Linked Open Data Perspectives: Incorporating Linked Open Data into Information Extraction on the Web

Linked Open Data Perspectives: Integration von Linked Open Data in Informationsextraktion im World Wide Web

Benjamin Adrian, Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Kaiserslautern,
Andreas Dengel, University of Kaiserslautern & Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Kaiserslautern

Summary Currently, the World Wide Web can be divided into two separate fields. The traditional Web of Documents consisting of hyperlinked web documents and the emerging Web of Data consisting of linked open data. We present ontology-based information extraction as core technology for bridging the gap between both fields. Based on this, we list three basic applications that integrate web data to web documents. Our SCOOBIE system can extract information of a linked open dataset mentioned as textual phrases in web documents. SCOOBIE returns machine interpretable metadata summarizing the content of a web document from the perspective of a linked open dataset. Based on SCOOBIE we present EPIPHANY, a system that returns extracted metadata back to the originating web document in form of semantic annotations. This allows users to request the Web of Data for more information about annotated subjects inside the web document. STERNTALER is a system that analyses extracted metadata from search results of a search engine. It generates semantic filters filled with facets of things that were extracted from web documents inside search results. This allows users filtering those web documents that contain information about specific subjects and facets. ▸▸▸ **Zusammenfassung** Das aktuelle „World Wide Web" lässt sich in zwei Welten untergliedern. Einerseits das traditionelle Netz der Dokumente, bestehend aus verknüpften Webseiten, andererseits das Netz der Daten, bestehend aus offenen und miteinander verknüpften Datensätzen (engl. „Linked Open Data"). Wir stellen ontologiebasierte Informationsextraktion als Basistechnologie vor, um beide Welten miteinander zu vereinen. Drei Anwendungen zeigen hierbei, wie sich das Netz der Dokumente mit dem Netz der Daten anreichern lässt. Beim Analysieren von Webseiten erkennt das SCOOBIE System, ob einzelne Textfragmente als Entitäten eines „Linked Open Data"-Datensatzes weitergehend beschrieben werden. Das Resultat von SCOOBIE sind maschinenverständliche Metadaten, die den Inhalt der Webseite aus der Perspektive des jeweilig verwendeten Datensatzes heraus zusammenfassen. Basierend auf den Resultaten von SCOOBIE präsentieren wir das System EPIPHANY. EPIPHANY reichert das Quelldokument mit von SCOOBIE extrahierten Metadaten an, indem semantische Annotationen über die von SCOOBIE berücksichtigten Textfragmente erstellt werden. Dies erlaubt es Benutzern, weitere Informationen aus dem Netz der Daten über annotierte Textpassagen anzufragen. Das System STERNTALER erweitert eine Suchmaschine, in dem es automatisch Metadaten aus Dokumenten der Suchresultate extrahiert. STERNTALER generiert auf Basis der extrahierten Metadaten semantische Filter, die mit Eigenschaften der im Dokument gefundenen Dingen gefüllt werden. Benutzern wird es hierdurch ermöglicht, solche Dokumente heraus zu filtern, die die gesuchten Informationen zu gewünschten Dingen mit bestimmten Eigenschaften enthalten.

Keywords H.3.3 [Information Systems: Information Storage and Retrieval: Information Search and Retrieval]; H.3.1 [Information Systems: Information Storage and Retrieval: Content Analysis and Indexing]; information extraction, semantic Web, semantic indexing, semantic annotations, semantic filtering, linking open data ▸▸▸ **Schlagwörter** Informationsextraktion, semantisches Web, semantisches Indexieren, semantische Annotationen, semantische Filter, Linking Open Data

1 Introduction

The World Wide Web (WWW) is a treasure cove full of information covering nearly all fields of human knowledge. Until the emergence of Linking Open Data, the WWW was almost solely a web of documents. Most information was embedded in web documents. By 2006, the Linking Open Data (LOD) community effort encouraged people for publishing data to the WWW in a similar way as publishing web documents by using HTTP, URIs, and links (please refer to [1] for more information about Linking Open Data). In consequence, a web of data emerged that added a new facet to the WWW. Figure 1 depicts an excerpt of existing LOD datasets, that have been registered at http://ckan.net/. This LOD cloud shows that like in the web of documents, information in the web of data covers multiple fields of human knowledge.

