What can be done with the Semantic Web? An Overview of Watson-based Applications*

Mathieu d'Aquin, Marta Sabou, Enrico Motta, Sofia Angeletou, Laurian Gridinoc, Vanessa Lopez, and Fouad Zablith

Knowledge Media Institute (KMi)
The Open University, Milton Keynes, United Kingdom
{m.daquin, r.m.sabou, e.motta, s.angeletou, l.gridinoc, v.lopez, f.zablith}@open.ac.uk

Abstract. Thanks to the huge efforts deployed in the community for creating, building and generating semantic information for the Semantic Web, large amounts of machine processable knowledge are now openly available. Watson is an infrastructure component for the Semantic Web, a gateway that provides the necessary functions to support applications in using the Semantic Web. In this paper, we describe a number of applications relying on Watson, with the purpose of demonstrating what can be achieved with the Semantic Web nowadays and what sort of new, smart and useful features can be derived from the exploitation of this large, distributed and heterogeneous base of semantic information.

1 Introduction

It is now commonly admitted that, while it still needs to grow and evolve, the Semantic Web has already become a reality. Millions of RDF documents are now published online, describing hundreds of millions of entities through billions of statements. The Semantic Web is now the biggest, most distributed and most heterogeneous knowledge base that ever existed, and is very quickly evolving, as thousands of users constantly create new knowledge and update the existing one.

While it now appears clearly that the effort deployed by the community in creating semantic information and making it available on the Web has been successful, it seems to be a good time to consider the applications of this huge information infrastructure the Semantic Web has become: What can be done with it?

Watson [1] is a Semantic Web Gateway: it collects and indexes semantic information on the Web, to provide a variety of access mechanisms for users and applications. As such, it provides support for building a new kind of applications taking benefit from the Semantic Web as it makes possible within these applications to dynamically find, explore and exploit online semantic content [2].

* This work was funded by the Open Knowledge and NeOn projects sponsored by the European Commission as part of the Information Society Technologies (IST) programme.

A number of applications relying on Watson have already been developed and provide demonstrators of the possibilities offered by this approach of exploiting the Semantic Web. In this paper, we describe these applications (at least, the ones we know of, as Watson is an open service that anybody can use), with the aim of providing an overview of the variety of tasks that can be achieved nowadays with the Semantic Web.

2 Watson: the Semantic Web Gateway

The need of next generation Semantic Web applications for a new kind of infrastructure motivated the development of Watson: a gateway that realizes a single access point to the semantic information published online and provides efficient services to support application developers in exploiting this large amount of distributed and heterogeneous data.

2.1 Overview

Watson collects online semantic documents through a variety of crawlers, exploring different sources, such as *PingTheSemanticWeb.com*. The particularity of these crawlers, compared with usual web crawlers, is that they consider semantic relations across documents in addition to classical hyperlinks. Also, when collecting online semantic content they check for duplicates, copies, or prior versions of the discovered documents.

Once collected, these documents are analyzed and indexed according to a variety of information concerning the content of the document, its complexity, quality and relations to other resources. This analysis step is core to Watson as it ensures that the key information is extracted, which can help applications in selecting, assessing, exploiting and combining these resources.

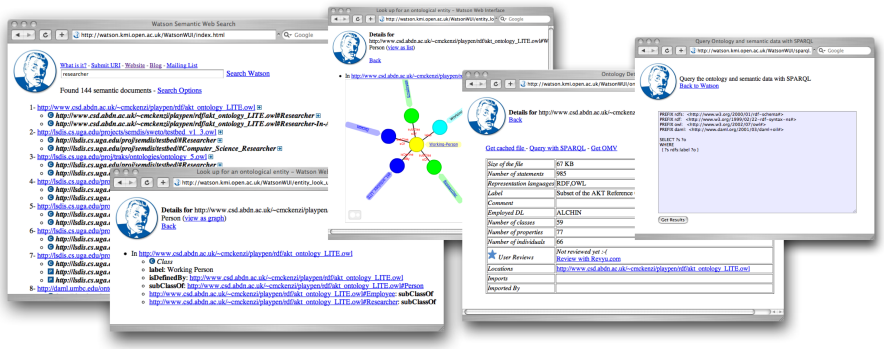

Fig. 1. Screenshots of the Watson Web interface (`http://watson.kmi.open.ac.uk`)

Finally, the goal of Watson is to provide efficient and adequate access to the collected information for applications and, to some extent, human users. A web interface allows users to search semantic content by keywords, to inspect them, to explore semantic documents, and to query them using SPARQL (see Figure 1).

