenSearch API[9], allowing a fuzzy search for matching Wikipedia Articles, including the resolving of redirects.

After selecting one of the Wikipedia articles from the auto-complete box, the URI is transformed to the corresponding DBpedia resource and the prediction started. The fused prediction results are then presented in the "Fused Prediction" tab (Figure 2b), in which the user can provide feedback about the generated targets (logged and used for future improvements). The user can also click the explain button to gain insight on why a target was predicted. All graph patterns that played a role in predicting this target will be highlighted and expanded in the "Graph Patterns" tab (Figure 2c).

4 Conclusion

In this demo paper we presented a high level overview over the graph pattern learner and show its application to simulate human associations. The algorithm, used datasets and interactive visualisation of the results are available online.

`https://w3id.org/associations/gp_learner/demo/predict.html`.

This work was supported by the University of Kaiserslautern CS PhD scholarship program and the BMBF project MOM (Grant 01IW15002).

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. Scientific American 284(5), 34–43 (may 2001)
2. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data - The Story So Far. International Journal on Semantic Web and Information Systems 5(3), 1–22 (jan 2009)
3. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - A crystallization point for the Web of Data. Web Semantics: Science, Services and Agents on the World Wide Web 7(3), 154–165 (sep 2009)
4. Hees, J., Bauer, R., Folz, J., Borth, D., Dengel, A.: An Evolutionary Algorithm to Learn SPARQL Queries for Source-Target-Pairs. In: Knowledge Engineering and Knowledge Management - EKAW 2016. vol. 10024, pp. 337–352. Springer LNCS, Bologna, Italy (nov 2016)
5. Hees, J., Bauer, R., Folz, J., Borth, D., Dengel, A.: Edinburgh associative thesaurus as RDF and DBpedia mapping. In: The Semantic Web - ESWC 2016 SE. vol. 9989 LNCS, pp. 17–20. Springer LNCS, Heraklion, Crete, Greece (may 2016)
6. Kiss, G., Armstrong, C., Milroy, R., Piper, J.: An associative thesaurus of English and its computer analysis. In: The Computer and Literary Studies, pp. 153–165. Edinburgh University Press, Edinburgh, UK (1973)
7. Lehmann, J., Bühmann, L.: AutoSPARQL: Let users query your knowledge base. In: The Semantic Web: Research and Applications - ESWC 2011. LNCS, vol. 6643, pp. 63–79. Springer, Heraklion, Crete, Greece (2011)
8. Potoniec, J.: An On-Line Learning to Query System. In: Proc. of the ISWC 2016 Posters & Demonstrations Track. vol. 1690. CEUR-WS.org, Kobe, Japan (2016)

[9] `https://www.mediawiki.org/wiki/API:Opensearch`