A Prototype for Automating Ontology Learning and Ontology Evolution

Gerhard Wohlgenannt, Stefan Belk and Matthias Schett
Vienna University of Economics and Business, Augasse 2-6, 1090 Wien, Austria

Keywords: Ontology Learning, Ontology Evolution, Crowdsourcing.

Abstract: Ontology learning supports ontology engineers in the complex task of creating an ontology. Updating ontologies at regular intervals greatly increases the need for expensive expert contribution. This naturally leads to endeavors to automate the process wherever applicable. This paper presents a model for automated ontology learning and a prototype which demonstrates the feasibility of the proposed approach in learning lightweight domain ontologies. The system learns ontologies from heterogeneous sources periodically and delegates all evaluation processes, eg. the verification of new concept candidates, to a crowdsourcing framework which currently relies on Games with a Purpose. Furthermore, we sketch ontology evolution experiments to trace trends and patterns facilitated by the system.

1 INTRODUCTION

Ontologies are a cornerstone technology for the Semantic Web, but the creation of ontologies is a cumbersome and very complex problem. Semi-automatic ontology learning helps to reduce effort by providing the ontology engineer with a starting point.

Ontology evolution is concerned with the adaptation of the ontology to changes in the domain (data-driven change), changed user requirements (user-driven change) or to correct flaws in the original design. Ontology evolution requires frequent updates or rebuilding of the ontology, esp. if investigating emerging trends and patterns in highly dynamic domains. In such a context, a greatly automated ontology learning process is very beneficial.

The work presented in this position paper builds upon and extends an ontology learning framework first published in 2005 (Liu et al., 2005). Since then the system has been improved to better support heterogeneous input sources (Wohlgenannt et al., 2012) and to detect non-taxonomic relations (Weichselbraun et al., 2010).

We introduce a prototype that aims to keep manual input in ontology learning and evolution to a minimum by automating the workflow in the ontology learning cycle. It delegates demand for human input to sources that are cheaper and much more scalable then conventional evaluation by domain experts. So, the goal is to minimize manual (domain expert and engineer) effort in repeated ontology learning cycles. This effort can be measured against other ontology learning systems. The presented architecture is built for a specific framework, but the ideas are supposed to have a general purpose. Finally, we draft experiments for trend and pattern detection.

2 RELATED WORK

Early work in ontology learning (Mädche and Staab, 2001) not only suggests methodologies for ontology learning, but also defines the tasks involved, broadly speaking the learning of concepts, taxonomic relations, non-taxonomic relations and axioms. The presented work focuses on lightweight ontologies, which include concepts and taxonomic relations. For the acquisition of new concepts related to existing concepts many authors exploit Harris' distributional hypothesis (Harris, 1968), which states that two words are similar to the extend that they share similar context.

Large projects like NeOn[1] developed complex ontology engineering environments. The NeOn toolkit includes the Text2Onto (Cimiano et al., 2005) ontology learning framework, which is Java-based, and geared towards the learning of rather expressive ontologies from domain text. Our work stems from a smaller project dedicated to learning lightweight ontologies from heterogeneous input sources with a focus on automation and evolution experiments.

[1] www.neon-project.org

Wohlgenannt G., Belk S. and Schett M..
A Prototype for Automating Ontology Learning and Ontology Evolution.
DOI: 10.5220/0004630504070412
In *Proceedings of the International Conference on Knowledge Engineering and Ontology Development* (KEOD-2013), pages 407-412
ISBN: 978-989-8565-81-5

