J Intell Inf Syst (2017) 48:141–163
DOI 10.1007/s10844-015-0393-0

Intelligent information processing for building university knowledge base

Jakub Koperwas[1] · Łukasz Skonieczny[1] · Marek Kozłowski[1] · Piotr Andruszkiewicz[1] · Henryk Rybiński[1] · Wacław Struk[2]

Received: 9 January 2015 / Revised: 2 September 2015 / Accepted: 21 December 2015 /
Published online: 18 January 2016

Abstract There are many ready-to-use software solutions for building institutional scientific information platforms, most of which have functionality well suited to repository needs. However, there have already been discussions about various problems with institutional digital libraries. As a remedy, an approach that is researcher-centric (rather than document-centric) has been proposed recently in some systems. This paper is devoted to research aimed at tools for building knowledge bases for university research. We focus on the AI methods that have been elaborated and applied practically within our platform for building such knowledge bases. In particular we present a novel approach to data acquisition and the semantic enrichment of the acquired data. In addition, we present the algorithms applied in the real life system for experts profiling and retrieval.

✉ Łukasz Skonieczny
L.Skonieczny@ii.pw.edu.pl

Jakub Koperwas
J.Koperwas@ii.pw.edu.pl

Marek Kozłowski
M.Kozlowski@ii.pw.edu.pl

Piotr Andruszkiewicz
P.Andruszkiewicz@ii.pw.edu.pl

Henryk Rybiński
hrb@ii.pw.edu.pl

Wacław Struk
W.Struk@elka.pw.edu.pl

[1] Institute of Computer Science, Warsaw University of Technology, Warsaw, Poland

[2] Faculty of Electronics and Information Technology, Warsaw University of Technology, Warsaw, Poland

Keywords Artificial intelligence · Knowledge base · Scientific resources · Repository · Digital library

1 Introduction

The last decade has shown an increased interest among universities in systems dedicated to research data management, providing the access to publicly funded research data. In 2010, a dedicated project, SYNAT, was launched in order to address deficiencies in the scientific information infrastructure in Poland. The main SYNAT construction is based on two levels of distributed knowledge bases – with a central database at the highest level, and university databases at lower levels. The ultimate goal of the knowledge base network is to ensure the nationwide dissemination of Polish scientific achievements and to improve the integration and communication of the scientific community, while also leveraging existing infrastructure assets and distributed resources. In this paper we concentrate on the university level, especially on tools that are devoted to building institutional knowledge bases.

There are many ready-to-use software solutions for building institutional scientific information platforms, most of which have functionality well suited to repository requirements (like e.g. Fedora Commons, or DSpace, see e.g. Berman 2008). However, there have already been discussions about problems with institutional digital libraries (e.g. Davis and Connolly 2007; Salo 2008).

As a remedy to these problems, another approach, which is researcher-centric and community-oriented, can currently be observed. To this end, having considered the needs of universities we have decided to build an institutional knowledge base around the repository, rather than the repository itself. Our main attempts were to find solutions for building such a researcher-centric knowledge base, where in addition to basic document retrieval functionalities one can search for experts, or extract knowledge about individuals and research teams expertise, discover networks of researchers, etc. Following the postulates of several researchers (see e.g. Losiewicz et al. 2000; Wu et al. 2014; Leidig and Fox 2014) we have incorporated into our platform intelligent services, which are aimed at providing users with advanced knowledge about the research carried out at universities. As a result, a platform, called Ω-Ψ^R, has been implemented.[1] The architecture of the knowledge base software is shown by Koperwas et al. (2013), and the extended functionality, and some solutions for the knowledge base system are detailed by Koperwas et al. (2014a, b).

In this paper we focus on the AI methods that have been elaborated and applied practically within the knowledge base platform, thereby reducing human effort in data acquisition, data preparation, and improving information retrieval. In particular we present a novel approach to data acquisition and the semantic enrichment of the acquired data. We will show how this process influences the quality of data, as well as the quality of profiling researcher expertise and performing searches for experts and teams.

The paper is organized as follows. Section 2 summarizes related work. In Section 3 we present the general architecture of the Ω-Ψ^R platform, then in Section 4 we present some of the implemented algorithms related to knowledge acquisition. Section 5 is devoted to the devised semantic tools. In Section 6 the algorithms for extracting expertise of individuals

[1] The system operates as the Research Knowledge Base of Warsaw University of Technology (WUT) under http://repo.bg.pw.edu.pl/index.php/en/

and teams, as well as, searching for experts are presented. We show here how the implemented tools improve the system parameters. Section 7 concludes the paper and presents our future plans.

2 Related work

When surveying contemporary information systems used for building institutional research knowledge bases, one can observe the approach represented by systems like Fedora-Commons, D-space (see e.g. Berman 2008), and dLibra (Mazurek and Werla 2005), which focus mainly on repository functions, such as the storage and indexing of research-related documents, also including aspects of long-term durability. This is a predominant method for the building of institutional research knowledge databases. The systems within this approach provide fairly simple end-user functionality, mainly limited to browsing and querying the repositories. They are bibliography oriented, usually document-centric, and provide end users with neither analytical functionalities, nor with sophisticated presentation capabilities. Additionally, the data acquisition procedures are rather straightforward, based on human work, or harvesting data from well-defined resources.

Although systems of this kind are in wide use, some essential problems have been reported (Davis and Connolly 2007; Salo 2008). The main criticism of the document-centric approach highlights the very weak interest of the research communities in using such repositories, and can be summarized in one sentence, which states that the institutional repository is "like a roach motel—data gets in but never gets out" (Salo 2008).

On the other hand one can see quite a high, and still growing, interest among research communities in systems that are researcher-centric. Good examples here are Google Scholar, Microsoft Academic Search, Arnetminer, ResearchGate and Academia.edu. Some systems of this kind (Scholar, Arnetminer, Microsoft Academic Search) rely heavily on web harvesting mechanisms, others (ResearchGate, Academia) are much more focused on crowd-sourcing.

Unfortunately, such global systems do not cover many of the needs of a typical research institution. One can therefore observe some initiatives towards building institutional research-centered knowledge base systems. A good example is the Stanford VIVO system (Krafft et al. 2010). The VIVO project aimed at creating a "Semantic Web-based network of institutional ontology-driven databases to enable national discovery, networking, and collaboration via information sharing about researchers and their activities". However, many prominent Stanford researchers can still not be found in the system.