This paper presents how to incorporate information of the web of data into the web of documents. More specific, we show how to use LOD datasets about a certain field of knowledge (e. g., information about products, encyclopedias, music) as a perspective that provides a specialized point of view on web documents by emphasizing those parts of the document content for which additional knowledge exists inside the LOD dataset. This enables agents to request the web of data for more information about subjects that exist inside the LOD dataset and that are mentioned inside a document's plain text.

For reaching this goal of building a bridge between text and data, we present the novel approaches of ontology-based information extraction (OBIE). The outstanding feature of this OBIE approach is the flexibility provided by initializing the OBIE engine with any existing LOD dataset. Initialized with a LOD dataset, the OBIE-engine extracts textual mentions in documents that refer to single subjects inside this particular dataset. By using our implementation called SCOOBIE (*Service for Ontology-based Information Extraction*), we expose three applications for integrating linked open data into the web of documents:

1. Automatic creation of machine understandable metadata about web documents;
2. Automatic enrichment of web documents with semantic annotations;
3. Semantic filtering of search results with facets about identified instance knowledge.

The following example substantiates these applications with a use case.

1.1 Example

Imagine Sarah, a student who wants to buy a laptop. She searches online and tries to compare different offers, product properties and existing customer reviews. Fortunately, Sarah's search engine is able to consume linked data and extracts relating metadata from web documents. Hence, Sarah initializes her search engine with data about