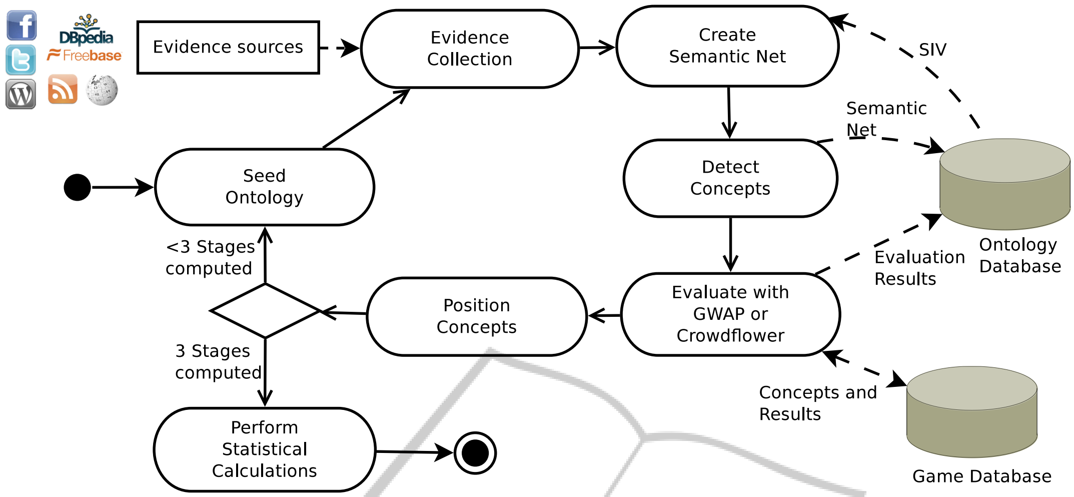


Figure 1: The Ontology Learning Process.

The evaluation of newly acquired concept candidates with Games with a Purpose (GWAPs) or human labor markets such as CrowdFlower is a central factor to make our system scalable. Noy et al. (Natasha F. Noy and Musen, 2013) demonstrate the suitability of Crowdsourcing with Amazon Mechanical Turk for evaluating hierarchical relations in ontologies. GWAPs have already been used for example for mapping Wikipedia articles to specific classes in the Proton ontology in the OntoPronto game (Siorpaes and Hepp, 2008) or for relation detection between concepts (Scharl et al., 2012). Existing tools typically do not offer a tight integration of evaluation results into the learning algorithms, however.

Ontology evolution can be defined as the “timely adaptation of an ontology to the arising changes and the consistent management of these changes” (Haase and Stojanovic, 2005). It helps to keep ontologies up-to-date and useful. The presented prototype integrates heterogeneous input sources in the evolution process, which to our knowledge is a novel approach except for initial efforts in the RELExO framework (Maynard and Aswani, 2010). In contrast to the Probabilistic Ontology Model (POM) in Text2Onto (Cimiano et al., 2009), which aims at change management aspects of ontology evolution, our automated approach targets the detection of trends and patterns in the data structures underlying and reflecting the ontology.

3 THE ONTOLOGY LEARNING FRAMEWORK

This section gives an overview of the process and prototype that performs ontology learning and captures ontology evolution with minimal manual input and effort. For more information about the underlying architecture and algorithms see (Wohlgenannt et al., 2012).

The system is written in Python, some minor components are developed in Java for performance reasons. It can roughly be divided into three parts:

1. A Web service & Web interface written in Python which orchestrates the processes and serves as a human interface for administrative tasks and as a monitoring tool.
2. The *ontology extension* component. It computes and positions new concepts in a domain ontology.
3. A keyword computation service written in Java, which is the most prominent source for evidence collection (from text).

This paper focuses on the Web service & Web interface, as those components are crucial for automating the process. The whole system is designed to reduce the amount of time experts have to invest in order to create new ontologies to a minimum. Expert contribution is only needed to install the system and initially configure the ontology learning cycle.

Figure 1 outlines the general workflow of a single extension step which extends a *seed ontology* into an *extended ontology*. At the end of the cycle the extended ontology serves as a new seed ontology for the next iteration. In our system the ontology extension iterations are called *stages*, by default the whole process consists of three stages (defined in the configuration of the Web service).

The initial seed ontology is typically a small set of concepts and relations (specified in an OWL file) which is characteristic of the respective domain. In order to extend the ontology we collect evidence for related concepts from a number of evidence sources.

This evidence includes keywords determined with co-occurrence statistics from domain corpora using the keyword computation service, *related terms* suggested by social sources such as Twitter, Flickr or Technorati to capture very recent terminology and trends, hyponyms and hypernyms proposed by WordNet (Fellbaum, 1998), etc. As we periodically generate new ontologies from scratch to trace the evolution of the domain, all evidence stems from the time period in question (by default the last month). For more details on evidence collection see (Liu et al., 2005).

The accumulated evidence data is collected in a semantic net, which is then transformed into a spreading activation network. The weights in the network are influenced by the so called *source impact value (SIV)* of the source which suggested the evidence. The source impact values reflect the estimated quality of the evidence source, and are currently our primary target when optimizing the ontology learning process. Through activating the spreading activation network, the system computes the 25 most important candidate concepts for the given seed ontology. Currently, a Facebook-based GWAP is used to eliminate unrelated concepts. The game has similar mechanics as the one described in (Scharl et al., 2012).