Yet another solution has been offered recently. It is a commercial system PURE provided by Elseviere.[2] To a large extent the idea of building the Ω-Ψ^R platform has emerged from similar motivations. However, as the PURE technologies are not public, we have focused on elaborating our owns.

Clearly, while building research knowledge base functionalities, many problems are to be solved with the tools of artificial intelligence and text/data mining (see e.g. Losiewicz et al. 2000; Wu et al. 2014). In the case of Ω-Ψ^R we concentrated on the following issues:

1. data acquisition from WWW, along with extracting information from the retrieved pages and building the knowledge base with the extracted facts;

[2] http://www.elsevier.com/online-tools/research-intelligence/products-and-services/pure

2. semantic enrichment of acquired data and facts by automatic indexing and classification of objects;
3. text/data mining and presentation.

There are numerous generic approaches to both data acquisition from WWW and information extraction. The methods can be generally divided into two groups: those where the HTML structure can be used in the process of extraction, and those that do not rely on the text structure. In the first case, the idea is usually to use the HTML structure for supervised learning (see Crescenzi et al. 2001, Arasu and Garcia-Molina 2003, Chang et al. 2003). These methods perform fairly well for extracting data from somehow standardized pages, for instance in the case of e-commerce pages. However, when researcher profiles are involved, those methods can hardly be used.

The methods of the second group, which is large and more diverse, assume unstructured texts. Extracting data from webpages containing information about researchers and their publications is a specific task of information extraction, and therefore dedicated approaches have been implemented in the Ω-Ψ^R platform. In particular, we have adopted the algorithms based on general sequential patterns (Hazan and Andruszkiewicz 2013), SVM (Han et al. 2003), conditional random fields (Lafferty et al. 2001), and Markov logic networks (Kok and Domingos 2005; Richardson and Domingos 2006).

Referring to the issue (2) mentioned above, we perform semantic enrichment of acquired bibliographic objects in three ways: (a) by providing a publication classifier by using the Ontology for Scientific Journals,[3] (b) by extracting meaningful terms from the documents, and (c) by providing senses to the extracted terms. The use of OSJ for classifying the publications provides the highest level of classification of the publications. For keyword extraction we have proposed a knowledge-poor approach, to an extent inspired by RAKE Rose et al. (2010) and KEA Witten et al. (1999). As our contribution we propose using our meaning discovery algorithm, called SnS (Kozlowski and Rybinski 2014).

Referring to the data mining issues (p. 3 above), in this paper we focus on expert profiling and searching for experts. Both issues are related to each other, namely in both cases the algorithms are based on the characteristics of the achievements of the researcher. However, from the point of view of the end-user they differ essentially—in the case of profiling an expertise, the end-user expects to see a characteristic of the research domain of the expert, whereas in the case of expert retrieval one expects to see a ranking of experts in a domain specified by the user query.

Profiling of a researcher is a process of evaluating the values of various properties that characterize given research. A typical way for representing a researcher's interests is to create a list of relevant keywords. Most of the existing methods use predefined rules or specific machine learning models to extract the different types of profile information (Alani et al. 2003; Yu et al. 2005). Arnetminer relies on rich researcher description created by Web user profiling, i.e. finding, extracting and fusing the "semantic-based" user profile from various Internet sources (Tang et al. 2010).

The literature presents a few approaches to searching for experts. Zeng et al. (2010) try to derive experts from co-authorship networks. Wu et al. (2011) propose the *p-index*, inspired by the well-known *h-index*, to measure the "quality" of a researcher in a given domain. Moreira and Wichert (2013) combine multiple estimators of expertise employing the

[3] http://www.science-metrix.com/en/classification, the Polish part is available under http://omegapsir.ii.pw.edu.pl/download/OSJ_Ontology_103.xls

Dempster-Shafer theory of evidence and Shannon entropy. A graph-based approach, which makes use of bipartite graphs (author-conference) and tripartite ones (author-conference-topics), was proposed by Zaiane et al. (2007). Deng et al. (2008) present two formal models (a language model and a topic-based model) and a hybrid combination for identifying experts in DBLP.

The Ω-Ψ^R knowledge base represents a kind of "semantic network" by means of a variety of interconnected objects, such as publications (full texts), patents, projects led and participated, expert's involvement in the conference program committees, etc. Both our methods are based on interconnectivity between researchers and these objects. For expert retrieval in the first phase all the connected objects are identified, then based on the search result the ranking of the scientists associated with the search results is evaluated. For the expert profiling, in a similar way all the interconnected objects characterizing the expert research are retrieved, then the expertize is evaluated by means of a research domain vector. In both cases the algorithms take into account the results of our semantic enrichment procedure performed on the objects.

3 General architecture of the Ω-Ψ^R platform

There are many public information sources on the Internet that can be used to build a University Research Knowledge Base. These sources are specific in nature, as they can often be changed, removed, or, though absent at a given moment, may appear some time later (like e.g. conference sites or calls for journal special issues). They are usually unstructured, or at best semi-structured. Examples are home pages of conferences that usually change for each separate event in consecutive years.

For this reason one of the key challenges was to provide efficient knowledge acquisition tools that would ensure the system is perpetually filled with new data. To this end, we have implemented a specialized platform for harvesting data from the web. This dedicated knowledge acquisition platform, named Ψ^R (for Platform for Scientific Information Retrieval), contains a set of tools for harvesting pages from the Internet, and then extracting necessary information, which after validation can be incorporated within the knowledge base. The knowledge components are passed to a knowledge repository subsystem (called Ω subsystem), which is responsible for building the knowledge base, and which provides end users with the means for the retrieval and visualization of the information from the knowledge base. Both subsystems are integrated within the knowledge base system, named Ω-Ψ^R.