However, the important part of Watson is the services and API it provides to support the development of next generation Semantic Web applications. Indeed, Watson deploys a number of Web services and a corresponding API allowing applications to:

- find Semantic Web documents through sophisticated keyword based search, allowing applications to specify queries according to a number of parameters (type of entities, level of matching of the keywords, etc.);
- retrieve metadata about these documents, e.g., size, language, label, logical complexity, etc;
- find specific entities (classes, properties, individuals) within a document;
- inspect the content of a document, i.e., the semantic description of the entities it contains;
- apply SPARQL queries to Semantic Web documents.

The combination of mechanisms for searching semantic documents (keyword search), retrieving metadata about these documents and querying their content (e.g., through SPARQL) provides all the necessary elements for applications to select and exploit online semantic resources. Moreover, the Watson web services and API are in constant evolution to support the requirements of novel applications.

2.2 Differences with Other Semantic Web Search Engines

There are a number of similar systems to Watson, falling into the category of *Semantic Web search engines*. However, Watson differs from these systems in a number of ways, the main one being that Watson is the only one to provide the necessary level of services for applications to dynamically exploit Semantic Web data. For example, one of the most popular Semantic Web Search engine is Sindice[1]. However, while Sindice indexes a very large amount of semantic data, it only provides a simple *look-up* service allowing applications/users to "locate" semantic documents. Therefore, it is still necessary to download and process these documents locally to exploit them, which in many cases, is not feasible. The Swoogle system[2] is closer to Watson. However, it does not provide some of the advanced search and exploration functions that are present in the Watson APIs (including the SPARQL querying facility). The Falcon-S[3] Semantic Web search engine has been focusing more on the user interface aspects, but now provides an initial API including a sub-set of the functions provided by Watson.

Another important aspect to consider is how open Semantic Web Search engines are. Indeed, Watson is the only Semantic Web search engine to provide unlimited access to its functionalities. Sindice, Swoogle and Falcon-S are, on the contrary, restricting the possibility they offer by limiting the number of queries executable in a day or the number results for a given query.

[1] `http://sindice.com/`
[2] `http://swoogle.umbc.edu/`
[3] `http://iws.seu.edu.cn/services/falcons/api/index.jsp`

3 Services on Top of Watson

In this section, we look at a first category of applications of Watson: System that make use of Watson to provide additional features for users and developers.

3.1 Scarlet: Relation Discovery

Scarlet[4] follows the paradigm of automatically selecting and exploring online ontologies to discover relations between two given concepts. For example, when relating two concepts labeled *Researcher* and *AcademicStaff*, Scarlet 1) identifies (at run-time) online ontologies that can provide information about how these two concepts inter-relate and then 2) combines this information to infer their relation. We have investigated two increasingly sophisticated strategies to discover and exploit online ontologies for relation discovery. The first strategy, S1, derives a relation between two concepts if this relation is defined within a single online ontology, e.g., stating that $Researcher \sqsubseteq AcademicStaff$. The second strategy, S2 addresses those cases in which no single online ontology states the relation between the two concepts, by combining relevant information which is spread over two or more ontologies - e.g., that $Researcher \sqsubseteq ResearchStaff$ in one ontology and that $ResearchStaff \sqsubseteq AcademicStaff$ in another. To support this functionality, Scarlet relies on Watson to access online ontologies.