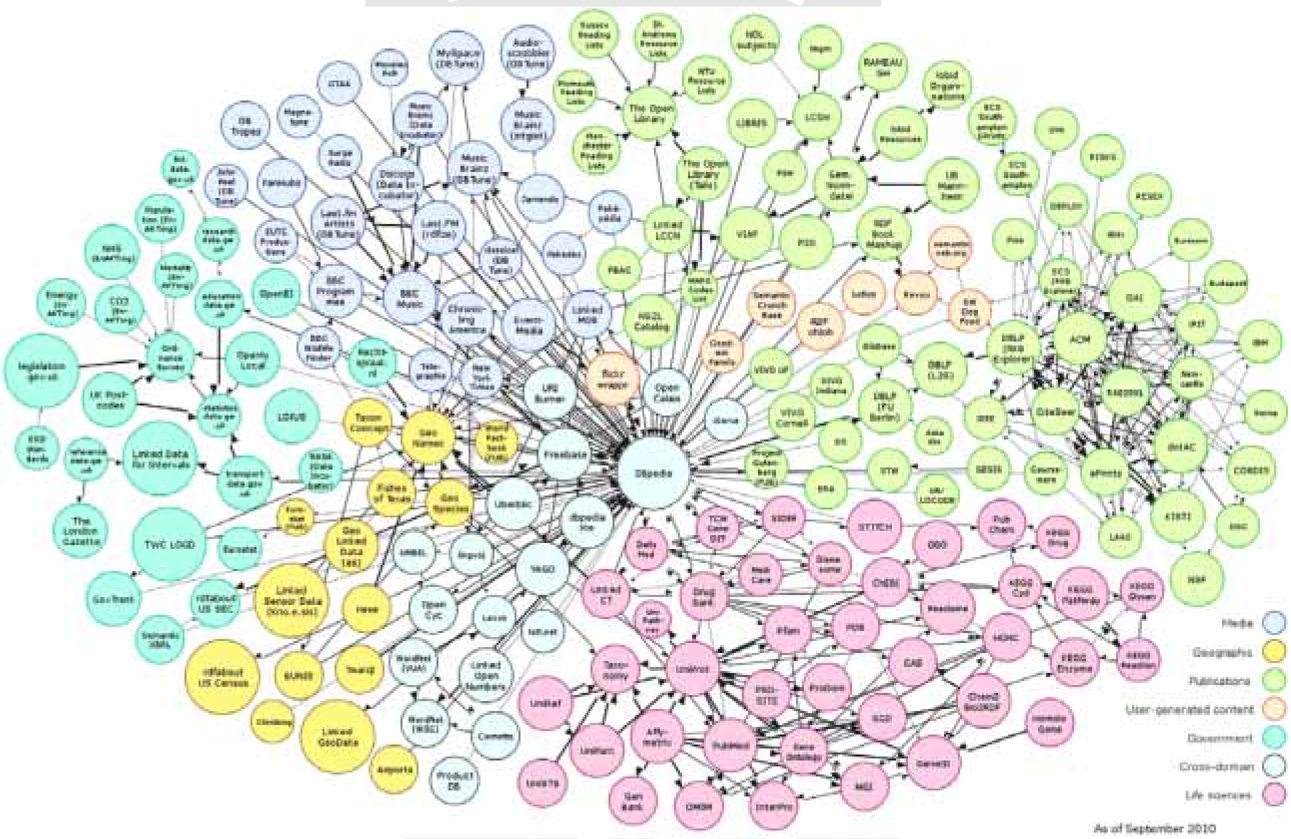

Figure 1 The LOD cloud consisting of published and registered datasets. Linking Open Data cloud diagram, by Richard Cyganiak and Anja Jentzsch, http://lod-cloud.net/.

products, vendors, offers, user generated ratings and experience reports by the online shop Amazon. Now, for each offered product mentioned inside any web document, Sarah can request more information about ratings, offers, or other details from the Amazon dataset. More than that, Sarah's search engine also provides semantic filters on search results that restricts documents in the result list to those facets or products Sarah is currently selecting.

1.2 Outline

The paper is structured as follows. At first, we summarize the current state-of-the-art of Semantic Web technologies incorporating the web of data into text documents. Second, we give an overview about the basic principles of linking open data. Next, we describe ontology-based information extraction (OBIE) and proceed with the applications: metadata generation, semantic annotation, and semantic filtering. Finally, we present evaluation results about applying our OBIE system SCOOBIE on a LOD dataset provided by BBC.

2 Related Work

Even before the existence of the web of data, annotation systems like S-Cream [4] enriched web pages with semantic annotations from domain ontologies, semi-automatically. S-Cream annotations in web pages are not machine-readable. They just highlight text passages, the annotated data is stored back into a domain ontology. S-Cream used the information extraction (IE) system Amilcare, a system without any ontology support. In consequence, S-Cream had to map non-ontological results from Amilcare to particular parts of the domain ontology.

Our OBIE approach directly incorporates LOD datasets into the IE process which provides advantages, i. e., disambiguating possible instance candidates sharing similar labels, using the RDF query language for specifying which entities to extract, or extracting even facts between instances as RDF triples [5].

The Firefox plug-in Piggy Bank extracts information from web documents by screen scrapers. Results are stored in a local or global RDF store [6]. A screen scraper is a piece of Javascript code that extracts RDF information from within a web document's content. A similar approach is GRDDL [7]. GRDDL allows users to add XSLT scripts to web page headers that transform XML data on that page into RDF.

The Open Calais service[1] provides named entity recognition (e. g., *Angela Merkel* as a person's name), instance recognition (e. g., *Angela Merkel* as a *person* with an HTTP URI) and facts with a couple of predefined properties (e. g., *Angela Merkel* is *chancellor*) with focus on News content. Open Calais is ontology-based, returns extraction results in RDF, and maintains Linked Data covering common sense instances (cities, countries, persons, companies). Unfortunately, it depends on its own proprietary dataset and the link coverage to other LOD datasets is very small.

As part of Open Calais the Gnosis Firefox plugin[2] highlights extraction results in text. It lists extracted entities (e. g., person, city) in a sidebar grouped by types. Gnosis renders tooltips while hovering over highlighted text passages with the mouse cursor that contain links to search highlighted text passages in Wikipedia, Google, or the Reuters database. Gnosis does not perform instance recognition nor does it return data in a machine readable format.

Zemanta [11] is a web service for building web mashups. It recognizes relevant web links or images about blog entries. Zemanta also spots for labels of linked data instances of DBpedia or Freebase in web documents.The API can return results in RDF format.

Compared to these systems, characteristic features of our OBIE approach are the *adaptivity* for initializing it with different LOD datasets, and *machine-readability*, as our systems extract data in RDF format from web documents and annotate the RDF in web documents as RDFa.

3 The Web of Data

Publishing data on the WWW is similar to publishing web documents. Resources and data about resources have to be addressed by dereferencable Unified Resource Identifiers (URIs). This allows agents to send HTTP requests for receiving the data about a resource. The data returned by the HTTP request has to be represented in a machine readable format. The Resource Description Framework (RDF [8]) is recommended to be used as knowledge representation format. Linking distributed data about resources is done similar to linking web documents with hyperlinks.