The architecture of Ω-Ψ^R is presented in Fig. 1. As mentioned above, the Ψ^R part is used for acquiring data from the Internet, then the extracted and validated data enrich the knowledge base and are used by the Ω part. The Ψ^R subsystem consists of the following modules:

- the Web Search Module that finds resources related to the scientific world on the Internet. This module, described in Section 4.1, is triggered by users actions or Scheduler that periodically invokes predefined searches;
- the modules Classifiers and Extractor; the module Classifiers is used to decide if the resources found in the web are of a given type (e.g., conference homepage, journal or publisher's page, etc.); the module Extractor extracts information from the resources that have been found and positively classified as a requested type – such as, for instance, the subject of a master's thesis from a given researcher's homepage, bibliographic descriptions, etc. More details can be found in Omelczuk and Andruszkiewicz (2015);

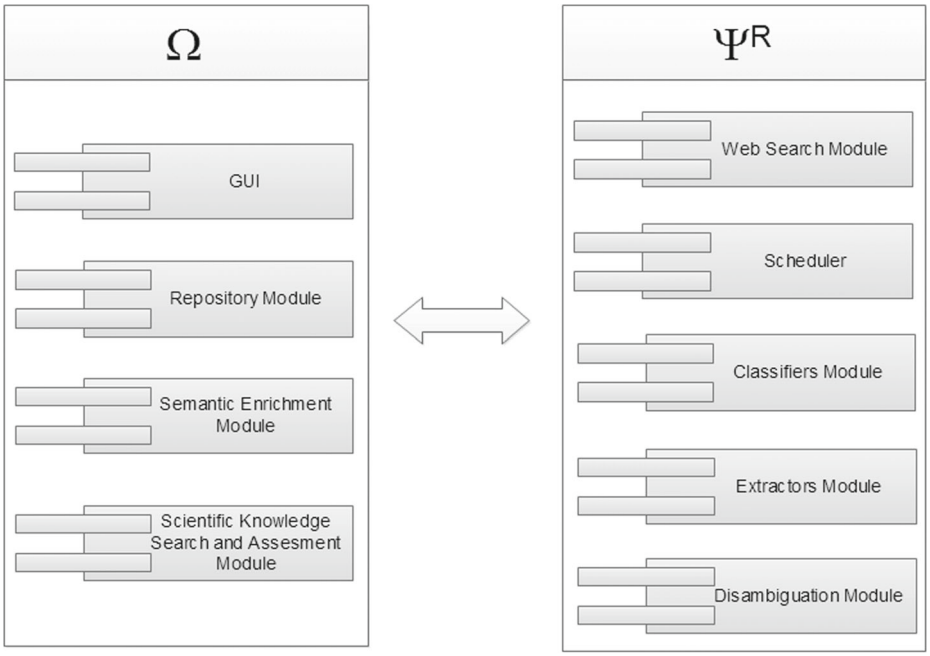

Fig. 1 A general architecture of the Ω-Ψ^R platform

– last but not least, the Disambiguation Module, which assigns publications to the right researcher record from a set of records describing people with the same first and last name.

In the Ω part one can distinguish the following four modules:

– the GUI module, which is responsible for providing extended functionality of the knowledge base. In this paper we do not describe this module, however, a glimpse of it can be found in Fig. 6;
– the Repository module plays the role of a classic repository. It stores metadata and the accompanying digital objects, and provides interfaces for accessing the objects, and also their visualization; additionally, it provides full text retrieval functionality, enhanced by semantic indexing, which is performed by the Semantic Enrichment Module;
– the Semantic Enrichment Module is used to enhance gathered objects by adding meaningful descriptions. More details can be found in Section 5;
– the last module, Scientific Knowledge Search and Assessment, provides analytic functionalities, which are based on already gathered knowledge. In particular, given an individual researcher, or a team, it retrieves the authored research outputs of the individual or team (such as publications, projects, supervised theses, etc.), and then, using the obtained results, it calculates analytical information requested by the end-user, such as e.g. the research interest of a researcher, presented in the form of a cloud (see Fig. 6), or a cooperation graph between researchers. Moreover, based on the university knowledge base that is gathered in the repository, this module is used for searching for experts

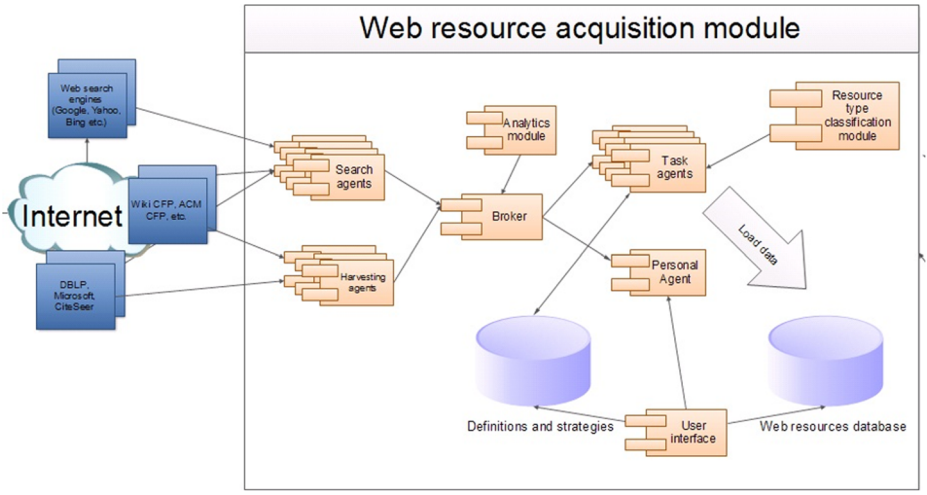

Fig. 2 Web resources acquisition module

with a given expertise, and providing them in the form of a ranking list. The request for experts is expressed by an expertise query. It is described in more detail in Section 6.

In the next sections we will present how in the platform Ω-Ψ^R the pipeline of acquiring data from the Internet is performed, and how the knowledge base is built.

4 Information acquisition

4.1 Unstructured information acquisition

As mentioned above, the platform Ψ^R (Platform for Scientific Information Retrieval) is responsible for acquiring data from the web. Its main component is the Web Resource Acquisition Module (WRAM). The objective of WRAM is to acquire addresses of web resources containing scientific information. The module has been implemented as a multi-agent system. As depicted in Fig. 2, it is divided into the following main sub-modules:

- the information searching, harvesting and information brokering sub-modules;
- the Search Definition and Strategies sub-modules;
- the Task Agents sub-module;
- the Resource Type Classification sub-module.

In order to present the idea of WRAM,[4] we explain the definitions and strategies, then we describe the process of finding relevant web resources. The pseudocode describing the logic of the process of gathering web resources is presented as Algorithm 1.

[4]More details can be found in Omelczuk and Andruszkiewicz (2015).