Scarlet originates from the design of an ontology matcher that exploits the Semantic Web as a source of background knowledge to discover semantic relations (mappings) between the elements of two ontologies. This matcher was evaluated in the context of aligning two large, real life thesauri: the UNs AGROVOC thesaurus (40K terms) and the United States National Agricultural Library thesaurus NALT (65K terms) [3]. The matching process performed with both strategies resulted in several thousands mappings, using several hundreds online ontologies, with an average precision of 70%.

3.2 The Watson Synonym Service

The Watson Synonym Service[5] is a simple service that creates a base of term clusters, where the terms of a cluster are supposed to be associated to the same sense. It makes use of the information collected by Watson in the form of ontologies to derive these clusters.

The basic algorithm to create term clusters is quite straightforward. Entities in Semantic Web ontologies all possess one and only one identifier (in a given namespace, e.g., we consider *Person* to be the identifier of `http://www.example.org/onto#Person`). They can also be associated to one or several labels, through the `rdf:label` property. Hence the algorithm simply assumes that a term t_1 is a synonym of another term t_2 if t_1 and t_2 are used either as label or identifier of the same entity. Our synonym discovery offline algorithm then simply iterates through all the entities in Watson's ontologies to create clusters of terms that are used together in the identifiers or labels of entities.

[4] `http://scarlet.open.ac.uk/`

[5] `http://watson.kmi.open.ac.uk/API/term/synonyms`

Of course, the quality of the results obtained with this method is not as good as the one obtained with the complex and costly approaches that are employed to build systems like Wordnet. However, the advantage of this algorithm is that its quality improves together with the growth of the Semantic Web, without requiring any additional effort for collecting the data. Quite a high number of good synonyms are found, like in the cluster $\{ending, death, termination, destruction\}$. In addition, this method does not only find synonyms in one language, but can provide the equivalent terms in various languages, providing that multi-lingual ontologies exist and cover these terms. It could be argued that these are not actually synonyms (but translations) and one of the possible extensions for this tool is to make use of the language information in the ontologies to distinguish these cases.

4 Ontology Engineering with Watson

In this section, we look at applications that exploit online knowledge for the purpose of creating new knowledge or enriching existing one.

4.1 The Watson Plugin for Knowledge Reuse

Ontology reuse is a complex process involving activities such as searching for relevant ontologies for reuse, assessing the quality of the knowledge to reuse, selecting parts of it and, finally, integrating it in the current ontology project. As the Semantic Web provides more and more ontologies to reuse, there is an increasing need for tools supporting these activities.

The Watson plugin[6] aims to facilitate knowledge reuse by integrating the search capabilities of Watson within the environment of an ontology editor (the NeOn Toolkit). The resulting infrastructure allows the user to perform all the steps necessary for large scale reuse of online knowledge within the same environment where this knowledge is processed and engineered.

In practice, the Watson plugin allows the ontology developer to find, in existing online ontologies, descriptions of the entities present in the currently edited ontology (i.e., the *base* ontology), to inspect these descriptions (the statements attached to the entities) and to integrate these statements into the base ontology. For example, when extending the base ontology with statements about the class *Researcher*, the Watson plugin identifies, through Watson, existing ontologies that contain relevant statements such as:

- *Researcher is a subclass of AcademicStaff*
- *PhDStudent is a subclass of Researcher*
- *Researcher is the domain of the property isAuthorOf*

These statements can be used to extend the edited ontology, integrating them to ensure, for example, that the class *Researcher* becomes a subclass of a newly integrated class *AcademicStaff*.

[6] `http://watson.kmi.open.ac.uk/editor_plugins.html`

4.2 Evolva: Ontology Evolution Using Background Knowledge

Ontologies form the backbone of Semantic Web enabled information systems. Today's organizations generate huge amount of information daily, thus ontologies need to be kept up to date in order to reflect the changes that affect the life-cycle of such systems (e.g., changes in the underlying data sets, need for new functionalities, etc). This task, described as the "timely adaptation of an ontology to the arisen changes and the consistent management of these changes", is called *ontology evolution* [4]. While it seems necessary to apply such a process consistently for most of the ontology-based systems, it is often a time-consuming and knowledge intensive task, as it requires a knowledge engineer to identify the need for change, perform appropriate changes on the base ontology and manage its various versions.