The players of the game evaluate the concepts by analyzing their relevance to the ontology's domain, the result is then sent back to the ontology Web service. A more powerful evaluation framework which performs evaluation tasks either with (refined) GWAPs or delegates the job to human labor markets such as CrowdFlower[2] is under development. The candidate concepts evaluated as relevant will then be positioned in the ontology, for positioning algorithm details see (Liu et al., 2005). Finally, the system creates a graphical representation of the ontology and saves it into the file system (in OWL format).

The result (extended ontology) from stage one is the starting point for the next stage, which repeats the whole computation and evaluation process. The framework is designed to compute an arbitrary number of stages (extension iterations), but for our purposes three stages are appropriate.

As briefly mentioned, the ontology learning system automatically optimizes its own performance by adapting source impact values per evidence source. After completion of the three ontology extension stages the Web service calculates new source impact values. They are based on the evaluation of concepts suggested by the source in the current run and a weighted arithmetic mean of previous ratings over the past 365 days.

As shown in Figure 1, all important data collected or computed by the system is stored in a database for various reasons: persistence, easy access, and support for evolution experiments (see Section 5). The database contains metadata about each ontology (stage), the evidence collected for the ontology, all concepts, all evaluation results, source impact values, etc.

Automation. A lot of effort has been made to automate the system as far as possible. A Web service (see next section) controls the workflow, evaluation (GWAPs/CrowdFlower) is the only task in the learning cycle where human input cannot be avoided. Furthermore, to speed up computations we use caching strategies in various processes:

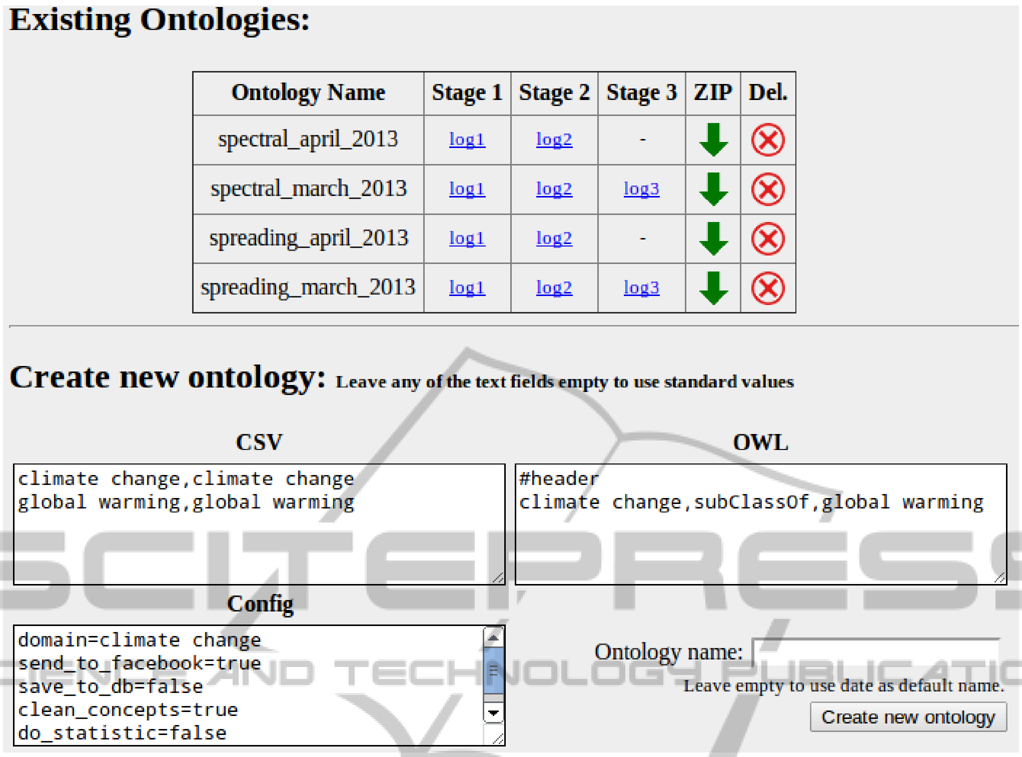
- The evidence collection phase covers processes that are computationally complex (such as the computation of keywords via co-occurrence statistics) or call third party APIs. With the help of the eWRT toolkit [3] the framework applies fine-grained caching strategies to only call the respective evidence collection service for a seed when the necessary data cannot be derived from previous computations already existing in the system.
- The evaluation service (Facebook GWAP) stores the results of past concept validation processes, and lets users only evaluate entirely new concepts. To allow for changes in the domain, concepts have to be re-evaluated after a period of six months.
- To improve the run-time performance of the spreading activation algorithms we experiment with an approximation technique called spectral association (Havasi et al., 2012).
- When manually calling the ontology extension process, eg. for experimenting with parameter settings, new domains or revised code, various steps in the process can be deactivated easily and thereby forced to re-use existing data.