Algorithm 1 The process of web resource acquisition

```
Data: 𝒮 // strategies
Data: 𝒟 // definitions
Data: 𝒫 // providers
Result: ℛ // a set of web resources, pages
begin
    foreach s ∈ 𝒮 do
        if shallTriggerAction(s) then // execute a search query
            𝒞 ← ∅
            ℛ ← ∅
            d ← s.getAssociatedDefinition
            foreach p ∈ 𝒫 do
                if p.provides(d.searchedObjectType) then
                    // search service provider
                    𝒞 ← 𝒞 ∪ p.search(d)
                end
            end
            foreach c ∈ 𝒞 do
                if not classfiedAsAGivenType(c, d.searchedObjectType) then
                    ℛ ← ℛ ∪ c
                end
            end
            store(ℛ)
        end
    end
    return(ℛ)
end
```

Where:

- $\mathcal{S}$ – a set of strategies,
- $\mathcal{D}$ – a set of definitions,
- $\mathcal{P}$ – a set of search service providers,
- $\mathcal{C}$ – a set of candidates, that is addresses of web pages,
- $\mathcal{R}$ – a set of found addresses of web pages of a given type, e.g., conferences,
- *shallTriggerAction(s)* – a function that checks whether a search parametrized by a definition d associated with a strategy s shall be triggered,
- *s.getAssociatedDefinition* - returns a definition associated with a strategy s,
- *d.searchedObjectType* – a type of the object that should be searched for, e.g., conferences' homepages or journals' homepages
- *p.provides(object_type)* – returns true if a search service provider p is able to search for objects of *object_type*
- *classfiedAsAGivenType(c, object_type)* – returns true if a resource r is of a type *object_type*,
- *store(ℛ)* – stores a set of web resources $\mathcal{R}$ in a database.

As an alternative solution to web crawlers, we proposed a mechanism for defining *query templates* to be used for browsing web space within a given time table, so that a predefined area of knowledge is harvested. The definition should consist of the following elements:

1. the type of web resource, instances of which should be found by the definition,
2. a query template that will be filled in with the given parameters values, and issued in order to find instances of the given web resource type.

This approach helps to solve the problems of knowledge domain identification and the adequate coverage of the search space. An example query template that forms the definition for harvesting the resource type *conference* is shown below:

```
Resource type: Conference;
/*     Defined queries to be issued in order
       to find a conference website: */

parametric / generic search query:   ‘‘{Full_Name}
[Following Year] international conference’’

parametric / generic search query:   ‘‘{Acronym}
[Current Year] international conference’’

parametric / generic search query:   ‘‘{Acronym}
OR {Full_Name} [Previous_Year] international conference’’
```

The above template is designed to search for a specific conference page by: (1) a query with full name of the conference, (2) a query with an abbreviation, (3) a query with the abbreviation ORed with the full name of the conference. So, if for example we are looking for an expected CFP page of the KDD conference:

```
<Full_Name>Knowledge Discovery and Data Mining</Full_Name>
<Acronym>KDD</Acronym>
```

and the current year is 2015, the system will issue the following three queries:

```
’’Knowledge Discovery and Data Mining 2016 international
conference’’
‘‘KDD 2015 international conference’’
‘‘KDD OR Knowledge Discovery and Data Mining 2015 international
conference’’
```

In order to assure complete and up-to-date information in the knowledge base one can define time intervals when specific query templates should be activated. This can be defined as a *harvesting strategy*, which is composed of a query template and a definition of time intervals to run the query. Time intervals can be estimated based on the analyzed resource domain and the expected changes. To sum up, strategies contain information on what the module should search for (specified by the query template, e.g. for querying *conferences, universities*), and how often it should be activated (e.g., once a week).

Figure 2 presents a multi-agent environment, responsible for finding relevant web resources. When the time comes, and a search should be performed according to a strategy, the Strategy Agent invokes the Task Agent. It causes the delivery of a search query template to the Broker Agent. The Broker Agent executes the search (connects to the search agents and then to particular data sources). In the next step, it aggregates responses. Then the Task Agent receives the results (URLs of web resources), and transfers them to the corresponding Classifier module for a classification process. The most appropriate classifier is used to decide whether a given URL is of a desired type, e.g., *conference home page*. In the end, resources (URLs) are inserted into the web resources database, together with metainformation about the classification and definition of the current search. In contrast to the Task Agent, the Personal Agent is designed to trigger on-demand searches that the user wants to perform.

In the system a variety of resources are distinguished, e.g. *person, university, conference, publisher,* etc. A query sent to the search agents is defined in terms of a key-value string, e.g. `conference:ICAART; year:2013`, then depending on the data source interface it is converted to the most suitable query for a given data source (i.e., Google, Yahoo, Bing). The name or the short name of a conference is fetched from the repository (from the list of conferences), and having completed the process of searching for additional information, a given conference record is enriched with the found information.

The main idea behind the classification module is to put all the tasks related to the classification in one place. The module supplies simple interfaces for agents and classifiers implementations. The implementation of `Classifier` is generally not considered a part of the module, and can be developed separately, as a brand new or a library wrapper. The main function of the module is the classification of a website. Its interface is simple and consists of variants of invoking classifiers. In the simplest case, only a website URL and resource type are needed. There is no need for an external system to manage the algorithm selection and configuration. However, it is possible to indicate an algorithm and the configuration parameters for a given object type. In the classification module we use two types of classifiers, namely, SVM and Naive Bayes. However, the classification module is designed for easy incorporation of other classifiers to be used in the system, the ones existing in the library (and utilized as library wrappers), or the ones developed.

4.2 Acquisition of publications

The process of unsupervised acquisition of publications from the web consists of three steps: searching for publications, extracting bibliographic metadata and finally merging acquired data into the knowledge base, which includes the detection of duplicates, disambiguation of the names, and also integration. The first step is realized by the Ψ^R module (Fig. 1), which works as an alternative solution to a focused web crawler. It periodically asks various search engines for the publications of WUT authors, conferences and journals.

The second step is performed by the Zotero software.[5] Zotero was developed as a browser extension, so it was not straightforward to build an application on top of it as the module expects to interact with the user. We implemented the Zotero-based web server and use it to extract metadata from websites containing bibliographic entries, and to convert them into BibTex.

The last step, i.e., importing BibTex into the repository, is performed in the following manner. The Bibtex obtained from Zotero is converted into a native xml format, which represents publications in the form of a tree-like structure (Fig. 3). The tree nodes represent bibliographic elements, which might be shared between many publications, e.g.

[5]Zotero is a free and open-source Firefox extension for managing researchers' bibliography (https://www.zotero.org/, see also e.g. Fernandez 2011). It performs automatic detection and retrieval of bibliographic entries from a variety of sources including publishers' databases, like Springer, Scopus or IEEE, as well as, publication search engines like Google Scholar or CrossRef. It is worth mentioning that the Ω-Ψ^R system itself supports Zotero users, as well.