Evolva[7] is an ontology evolution system starting from external data sources (text documents, folksonomies, databases, etc.) that form the most common mean of storing data. First, a set of terms are extracted from these sources as potentially relevant concepts/instances to add to the ontology, using common information extraction methods. Evolva then makes use of Watson (through the intermediary of Scarlet) to find external sources of background knowledge to establish relations between these terms and the knowledge already present in the ontology, providing in this way the mean to integrate these new terms in the ontology. For this purpose, we devised a relation discovery process that combines various background knowledge sources with the goal of optimizing time-performance and precision.

4.3 FLOR: FoLksonomy Ontology enRichment

Folksonomies, social tagging systems such as Flickr and Delicious, are at the forefront of the Web2.0 phenomenon as they allow users to tag, organize and share a variety of information artifacts. The lightweight structures that emerge from these tag spaces, only weakly support content retrieval and integration applications since they are agnostic to the explicit semantics underlying the tags and the relations among them. For example, a search for *mammal* ignores all resources that are not tagged with this exact word, even if they are tagged with specific mammal names such as *lion*, *cow*, and *cat*. The objective of FLOR [7] is to attach formal semantics to tags, derived from online ontologies and make the relations between tags explicit (e.g., that *mammal* is a superclass of *lion*). The enrichment algorithm that has been experimentally investigated builds on Watson: given a set of tags, the prototype identifies the ontological entities (classes, properties and individuals) that define the tags in their respective context. Additionally, it aims to identify formal relations between the tags (subsumption, disjointness and generic relations) utilizing Scarlet.

The experiments [8] have led to further insights into to nature of ontologies on the Semantic Web, from which we highlight two key ones. First, we found that

[7] An overview of Evolva can be found in [5, 6].

online ontologies have a poor coverage of a variety of tag types denoting novel terminology scientific terms, multilingual terms and domain specific jargons. Secondly, we observed that online ontologies can reflect different views and when used in combination can lead to inconsistencies in the derived structures.

5 End User Applications Relying on Watson

Finally, this section considers applications directly providing new features for users by relying on online semantic content.

5.1 Wahoo/Gowgle: Query Expansion

Wahoo and Gowgle[8] are 2 demonstrators, showing how Web ontologies can be used for a simple application to query expansion in a classical Web search engine. For example, when given a keyword like *developer*, such a tool could find out that, in an ontology, there is a sub-class *programmer* of *developer* and could therefore suggest this term as a way to specify the query to the Web search engine. Without Watson, this would require to integrate one or several ontologies about the domain of the queries and an infrastructure to store them, explore them and query them. However, if the considered search engine is a general Web search engine, such as Google or Yahoo, the domain of the queries cannot be predicted a priori: the appropriate ontology can only be selected at run-time, depending on the query that is given. In addition, this application would require a heavy infrastructure to be able to handle large ontologies and to query them efficiently. Gowgle and Wahoo rely on Semantic Web ontologies explored using Watson instead.

The overall architecture of these applications is made of a Javascript/HTML page for entering the query and displaying the results, which communicates using the principles of AJAX with the Watson server. In the case of Gowgle, Google is used as a the Web Search Engine and the Watson SOAP Web services are employed for ontology exploration (`http://watson.kmi.open.ac.uk/WS_and_API.html`). In the case of Wahoo, Yahoo and the Watson REST API (`http://watson.kmi.open.ac.uk/REST_API.html`) are used.

Both applications use Watson to exploit online ontologies, in order to suggest terms related to the query, that is, if the query contains the word *developer*: 1- to find ontologies somewhere talking about the concept of developer, 2- to find in these ontologies which entities correspond to *developer* and 3- to inspect the relations of these entities to find related terms.