4 THE WEB SERVICE & ADMINISTRATION INTERFACE

This section includes technical information about the Web service and the corresponding administrative interface. The main function of the Web service is to guide the workflow, ie. calling the involved components with the right parameters and handling the communication between internal and external services.

[2] `crowdflower.com`

[3] www.weblyzard.com/ewrt



Existing Ontologies:

Ontology Name	Stage 1	Stage 2	Stage 3	ZIP	Del.
spectral_april_2013	log1	log2	-	↓	⊗
spectral_march_2013	log1	log2	log3	↓	⊗
spreading_april_2013	log1	log2	-	↓	⊗
spreading_march_2013	log1	log2	log3	↓	⊗

Create new ontology: Leave any of the text fields empty to use standard values

CSV

```
climate change,climate change
global warming,global warming
```

OWL

```
#header
climate change,subClassOf,global warming
```

Config

```
domain=climate change
send_to_facebook=true
save_to_db=false
clean_concepts=true
do_statistic=false
```

Ontology name:

Leave empty to use date as default name.

Figure 2: The Administration Interface (clipped).

In our environment, a cron job initiates the generation of new ontologies for all predefined configurations at the end of each month via the REST API of the Web service. A monthly interval is appropriate for our purposes, but any other interval is conceivable. The ontology learning system uses the evidence collected for the respective period.

The communication to the GWAP API to create evaluation tasks and to receive the results for those tasks is a critical component. The system uses a JSON format to communicate with the crowdsourcing framework. The format contains the ID of the ontology as key on the root level, and for any ontology we use its domain (eg. “climate change”) as key, and the candidate concepts (the terms which represent them) as values. The JSON objects returned from evaluation additionally contain the results, encoded as the number of votes “relevant”, “not relevant”, and “undecided” for a candidate, as in the example below:

```
{"Ontology CC 2013-04 spectral":
    { "climate change": [
        ["CO2",4,0,0],
        ["water",0,2,2],
        [....], ]
}     }
```

To raise validity of results, the system uses inter-player agreement on every evaluation task. The number of conforming votes necessary for evaluating a concept candidate is configurable.

Moreover, the Web service handles the following jobs which help to minimize manual intervention:

- Check for the existence and correct installation of the required Linux and Python components and the availability of the keyword computation service; notify the user if anything is missing.
- Create the folder structure for new ontologies in the file system
- Handle and save log, config and JSON files for each ontology
- Create graphical representations of the created ontologies for each stage
- Compute new source impact values based on the results of the evaluation.

Figure 2 shows parts of the administration interface (clipped to contain only a very few ontologies to save space). The interface is divided into four parts. At the top (not shown in the screenshot) it displays information about the current status of the system and provides a link to the Web service’s global log file. Below there is a list of ontologies existing in the system. For any ontology the user can view the logs for the three stages, download all data or delete it. The logs also contain the resulting ontology graph.