Koperwas Jakub Janusz, Skonieczny Łukasz, Rybiński Henryk, Struk Wacław: Development of a University Knowledge Base, w: Intelligent Tools for Building a Scientific Information Platform: Advanced Architectures and Solutions / Bembenik Robert *[i in.]* (*red.*), Studies in Computational Intelligence, vol. 467, 2013, ISBN 978-3-642-35646-9, ss. 97-110, DOI:10.1007/978-3-642-35647-6_8

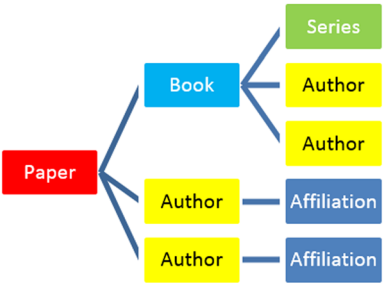

Fig. 3 A bibliographic entry and its tree-like representation

authors, books, journals, series. Each tree node has its own properties (e.g. *title, name, surname*). Every element of the tree has to be checked for its existence in the repository, then merged and integrated with the existing "objects". We have implemented a general top-down method for the tree matching (Algorithm 2), based on similarity queries. The procedure for each type of tree node, however, can be overridden. The matching for authors, for example, is performed by the algorithm which performs not only matching and merging, but also name disambiguation.

The name disambiguation process is one of the most important steps in integrating the acquired data with the existing knowledge base. In the process we have a set of publications with a given text representing authors' names and a set of researchers (also with text strings representing their names), and we would like to assign each publication to the correct researcher(s). The problem is complicated because in the publications there are usually various forms of researchers' names (e.g., with or without a middle name) and for popular names it is often the case that various persons hold the same pairs of names (first name, and surname).

To deal with the disambiguation problem, as a starting point we used the algorithm proposed by Tang et al. (2010), which consists in grouping publications with matching authors' first and middle names, and then clustering each group, taking into account co-authorship, citations, extended co-authorship, and user restrictions. When comparing this with the original algorithm, for the distance measure in the clustering algorithm we added the similarity of the titles, as it is known that scientists often use the same words in their publication titles. Additionally, we proposed a genetic algorithm that makes our approach different from the others. Moreover, we developed a clustering method that iteratively assigns publications to groups. The disambiguation algorithm is the subject of a separate paper, which is currently under preparation, we therefore do not describe it here.

Algorithm 2 Function match(Entry e) //Bibliographic entry tree matching

```
Data: e // entry to be matched against objects from repository
Result: r // matched and/or merged entry
begin
    R ← similaritySearch(e)// a list of entries from repository similar to
    e, sorted by similarity to e
    if R.first.similarity > HIGH_SIMILARITY and
    R.second.similarity < HIGH_SIMILARITY then
        r ← R.first
        return r
    end
    if R.first.similarity > MEDIUM_SIMILARITY then
        r ← Present R to the user and ask him to select best matching
        if r! = null then
            return r
        end
    end
    r ← e
    foreach c ∈ entries encapsulated in r do
        c ← match(c)
    end
    return r
end
```

5 Semantic enrichment

Semantic processing aims at enriching acquired objects by adding semantically meaningful descriptions and labels in order to improve information retrieval. Additionally, it influences the quality of profiling researchers by discovering their research areas. As a side-effect, it also improves the quality of searching for experts. The processing is performed on repository documents (publications, theses, patents, etc). Two special semantic resources are used for this purpose, namely Ontology for Scientific Journal[6] (hereafter OSJ), and Wikipedia resources. The process consists of:

1. classifying publications by assigning them OSJ categories (domains, fields, subfields);
2. indexing with keywords – extracting semantically meaningful descriptors from the documents;
3. sense indexing – inducing senses of a given term and labeling text with them.

The target goals of this process are: (1) the retrieved OSJ publication categories are mainly used for building maps of research areas for individual researchers, and then, propagating the researchers interest to the affiliation-related university units; steps (2) and (3) are crucial for enriching texts with semantic labels, which are also used for building researcher interest vectors (visualized in the form of word clouds); mainly, however, they are used for improving search parameters, such as precision and recall. Below we describe the three modules in more detail.

[6] http://omegapsir.ii.pw.edu.pl/download/OSJ_Ontology_103.xls

Table 1 The results of publication classification accuracy (in %)

Science domain	Single mode (%)	Tree mode (%)
Health sciences	66	82
Natural sciences	74	90
Applied sciences	68	88
Economic and social sciences	66	84
Art and Humanities	64	82

5.1 Publication classifier

Scientific domain classification is the task of providing a publication with one or more relevant tags, which in turn assign the publication to one or more scientific classes. In Ω-Ψ^R we have decided to use OSJ ontology as a classification schema. OSJ is a three levels hierarchy, ended with the leaves on the last level—these are simply scientific journal titles, so the path in OSJ from the root to a leaf (i.e., a journal title) assigns domain tags to the papers from the journal. The levels in the OSJ hierarchy are respectively *scientific domain, field,* and *subfield.* Clearly, OSJ can be used directly for assigning tags to all the papers published in the journals that are contained in the OSJ list. The problem appears for the publications outside of the OSJ journal lists, as well as theses, publications that are conference papers, chapters in books, etc. To this end, we have designed and implemented a Bayesian classifier model, which was trained on the OSJ papers. So, the science domain classifier works as follows for each document:

1. if the document is a paper from the OSJ list, take the tags assigned by OSJ to the journal;
2. otherwise, use the model of Bayesian classifier on the available metadata, preferably including title, keywords and abstract, and use the resulting OSJ categories to classify the document.

We verified two solutions: one classifier for all the OSJ fields (single mode), or a tree of specific classifiers (tree mode), each node representing a "specialized" classifier. Our experiments have shown that the solution with the tree of "specialized" classifiers outperforms one common classifier (see Table 1). The tree of classifiers is a hierarchical structure with the depth of 2, where each node represents a specialized classifier. The root is a classifier for the first OSJ level, its children are composed of 6 classifiers at level 2 (for each OSJ domain there is one field classifier built). An average accuracy (10-fold cross validation) in tree mode has reached 85 %.