5.2 PowerAqua: Question Answering

To some extent, PowerAqua[9] can be seen as a straightforward human interface to any semantic document indexed by Watson. Using PowerAqua, a user can

[8] `http://watson.kmi.open.ac.uk/wahoo` and `http://watson.kmi.open.ac.uk/gowgle`

[9] `http://poweraqua.open.ac.uk/`

simply ask a question, like "Who are the members of the rock band Nirvana?" and obtain an answer, in this case in the form of a list of musicians (Kurt Cobain, Dave Grohl, Krist Novoselic and other former members of the group). The main strength of PowerAqua resides in the fact that this answer is derived dynamically from the relevant datasets available on the Semantic Web. Note that even if PowerAqua is meant to be an end-user application, lots of effort are still required on interaction and user interface issues.

Without going into too many details, PowerAqua first uses a Gate-based [9] linguistic component to transform a question into a set of possible "query triples", such as <person/organization, members, rock band Nirvana>. The next step consists then in locating, thanks to Watson, online semantic documents describing entities that correspond to the terms of the query triples, locating for example an individual called *Nirvana* in a dataset about music. During this step, WordNet is used to augment the terms in the query triples with possible synonyms. Once a collection, usually rather large, of potential candidate ontologies is found, PowerAqua then employs a variety of heuristics and a powerful matching algorithm, PowerMap [10], to try and find answers from the collection of candidate ontologies. In our example, the query triple shown above can be successfully matched to the schema <Nirvana, has_members, ?x:Musician>, which has been found in a music ontology on the Semantic Web. In more complex examples, an answer may require integrating a number of statements. For instance, to answer a query such as "Which Russian rivers flow to the Black Sea", PowerAqua may need to find information about Russian rivers, information about rivers which flow to the Black Sea and then combine the two. In general, several sources of information, coming from various places on the Web, may provide overlapping or complementary answers. These are therefore ranked and merged according to PowerAqua's confidence in their contribution to the final answer.

5.3 PowerMagpie: Semantic Browsing

PowerMagpie[10] is a Semantic Web browser that makes use of openly available semantic data to support the interpretation process of the content of arbitrary web pages. Unlike Magpie, which relied on a single ontology selected at design time, PowerMagpie automatically, i.e., at run-time, identifies and uses relevant knowledge provided by multiple online ontologies. From a user perspective, PowerMagpie is an extension of a classical web browser and takes the form of a vertical widget displayed on top of the currently browsed web page. This widget provides several functionalities that allow exploring the semantic information relevant to the current web page. In particular, it summarizes conceptual entities relevant to the web page. Each of the entities can then be shown in the text, where the user may initialize different ways of exploring the information space around a particular entity. In addition, the semantic information discovered by PowerMagpie, which relates the text to online semantic resources, is injected into the web page as embedded annotations in RDFa. These annotations can

[10] `http://powermagpie.open.ac.uk`

then be stored into a local knowledge base and act as an intermediary for the interaction of different semantic-based systems.

5.4 Word Sense Disambiguation

Finally, we describe the work by Gracia et al. [11], which exploits large scale semantics to tackle the task of word sense disambiguation (WSD). Specifically they propose a novel, unsupervised, multi-ontology method which 1) relies on dynamically identified online ontologies as sources for candidate word senses and 2) employs algorithms that combine information available both on the Semantic Web and the web in order to integrate and select the right senses. The algorithm uses Watson to access online ontologies. Due to the rich Watson API, they can access all the important information without having to download the ontologies, thus providing an efficient and robust functionality.

The development and use of the WSD algorithm revealed that the Semantic Web provides a good source of word senses that can complement traditional resources such as WordNet. Also, because the extracted ontological information is used as a basis for relatedness computation and not exploited through formal reasoning as in the case of ontology matching, the algorithm is less affected by the quality of formal modeling than Scarlet. What affects the WSD method, however, is that most ontologies have a weak structure and as such provide insufficient information to perform a satisfactory disambiguation.