For example, describing the real world person Angela Merkel, DBpedia uses this URI as address: http://dbpedia.org/resource/Angela_Merkel. Data about Angela Merkel is published at an additional address http://dbpedia.org/data/Angela_Merkel. This allows clients request these URIs for getting RDF data about Angela Merkel [3].

3.1 Resource Description Framework

In RDF, information is represented in form of triple statements consisting of subject, predicate, and object. The subject contains the URI of the resource the statement is about. The predicate contains a URI of a property of this resource. The object contains a value of this property and can either be a literal value or again a URI.

If the object is a URI, the RDF statement is called RDF link in terms of the web of data. Multiple RDF links be-

[1] http://www.opencalais.com

[2] http://www.opencalais.com/Gnosis

tween multiple resources create a graph of interconnected and distributed datasets similar to a graph of hyperlinked web pages.

The following RDF statements are written in Turtle syntax [14], one of multiple existing RDF serializations. Turtle allows prefixes (here `dbp`, `foaf`, and `rdf`) substituting common namespaces of URIs with prefix labels. The graph represents data about Angela Merkel, consisting of her name as a literal value, an RDF link to Hamburg typed with a property that describes her birthplace, and finally an RDF link that defines her as member of a class person inside the foaf vocabulary.

```
@prefix dbp: <http://dbpedia.org/resource/>
@prefix foaf: <http://xmlns.org/0.1/foaf/>
@prefix rdf: <http://www.w3.org/1999/02/22-
                rdf-syntax-ns#>

dbp:Angela_Merkel foaf:name "Angela Merkel".
dbp:Angela_Merkel dbp:birtplace dbp:Hamburg.
dbp:Angela_Merkel rdf:type foaf:Person.
```

Vocabularies such as `foaf` provide a description of a whole information field consisting of contained classes and properties. Hence, such a vocabulary formalizes a domain of information. Vocabularies describe details about properties and classes used in a dataset understandable for machines. In consequence, they are referred to as ontologies if they formalize a shared understanding about the conceptualization describing the dataset.

3.2 DBpedia

DBpedia is a linked data wrapper about Wikipedia articles [12]. DBpedia's algorithms transform semistructured parts in Wikipedia articles into a structured RDF representation. Considered parts of articles are: the infoboxes containing property-value pairs, Wikipedia's article categorization system, and hyperlinks between articles. As the RDF data is generated automatically, the data quality is more or less poor. However, as shown in Fig. 1, the DBpedia dataset is the largest part of and a real hub inside the LOD cloud. Most data publishers link their data to DBpedia because it contains many instances covering a large field of information domains.

4 Ontology Based Information Extraction

Our ontology-based information extraction (OBIE) approach can be initialized with RDF data from any LOD dataset. By using the information inside this RDF graph, the OBIE system is able to extract text passages in documents that refer to single instances in the LOD dataset [5]. Hence, for extracted mentions of instances, a lookup to the RDF data may reveal existing facts (here RDF statements) between two or more instances. Extracted results from documents consisting of instances and facts are returned as RDF graph. This output RDF graph describes extracted results by using the same vocabularies as used in the original input RDF graph.

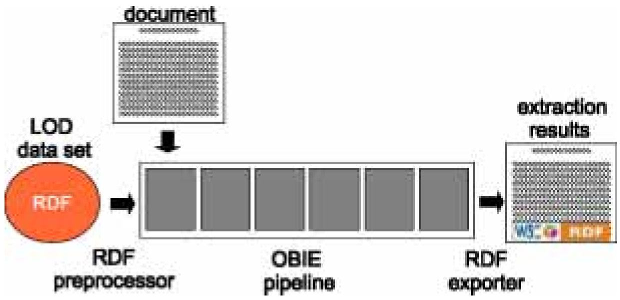

Figure 2 The architecture of SCOOBIE, an ontology-based information extraction.

Our OBIE implementation is called SCOOBIE (Service for OBIE). It builds upon the following components:
1. RDF preprocessor,
2. OBIE pipeline,
3. OBIE tasks,
4. RDF exporter.

Figure 2 depicts the architecture consisting of these components.

4.1 Preprocessing the RDF Graph

The preprocessing step of SCOOBIE transforms the input RDF model consisting of triples into an efficient representation inside a relational database.