5.2 Extraction of keywords

Extraction of keywords plays a crucial role in enhancing the intelligence of enterprise search. Keywords may be retrieved from an original text in order to summarize it, or they can be acquired from structured knowledge resources like ontologies, dictionaries, lexicons. Nowadays, most semantic resources cover only specific domains. Bearing in mind that a whole University research domain cannot be covered by one specific domain ontology, we have decided to apply Wikipedia (Polish and English) as a semantic knowledge resource and implement Wikipedia-based semantic indexing of documents in the Ω-Ψ^R system.

Firstly, we propose a novel method for keyword extraction. It works on a single document, and extracts keywords from the text. The approach is knowledge-poor, i.e. it does not

use any external knowledge resources, including Wikipedia. The approach is inspired by RAKE (Rose et al. 2010) and KEA (Witten et al. 1999).

KEA (Keyphrases Extraction Algorithm) is an algorithm for extracting keyphrases from text documents. First, it creates a model that learns the extraction strategy from manually indexed documents. The Naive Bayes classifier is trained using a set of manually labeled documents. KEA extracts n-grams of a predefined length (e.g. 1 to 3 words). For each candidate phrase KEA computes 4 feature values: $tf\text{-}idf$, first occurrence (terms that tend to appear at the start or at the end of a document are more likely to be keyphrases), length (number of component words), and node degree (only in a case when a thesaurus is used). While extracting keyphrases from new documents, KEA takes the Naive Bayes model and feature values for each candidate phrase and computes its probability of being a keyphrase. Phrases with the highest probabilities are retrieved as the final keywords.

Contrary to KEA (which is a supervised method), RAKE is an unsupervised, domain-independent, and language-independent approach. It is based on the observation that keywords frequently contain multiple words but rarely contain standard punctuation or stop words. First, the document text is split into sequences of contiguous words at phrase delimiter and stop-word positions. Words within a sequence are assigned the same position in the text and together are considered a *candidate keyword*. When all candidate keywords are identified and the graph of word co-occurrences is complete, a score defined as the sum of its member's word scores is calculated for each candidate keyword.

Compared to the above methods, the proposed solution (called hereafter TKE) has a dedicated lemmatizer, Part-of-Speech filters, and candidate evaluation method, which is combined from the statistical function ($freq(w)$), and the Naive Bayes classifier. The algorithm is based on a candidate selection method exploiting a set of PoS rules. It starts with splitting text into sentences. Each sentence is represented as a sequence of words. Next, the words are normalized (lemmatized), and tagged with Part-of-Speech properties. Such preprocessed sentences are transformed into the set of n-grams (with a predefined length of 1 to 3 words). Keyword candidate selection is performed in order to find a finite number of potentially significant words. It starts from the longest n-grams and matches each n-gram against the well defined PoS rules. The positively verified ones are called candidates, and the n-grams which are parts of them are not further processed. Finally, the candidates are evaluated by the models (the classifier and the statistic function, which are combined with the equal weights), and the ones with the highest scores are retrieved.

In order to compare our algorithm (TKE) with RAKE and KEA we have performed experiments using a set of Polish abstracts (with the size of up to 2200 characters) gathered from the WUT repository. The experiments have shown that TKE achieves better quality measures (precision, recall, F-measure) than RAKE and KEA (Table 2).

Unfortunately, the methods discussed above have a drawback: none of them is able to assign to the document such keywords that do not appear in the document. This means that often documents cannot be tagged with generalizing keywords, such as e.g. the higher-level

Table 2 The results of keyword extraction (in %)

Method	Precision (%)	Recall (%)	F-measure (%)
RAKE	5	13	7
KEA	11	34	17
TKE	14	45	21

words that are abstract categories (called meta-tags), which are very important from the point of view of search quality and granularity level.

We propose resolving this drawback by using external knowledge resources, such as Wikipedia. For a few years, both Wikipedia and DBpedia have been used in many areas concerned with natural language processing, in particular for information retrieval and information extraction. In our approach, we use Wikipedia in the term oriented way (focused on semantic information about an analyzed term). This approach is inspired by Medelyan et al. (2009), and Milne and Witten (2008). We process data in two steps. First, given a keyword extracted from the processed document using TKE, the module searches for an article with a title equal to or at least containing the term. Then the found article is processed in order to extract its labels, categories, and translations.

Additionally, for each term to be assigned we extend this approach by sense indexing. There are many words that have different meanings, and hence different semantic description. If, for example, we search in Wikipedia for articles with the title containing the term *bass*, we retrieve dozens of them referring to semantically far distanced entities. Therefore, each word having many Wikipedia articles assigned is disambiguated. We first perform sense induction (we build the sense repository for the given term using the Wikipedia raw corpora), then the most likely sense is used as the semantic label for the analyzed word, along with the accompanying contexts. Having senses discovered in Wikipedia, represented as context vectors, the appropriate sense is chosen by intersecting them with the context words in the input text, then the largest intersection points out the final sense, giving the resulting Wikipedia article to be assigned to the input text. The word sense induction part is performed by the SenseSearcher algorithm (Kozlowski and Rybinski 2014), briefly described below.

5.3 Sense indexing

For word sense induction we have used the Sense Searcher algorithm (hereafter called SnS) which is based on closed frequent termsets. It provides as a result a tree of senses, and represents each sense as a context. The key feature of SnS is that it finds infrequent and dominated senses. It can be used on the fly by end-user systems, and the results can be used to tag the keywords with the appropriate senses.

SnS consists of five phases. In Phase I, a full-text search index is built using a provided set of documents. In Phase II, we send a query with a given term to the index, and retrieve elements (paragraphs/snippets), which describe the mentioned term. Then the paragraphs/snippets are converted into a context representation (bag-of-words representation). In Phase III, significant contextual patterns are discovered in the contexts generated in the previous step. The contextual patterns are closed frequent termsets occurring in the context space. In Phase IV, the contextual patterns are formed into sense frames, which build a hierarchical structure of senses. Finally, in Phase V sense frames are clustered in order to merge similar frames referring to the same meaning. Clustered sense frames represent senses.

An extensive set of experiments performed by Kozlowski and Rybinski (2014) confirms that SnS provides significant improvements over existing methods by means of sense consistency, hierarchical representation, and readability. We tested SnS as a web search result clustering WSI-based algorithm. These experiments aimed at comparing SnS with the other WSI algorithms within the 2013 SemEval Task no 11 (Navigli and Vannella 2013).