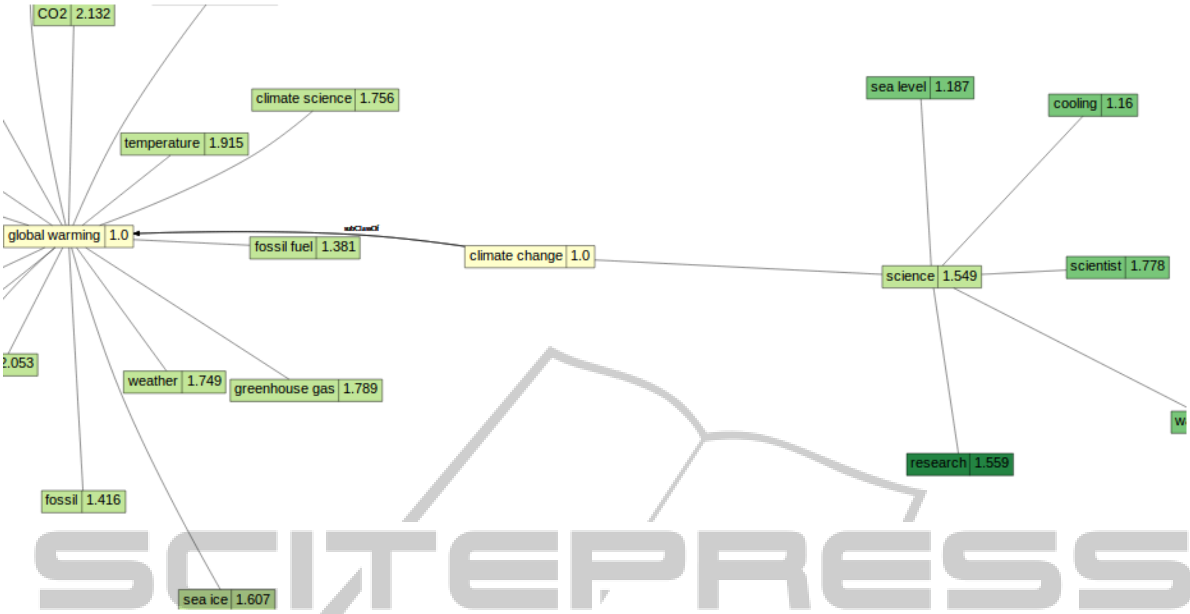


Figure 3: An extended ontology (clipped).

Additionally to the fully automated generation of ontologies, the user can also create an ontology manually, this can be easily done via the Web interface's "Create new ontology" form found below the list of existing ontologies. This allows the user to define and experiment with various configurations which affect the ontology learning process.

The user has a wide variety of parameter settings to choose from, these can be grouped into the following classes:

- *Algorithms and evidence sources:* Set the algorithms to be used to create the new ontology (eg. *spreading activation* or *spectral association*), or set the period of time to be used.
- *Testing:* Just compute the ontologies, but do not save the results into the database (`save_to_db`), save the results into another database to better separate results for production and testing environments (`db_name`), do (not) update the source impact values after the completed run (`do_statistics`).
- *Evaluation:* Disable evaluating and filtering terms via the evaluation service but just keeping all concept candidates automatically (`send_to_facebook`), or not filtering the concepts even if GWAP evaluation has been done (`clean_concepts`).

The text areas *CSV* and *OWL* are for entering the seed ontology for a new ontology learning process. The OWL text area receives the seed concepts and their relations as triples of subject, predicate and object. These concepts are consistent with the CSV area where a regular expression can be set for each concept; the text based evidence sources (eg. keyword detection) use the regular expression as a lexical representation of the concept.

Finally the user can give the new ontology a name, if omitted, a name including creation date and time will be generated.

The last part of the interface (not shown in the screenshot) displays information about ontology computations currently running, including their names, starting time, parameter settings, etc., and gives the option to terminate running computations.

Figure 3 depicts parts on an extended ontology, the yellow boxes represent the original seed concepts, whereas shades of green denote concepts added in stage one (light-green), two (green) and three (dark-green).

5 ONTOLOGY EVOLUTION EXPERIMENTS PLANNED

As already discussed, a relational DBMS (PostgreSQL [4]) manages all of the information that is relevant to trace the evolution of the ontology and therefore the domain – on the level of concepts and evaluation results, but also on the fine-grained level of evidences which finally lead to concept candidates.

Based on the database, we plan to detect various

[4] www.postgresql.com

types of trends, for example rising, falling and cyclic patterns. SQL-queries and data visualization will help achieve the following:

- Trace the observed quality of evidence sources based on the history of source impact values.
- Monitor the quality of the ontology learning system itself via the ratio of relevant to irrelevant concept candidates.
- Investigate which sources suggest which concepts, and shifts between sources.
- Examine aggregated (eg. all text or all social evidence sources) patterns, or comparisons across domains.