At first RDF statements are separated into triples with literal object values called symbols and those with URIs as object values called facts. Symbols and facts are stored into separate tables. Two dictionaries map literal and URI values to integer numbers for minimizing the costs of memory and string comparisons.

4.2 OBIE Pipeline

Based on the results returned by the preprocessor and a given document, SCOOBIE's OBIE pipeline executes a series of OBIE tasks. The pipeline is designed as believing finite-state cascade [10]. Hence, each OBIE task is a believing finite-state transducer transforming passed uncertain input to task specific output by using certain background knowledge. Consumed input and generated output are both hypotheses weighted with belief values in a range between zero and one.

4.3 OBIE Tasks

The following OBIE tasks are processed in the current version of SCOOBIE.

Normalization transforms text content of document formats into a plain textual representation. Here, the Aperture framework[3] is used. A language detection identifies the language the text is written in.

Segmentation partitions a text into units of words and sentences. Words are enriched with POS tags. Based on these POS tags, noun phrase chunking is performed for

[3] http://aperture.sourceforge.net

identifying noun phrases in sentences. The words of detected noun phrases are used to compute a suffix array, which is a list sorted lexicographically. It contains all suffixes of input strings in units of words. A suffix array allows efficient lookups for substrings in $O(\log(n))$ by performing character-based prefix comparisons.

Symbolization recognizes existing literal property values (symbols) of the input RDF graph as mentioned in text. For each entry of the suffix array, a hash value is computed. Here, the hash value consists of the first four characters written in lower case. This set of hash values is used to query the database for a sorted list of property – "literal value" pairs. The hash values ensure that only smaller parts of all literal values inside the database are considered in the following comparison of suffix array entries and the query result. As both list are sorted, the complexity of this match is $O(n+m)$. Matches are returned as symbols that identify text passages as literal values of properties used in the RDF dataset.

Instantiation recognizes instances for extracted symbols by retrieving RDF statements that match with the symbol's property and literal value. The resulting list of URIs is called instance candidates. In the case that multiple instance candidates share the same symbols we call these ambiguous instance candidates. We implemented the following heuristics about cohesion properties between instances for disambiguating these ambiguous instance candidates:

Connectivity. Existing facts between instance candidates heighten their belief values.

Proximity. Facts of instance candidates with identical object values heighten their belief values.

Highest Cohesion Wins. Finally, for ambiguous instance candidates sharing the same content symbols select the instance candidate having the highest belief value. If n instances have equal maximal belief values, divide their belief value by n. The resulting instances are called resolved instances.

Contextualization recognizes facts between resolved instances and heightens belief values of those instances that are connected with facts.

4.4 RDF Exporter

The final tasks of SCOOBIE is called *Population*. It populates an RDF graph called scenario graph with resolved instances, symbols, and facts. As each instance, symbol, and fact is rated with a belief value between zero and one, ranked scenario graphs can be created by passing thresholds to the RDF exporter.

5 Creating Machine Understandable Metadata

SCOOBIE's extracted scenario graph from a document is a formal description or at least a formal summary about the document's content relating to the used LOD dataset. Hence, the RDF described in the scenario graph links the unstructured document to the structured domain of a LOD dataset like DBpedia. Extracting multiple scenario graphs of a document by using SCOOBIE with different LOD datasets generates multiple perspectives. Each perspective summarizes and emphasizes different facets of the document content.

The existence of such kind of metadata allows users to perform sophisticated operations:

1. Query the document content by using RDF query languages like SPARQL;
2. Create semantic mashups combining multiple scenario graphs;
3. Enrich scenario graphs with additional information provided by LOD datasets;
4. Use the structured content of scenario graphs as input for web services (e. g., Google's map or calendar service).

6 Semantic Annotations

A remaining issue is integrating the RDF metadata of extracted scenario graphs back into the textual content of the originating document as semantic annotations. This allows agents to not only reuse specific topics inside the RDF metadata about the document content. It enables them to also access the text passages that mentioned these topics.

Based on SCOOBIE, we created the EPIPHANY service[4] that annotates web documents with information about instances and properties of extracted scenario graphs [13]. The annotations are represented in RDFa [2], an HTML extension that adds RDF data in attribute values of HTML elements.