6 Discussion: the Role of Watson

The availability of large amounts of semantic information is not sufficient in making the Semantic Web to achieve its full potential. Users should be provided with smart, useful and efficient applications demonstrating the added value of the Semantic Web. The development of Watson is guided and informed by the requirements of such applications, which need an efficient and robust system to access the Semantic Web. In this paper, we provided an overview of a number of applications made possible by Watson, with the aim of showing what currently can be achieved with the Semantic Web and, hopefully, to provide inspiration for developers to develop new applications.

In addition, we believe that these applications demonstrate the need for infrastructure components like Watson, offering high level services to support the dynamic selection and exploitation of Semantic Web data. Indeed, a characteristic shared by the presented applications is that they all require to identify and process semantic information at run-time, in an open domain. Considering this, usual approaches consisting in gathering relevant ontologies and datasets in a local infrastructure are not applicable. It is necessary to gather the relevant knowledge dynamically, from various resources available on the Web.

The focus Watson put on providing a complete infrastructure for supporting application development is also essential to make these applications possible. Most of them would not be achievable using any other Semantic Web search engine, as it would require for them to download and process locally the semantic

documents. Apart from the obvious scalability issues this raises, it would also be needed for these applications to include an infrastructure capable of handling at run-time the heterogeneity of the semantic documents available online. Watson provides a common API to access pre-processed semantic documents in an homogeneous and efficient way. In addition, apart from this obvious reason, several of the presented applications use Watson instead of other similar systems either because of the added functionalities it provides (e.g., Watson implements advanced ontology selection algorithms used by PowerMagpie) or because of its robustness (e.g. Scarlet switched from Swoogle to Watson for its ability to handle efficiently a large number of queries without restriction and without crashing).

However, one issue on which additional effort is required concerns the quality of the delivered information. Indeed, as these applications relies on heterogeneous data, coming from anywhere on the (Semantic) Web, it is essential to provide support for assessing the quality of such data and to filter them accordingly. Watson already applies basic mechanisms to improve the quality of the results (e.g., filtering duplicates, extracting complexity information, etc.) A more flexible framework combining both automatic metrics for ontology evaluation and user evaluation should be considered to make possible a more fine-grained, customizable quality ranking of semantic documents.

References

1. d'Aquin, M., Baldassarre, C., Gridinoc, L., Sabou, M., Angeletou, S., Motta, E.: Watson: Supporting Next Generation Semantic Web Applications. In: Proc. of the WWW/Internet conference. (2007)
2. dAquin, M., Motta, E., Sabou, M., Angeletou, S., Gridinoc, L., Lopez, V., Guidi, D.: Toward a New Generation of Semantic Web Applications. IEEE Intelligent Systems (2008)
3. Sabou, M., d'Aquin, M., Motta, E.: Exploring the Semantic Web as Background Knowledge for Ontology Matching. Journal of Data Semantics (2008)
4. Haase, P., Stojanovic, L.: Consistent evolution of owl ontologies. European Semantic Web Conference (ESWC) (2005)
5. Zablith, F.: Dynamic ontology evolution. ISWC Doctoral Consortium (2008)
6. Zablith, F., Sabou, M., d'Aquin, M., Motta, E.: Using Background Knowledge for Ontology Evolution. In: International workshop on ontology dynamics. (2008)
7. Angeletou, S., Sabou, M., Motta, E.: Semantically enriching folksonomies with FLOR. In: Workshop Collective Intelligence & the Semantic Web. (2008)
8. Angeletou, S., Sabou, M., Specia, L., Motta, E.: Bridging the gap between folksonomies and the semantic web: An experience report. In: Workshop Bridging the Gap between Semantic Web and Web 2.0. (2007)
9. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A framework and graphical development environment for robust NLP tools and applications. In: Proc. of the 40th Anniversary Meeting of the Association for Computational Linguistics. (2002)
10. Lopez, V., Sabou, M., Motta, E.: PowerMap: Mapping the Real Semantic Web on the Fly. In: International Semantic Web Conference (ISWC). (2006)
11. Gracia, J., Trillo, R., Espinoza, M., Mena, E.: Querying the Web: A Multiontology Disambiguation Method. In: Proc. of the Sixth International Conference on Web Engineering (ICWE). (2006)