The SemEval 2013 Task 11 is measured by a diversified number of indicators: Rand Index (RI), Adjusted Rand Index (ARI), Jaccard Index (JI) and F1 measure We show the

results for those four measures in Table 3. As one can see, SnS outperforms the best systems that participated in SemEval - HDP based methods. The SnS-based system reports considerably higher values in RI and ARI. It achieves significantly better results in terms of F1. In the case of JI the best values of SnS and UKP-WSI-WACKY-LLR are similar. Generally, SnS obtains the best results in all measures. To get more insight into the performance of the various systems, we calculated the average number of clusters and the average cluster size per clustering produced by each system, and then compared this with the gold standard average. The best performing system in the case of all above mentioned categories has the number of clusters and cluster size similar to the gold standard. SnS reports results similar to the HDP algorithms (the best ones in the WSI class). The expected values of number of snippets per query is close to 64.

6 Expert profiling and search

In this section we will present two important features implemented within the platform Ω-Ψ^R, namely expert profiling and searching for experts. From the point of view of the end-user the two functionalities differ essentially—in the case of searching for experts one expects to see a ranking of experts in a domain specified by the user query, whereas in the case of profiling an expertise the user expects to see the characteristics of research domain of the expert. The algorithms implementing the two features are similar to each other, namely in both cases they are based on the characteristics of the achievements of the researcher. What is important, is that for the algorithms it is not the declared keywords that are the most essential, but various researcher-related objects contained in the repository which characterize his/her scientific achievements, such as publications, patents, projects, and activities. This means that the functionalities work properly if the following conditions are satisfied:

1. the knowledge base represents a kind of "semantic network" by means of a variety of interconnected objects, such as publications, patents, projects led and participated in, expert' involvement in conference program committees, etc.
2. the knowledge base is as complete as possible.

Below we present the features in more detail.

Table 3 The results of clustering experiments on SEMEVAL data set (in %)

Type	System	RI	ARI	JI	F1
WSI	HDP-CLS-LEMMA	65.22	21.31	33.02	68.30
	HDP-CLS-NOLEMMA	64.86	21.49	33.75	68.03
	SATTY-APPROACH1	59.55	7.19	15.05	67.09
	DULUTH.SYS9.PK2	54.63	2.59	22.24	57.02
	DULUTH.SYS1.PK2	52.18	5.74	31.79	56.83
	DULUTH.SYS7.PK2	52.04	6.78	31.03	58.78
	UKP-WSI-WP-LLR2	51.09	3.77	31.77	58.64
	UKP-WSI-WP-PMI	50.50	3.64	29.32	60.48
	UKP-WSI-WACKY-LLR	50.02	2.53	33.94	58.26
WSD	RAKESH	58.76	8.11	30.52	39.49
SNS	SNS	65.84	22.19	34.26	70.16

6.1 Expert profiling

Based on the evidence stored in the repository (authored publications, reports, patents, projects, etc.), the system builds for each researcher a profile in the form of the expertise vector. The vector is visualized in the form of a word cloud, when the researcher profile is displayed. In general, the word cloud can be used to visualize not only the areas of interest of a single person, but also that of a team or unit.

Given a person p, we denote as $D(p)$ the set of all documents associated with the person p, i.e. authored publications, patents, supervised theses, and even the description of the research activities of the department, to which the researcher is affiliated. By $keywords(d)$ we denote the function which returns a set of terms which relevantly describe document d. More precisely,

$$keywords(d) = OSJ(d) \cup EXTRACT(d) \tag{1}$$

where

- $OSJ(d)$ denotes the function providing for the given text d the higher level classification keywords from the OSJ ontology, as described in Section 5.1;
- $EXTRACT(d)$ denotes the function that for a given d provides keywords characterizing semantically d, as described in Section 5.2.

So, given person p we have the dictionary characterizing his/her research – $K(p) = \bigcup_{d \in D(p)} keywords(d)$. For each keyword $k \in K(p)$ the "keyword score", denoted as $score(k, d)$, is calculated in such a way that its value depends on the role of the keyword in the document and the scientific value of the document:

$$score(k, d) = \sum_{d \in D(p)} rel(k, d) \times (sif(d) + 1) \tag{2}$$

where

- $rel(k, d)$ measures the relevance of keyword k with respect to document d; Usually it is the value of $tf\text{-}idf$; however, in the case of publications and technical reports the values of the OSJ keywords and the keywords provided by the authors are boosted;
- $sif(d)$ is a scientific impact factor of the document d. For the journal papers this is a linear combination of the impact factor of the journal and the citations of the document. Arbitrary values are given to other objects, like conference papers, patents, supervised theses.

Once the scores are calculated, the set of keywords P_k is sorted by descending "keyword score" and presented to the user in the form of word-cloud.

Clearly, the same algorithm can be used for building an expertise vector for any subset of documents. So, given a query q we can obtain the set of documents $D(q)$ and calculate the score vector the same way as $D(p)$ for researcher p. This means that we can easily obtain an aggregated cloud of research interest for a faculty, department, as well as for a whole university. The profiles for a whole university and two different faculties can be seen in Fig. 4.

We have compared clouds resulting from our approach with the expertise description available at ResearchGate,[7] where the expertise is built manually by peers. Figure 5 shows the side-by-side comparison of research areas of two experts, calculated and visualized in

[7] https://www.researchgate.net/home

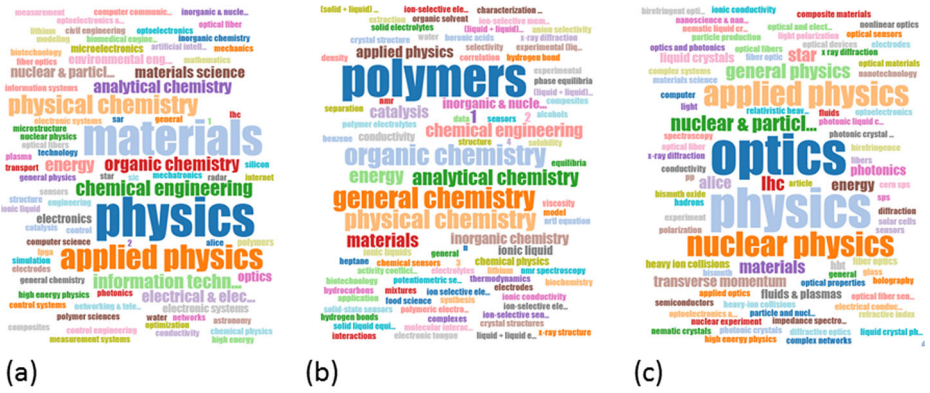

Fig. 4 Aggregated expertise clouds for a university (**a**), and two faculties (**b**) and (**c**)

Ω-Ψ^R, and the ones in ResearchGate. In general, one can note that the clouds generated automatically by Ω-Ψ^R are compatible with the ones made manually, although the clouds generated by Ω-Ψ^R visualize much better the proportions between various areas. In addition, with a closer look we can observe that the cloud generated by Ω-Ψ^R for Expert 2 is richer than the profile be ResearchGate. It results from the fact that Expert 1 is followed by more peers, whereas Expert 2 is beginning his presence in the ResearchGate and has very few followers. In such cases the automatic cloud is more relevant.