EPIPHANY parses a web document and compares literal property values of the scenario graph with the content of text nodes within the web document's DOM tree. Finally, EPIPHANY returns a new transformed version of the web document. Around each matching text passage with an existing property's literal value, the new version contains additional HTML SPAN-elements enriched with RDFa markup. The follow HTML+RDFa snippet shows RDFa markup around the phrase "Karl-Theodor zu Guttenberg" annotating it as value of a property foaf:name of a DBpedia instance about the real Karl-Theodor zu Guttenberg.

```
<body
 prefix="dbpedia=http://dbpedia.org/resource/
        foaf="http://xmlns.org/foaf/0.1/">
  ...
 <span about="dbpedia:Karl-Theodor_zu_Guttenberg"
       property="foaf:name">
  Karl-Theodor zu Guttenberg
 </span>
```

EPIPHANY adds stylesheet information to the RDFa enhanced web document that highlights RDFa content with colored borders (see screenshot in Fig. 3).

[4] http://projects.dfki.uni-kl.de/epiphany

Figure 3 Highlighted RDFa markup in a web document.

Figure 4 Requesting additional information about a certain instance from a LOD dataset.

In addition, users clicking on RDFa enriched phrases cause an asynchronous HTTP request to the URI address of this subject that returns more information about this subject in RDF. The resulting RDF information is rendered into an HTML template. Figure 4 shows such a rendered information result in style of a light box.

EPIPHANY's generated RDFa markup integrates metadata directly into text passages of web documents and thus annotates and thereby links these text passages as logical topics of the document to additional information provided by external LOD datasets.

7 Semantic Filtering

The presented EPIPHANY application consumed generated RDF metadata within the scope of a single document. In contrast, our STERNTALER[5] application consumes generated RDF metadata about multiple documents. The goal of STERNTALER is to extract facets about instances extracted from text by SCOOBIE. These facets are used to populate semantic filters. STERNTALER was integrated in a web search engine implemented on the basis of the Google API. For a returned search result list of documents, STERNTALER collects the scenario graphs extracted by SCOOBIE. An analysis of occurring properties and values of instances inside these scenario graphs reveals common properties that are used and rendered as filters by STERNTALER. The values of each property populates each filter.

Figure 5 shows the user interface of STERNTALER that used SCOOBIE on a LOD dataset about products by Amazon. Extracted filters are about product prizes, product ratings, product titles, and overall ratings. The user is now enabled to select certain filter values. This causes STERNTALER to present only those documents

[5] http://projects.dfki.uni-kl.de/sterntaler

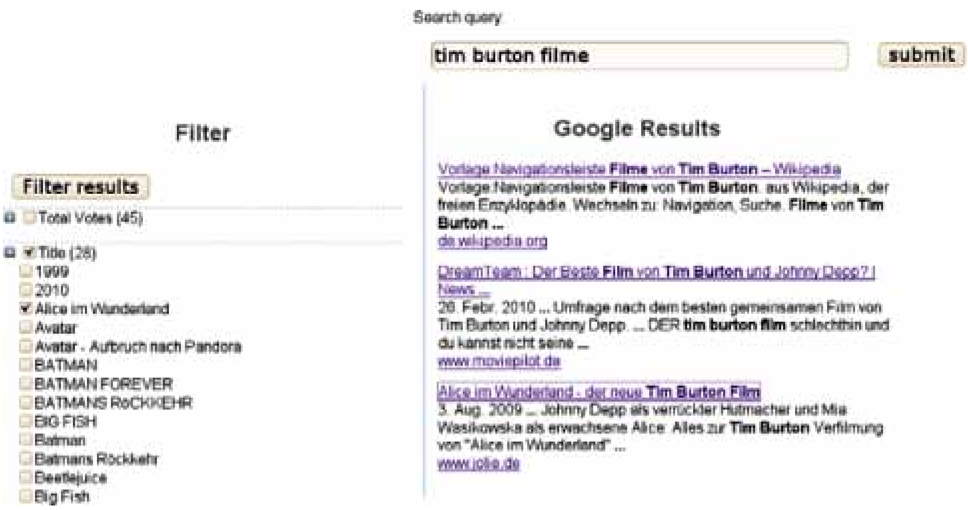

Figure 5 Extending a standard web search with semantic filtering facilities.

in the result list whose scenario graphs contain instances that have known properties and values corresponding to the selected filters and values.