6 CONCLUSIONS

This position paper presents the enhancements to an existing ontology learning system – adding novel features to automate the ontology learning cycle as far as possible. These features allow for a wide range of ontology evolution experiments which reflect and detect data-driven change in the domain.

The main contributions of the paper are (i) providing a model which supplies a high level of automation for learning and evolving lightweight ontologies, (ii) describing a prototype which implements this model as a Web service, including the administration interface and parameters, (iii) presenting trend and pattern detection experiments facilitated by the automated architecture and the database that collects fine-grained data about ontological elements over time.

Future work includes the completion of a more powerful evaluation framework which performs evaluation tasks either with (refined) GWAPs or delegates them to CrowdFlower. The new evaluation framework is under development. Furthermore, after collecting longitudinal data, we will conduct and extend the ontology evolution experiments described in Section 5.

ACKNOWLEDGEMENTS

The presented work was developed within DIVINE (www.weblyzard.com/divine), a project funded by the Austrian Ministry of Transport, Innovation & Technology (BMVIT) and the Austrian Research Promotion Agency (FFG) within FIT-IT (www.ffg.at/fit-it). The work has also been supported by uComp (www.ucomp.eu), a project in EU's ERA-NET CHIST-ERA programme.

REFERENCES

Cimiano, P., Maedche, A., Staab, S., and Voelker, J. (2009). Ontology learning. In Staab, S. and Rudi Studer, D., editors, *Handbook on Ontologies*, International Handbooks on Information Systems, pages 245–267. Springer Berlin Heidelberg.

Cimiano, P., Pivk, A., Schmidt-Thieme, L., and Staab, S. (2005). *Ontology Learning from Text*, chapter Learning Taxonomic Relations from Heterogeneous Sources of Evidence, pages 59–76. IOS Press, Amsterdam.

Fellbaum, C. (1998). Wordnet an electronic lexical database. *Computational Linguistics*, 25(2):292–296.

Haase, P. and Stojanovic, L. (2005). Consistent evolution of owl ontologies. In *Proceedings of the Second European Semantic Web Conference, Heraklion, Greece*, pages 182–197.

Harris, Z. S. (1968). *Mathematical Structures of Language*. Wiley, New York, NY, USA.

Havasi, C., Borovoy, R., Kizelshteyn, B., Ypodimatopoulos, P., Ferguson, J., Holtzman, H., Lippman, A., Schultz, D., Blackshaw, M., and Elliott, G. T. (2012). The glass infrastructure: Using common sense to create a dynamic, place-based social information system. *AI Magazine*, 33(2):91–102.

Liu, W., Weichselbraun, A., Scharl, A., and Chang, E. (2005). Semi-automatic ontology extension using spreading activation. *Journal of Universal Knowledge Management*, 0(1):50–58.

Mädche, A. and Staab, S. (2001). Ontology learning for the semantic web. *IEEE Intelligent Systems*, 16(2):72–79.

Maynard, D. and Aswani, N. (2010). Bottom-up Evolution of Networked Ontologies from Metadata (NeOn Deliverable D1.5.4).

Natasha F. Noy, Jonathan Mortensen, P. A. and Musen, M. (2013). Mechanical turk as an ontology engineer? In *Proceedings of the ACM Web Science 2013 (WebSci'13)*, Paris, Forthcoming.

Scharl, A., Sabou, M., and Föls, M. (2012). Climate quiz: a web application for eliciting and validating knowledge from social networks. In Bressan, G., Silveira, R. M., Munson, E. V., Santanchà, A., and da Graça Campos Pimentel, M., editors, *WebMedia*, pages 189–192. ACM.

Siorpaes, K. and Hepp, M. (2008). OntoGame: Weaving the semantic web by online games. In Bechhofer, S., Hauswirth, M., Hoffmann, J., and Koubarakis, M., editors, *5th European Semantic Web Conference (ESWC)*, volume 5021, pages 751–766. Springer.

Weichselbraun, A., Wohlgenannt, G., and Scharl, A. (2010). Refining non-taxonomic relation labels with external structured data to support ontology learning. *Data & Knowledge Engineering*, 69(8):763–778.

Wohlgenannt, G., Weichselbraun, A., Scharl, A., and Sabou, M. (2012). Dynamic integration of multiple evidence sources for ontology learning. *Journal of Information and Data Management (JIDM)*, 3(3):243–254.