8 Evaluation

We evaluated the SCOOBIE system on an existing corpus of web documents that provide RDF metadata about its content. As data basis, we used web documents from `bbc.co.uk/music`[6] describing biographies about popular music groups. For each biography on a web document, BBC provides RDF metadata about this content consisting of instances representing music groups, members, and solo music artists. The RDF graphs about documents were used as gold standard. Extracted scenario graphs from SCOOBIE for a given web document are compared against corresponding metadata by BBC.

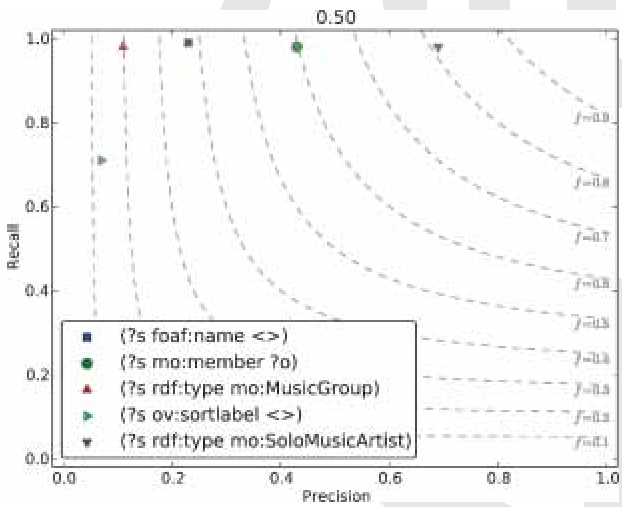

Figure 6 Evaluation results of SCOOBIE.

The chart in Fig. 6 depicts a precision-recall diagram. Precision is the amount of instances extracted correctly from text compared to all overall amount of extracted instances. Recall is the coverage of correct instances extracted from text compared to the correct instances for this document inside the gold standard. The single measure points represent RDF triple patterns that should exist in the scenario graphs. In general, it can be seen that SCOOBIE extracted a good coverage of relevant RDF information from the document's content. Nevertheless, the precision of extracting music groups and general names should be improved. One problem was that nearly all of the web documents contained phrases in the navigation panel, that matched with existing band names (e. g., food, contact). More information about this evaluation and comparisons between SCOOBIE and Open Calais can be found in [13].

9 Summary

We presented approaches for bridging the gap between the web of documents and the web of data. Ontology-based information extraction was presented as core solution for identifying instances inside a LOD dataset as mentioned in text passages. Our SCOOBIE system can be initialized with a LOD dataset. Extraction results summarize the document content relating to this dataset. Results are represented in machine understandable RDF format. Based on this extracted metadata we presented EPIPHANY, a system that integrates extraction results back into the originating document content by enriching the document with additional RDFa markup. Users can request a LOD dataset for more information about an annotated subject inside the web document. The STERNTALER system analyses extracted metadata from search results of a search engine. It generates semantic filters filled with facets of instances that were extracted from documents inside the search result list. Now, users can

[6] http://www.bbc.co.uk/music/developers

filter those documents that contain information about specific subjects and facets.

10 Future Work

Currently, we evaluate a SCOOBIE instance initialized with DBpedia with the content of random pages from Wikipedia. Here, the challenges are to cope with the vast amount of noisy and ambiguous data and still provide acceptable Precision values.

Acknowledgements

This work was financed by the BMBF project Perspecting (Grant 01IW08002) and the German research funding Zentrales Innovationsprogramm Mittelstand (ZIM) in project REMIX (Grant KF2013005SM9).

References

[1] Bizer, C., Heath, T., and Berners-Lee, T.: Linked Data – the story so far. In: *Int'l Journal on Semantic Web and Information Systems (IJSWIS)*, Special Issue on Linked Data, 2009.

[2] Adida, B., Birbeck, M., McCarron, S., and Herman, I.: RDFa Core 1.1 Syntax and processing rules for embedding RDF through attributes W3C Working Draft, W3C. Web page: http://www.w3.org/TR/rdfa-core, 2010.

[3] Bizer, C., Cyganiak, R., and Heath, T.: How to publish linked data on the web. Web page: http://www4.wiwiss.fu-berlin.de/bizer/pub/LinkedDataTutorial, 2007.

[4] Handschuh, S., Staab, S., and Ciravegna, F.: S-CREAM – Semi-automatic CREAtion of Metadata. In: *EKAW 2002: Ontologies and the Semantic Web*, LNCS 2473, LNAI, Springer, pages 358–372, 2002.

[5] Adrian, B.: Incorporating ontological background knowledge into information extraction. In: *Int'l Semantic Web Conf. (ISWC 2009)*, Doctoral Consortium, 2009.

[6] Huynh, D., Mazzocchi, S., and Karger, D.: Piggy bank: Experience the semantic web inside your web browser. In: *Journal of Web Semantics*, 5(1):16–27, 2007.

[7] Conolly, D.: Gleaning resource descriptions from dialects of languages (GRDDL). W3C Recommendation, W3C. Web page: http://www.w3.org/TR/grddl, 2007.

[8] Klyne, G., and Carroll, J. J.: Resource Description Framework (RDF): Concepts and Abstract Syntax W3C Recommendation, W3C. Web page: http://www.w3.org/TR/rdf-concepts, 2004.

[9] Adrian, B., Hees, J., van Elst, L., and Dengel, A.: iDocument: using ontologies for extracting and annotating information from unstructured text. In: *KI 2009: Advances in Artificial Intelligence*, LNCS 5803, LNAI, Springer, pages 249–256, 2009.

[10] Adrian, B., and Dengel, A.: Believing finite-state cascades in knowledge-based information extraction. In: *KI 2008: Advances in Artificial Intelligence*, LNCS 5243, LNAI, Springer, pages 152–159, 2008.

[11] Tori, A.: Zemanta API companion: Zemanta service. Web Page: http://developer.zemanta.com/docs/Zemanta_API_companion, 2008.

[12] Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, S., Cyganiak, R., and Hellmann, S.: DBpedia – A Crystallization Point for the Web of Data. In: *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, 7(3):154–165, 2009.

[13] Adrian, B., Hees, J., Herman, I., Sintek, M., and Dengel, A.: Epiphany: Adaptable RDFa Generation Linking the Web of Documents to the Web of Data. In: *EKAW 2010: Knowledge Engineering and Knowledge Management by the Masses*, LNCS 6317, LNAI, Springer, pages 178–192, 2010.

[14] Becket, D., and Berners-Lee, T.: Turtle – Terse RDF Triple Language W3C Team Submission. Web page: http://www.w3.org/TeamSubmission/turtle, 2008.

Received: March 2011

Diplom Informatiker Benjamin Adrian graduated in Computer Science 2006 and currently is a PhD student at the University of Kaiserslautern. He is working for the German Research Center for Artificial Intelligence (DFKI GmbH). Benjamin is member of the W3C RDFa Working Group. His main research subjects are the application of RDF and Linked Open Data to ontology-based information extraction.

Address: Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Trippstadterstraße 121, 67663 Kaiserslautern, Germany, Tel.: +49-631-205751450, Fax: +49 631 205-751020,
e-mail: benjamin.adrian@dfki.de

Prof. Dr. Prof. h. c. Andreas Dengel is the Managing Scientific Director at the German Research Center for Artificial Intelligence (DFKI GmbH) in Kaiserslautern. In 1993, he became a Professor at the Computer Science Department of the University of Kaiserslautern where he holds the chair "Knowledge-Based Systems" and since 2009 he is appointed Professor (Kyakuin) at the Department of Computer Science and Information Systems at the Osaka Prefecture University. He received his Diploma in CS from the University of Kaiserslautern and his PhD from the University of Stuttgart. He also worked at IBM, Siemens, and Xerox Parc. Andreas is member of several international advisory boards, chaired major international conferences, and founded several successful start-up companies. Moreover, he is co-editor of international computer science journals and has written or edited 10 books. He is author of more than 160 peer-reviewed scientific publications and supervised more than 120 PhD and master theses. Andreas is a IAPR Fellow and received prominent international awards. His main scientific emphasis is in the areas of Semantic Technologies, Agile Knowledge Workflows, Document Understanding, Information Retrieval, Multimedia Mining, and Social Media.

Address: University of Kaiserslautern & Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Trippstadterstraße 121, 67663 Kaiserslautern, Germany, Tel.: +49-631-205751000, Fax: +49 631 205-751020,
e-mail: andreas.dengel@dfki.de