6.2 Searching for experts

The method of searching for experts is in general conventional and similar to the one proposed by Aberer et al. (2011). So, the main idea is that for a query q the system performs a search for all types of resources in the repository, and then assigns some weights to the retrieved objects, groups the objects around persons linked to the objects, and calculates the resulting scores for the persons, according to their roles (contributions), such as authorship of a publication, supervising a thesis, leading a project, etc. In this approach the quality boost comes from the fact that the back-end knowledge base is compound and semantically enriched. Figure 6 illustrates the result screens of a search for an expert. The search is based on the following components:

- the modules performing semantic enrichment of the repository objects, described in Section 5, i.e. OSJ classifier, keywords extraction, and sense disambiguation with SnS. The modules build major fields of the suggested expertise;
- search engine: the Ω-Ψ^R search engine is built on top of the Apache Lucene library;[8] it provides full-text search, and a rank on the basis of the well-known tf-idf measure;
- ranking modules, which are definable by the system administrator with a special scripting language; with such an approach one can implement very specific rank algorithms, which can provide specific weightings to particular evaluation elements (e.g. impact factor of the journals, number of citations, special ranks for managing projects, etc.);

[8] Apache Lucene is an Open Source Apache project devoted to advanced full text retrieval (see https://lucene.apache.org/)

Fig. 5 Research areas of Expert 1 (**a**), and Expert 2 (**b**) in Ω-Ψ^R (*left*) and ResearchGate (*right*)

actually, it is possible to define many ranking algorithms, and make them available to end-users, so that they can search for experts, according to specific criteria;

– result presentation – the identified experts are presented in a table in a given order, according to the selected ranking algorithm, along with the experts' portfolios, which contain all the details about their activities. The results are provided with the ranking score bar.

Below we present a general layout of the ranking algorithm:

– all the knowledge base resources are searched with a specified search phrase q (formulated the same way as for searching publications, theses, etc.). The result of the query q is denoted by $D(q)$;

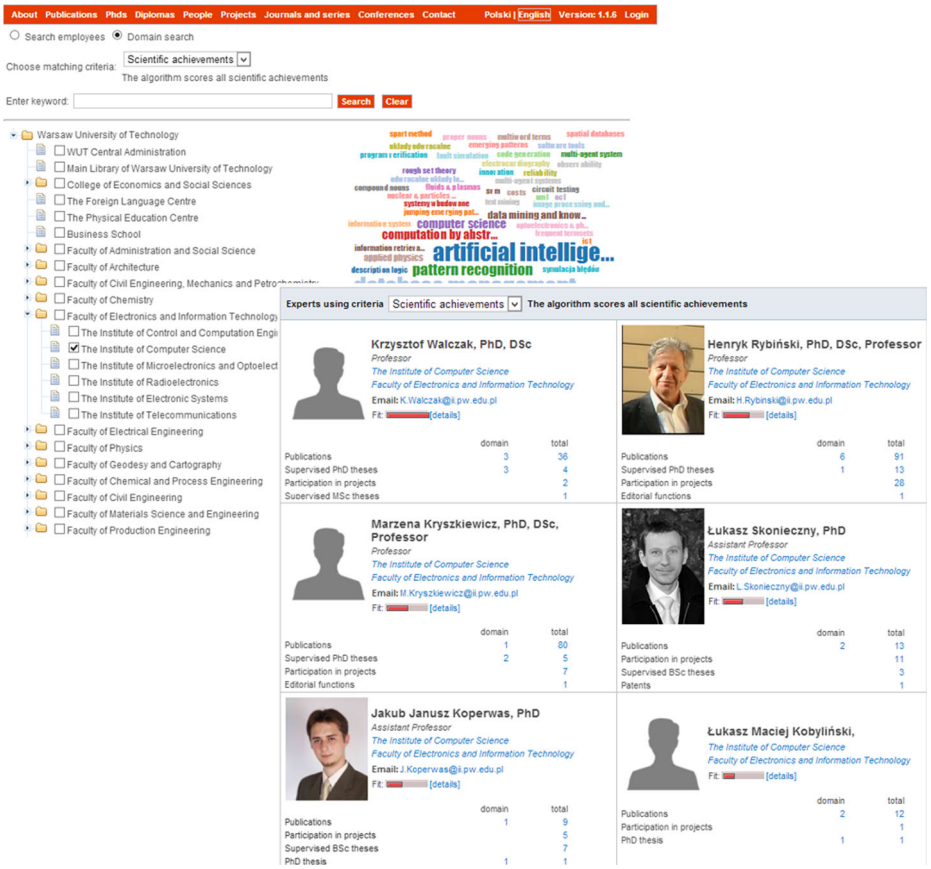

Fig. 6 A search for experts in a domain and the result page

– a set of all persons $P(q)$ related to items from $D(q)$ is calculated as follows:

$$P(q) = \bigcup_{d \in D(q)} \{p : role(p, d) > 0\} \tag{3}$$

where the function $role(p, d)$ provides a measure of relevance of role of person p in elaborating the document d; by $role(p, d) > 0$ we mean that p has some role in d, that is, p is an author of publication d, or is a supervisor of thesis d, or is a leader of project d, etc.; in particular, the function takes into account the roles which can be *article author*, *book author*, *book editor*, *phd author*, *phd supervisor*, *master thesis author*, *master thesis supervisor*, *project member*, *project leader*; in this way we can, e.g., value the role *book author* more than *book editor*;

– for each person $p \in P(q)$ the person score measure, denoted by $Pscore(p, q)$, is calculated:

$$Pscore(p, q) = \sum_{d \in D(q): role(p,d) > 0} score(p, d, q) \tag{4}$$

where $score(p, d, q)$ is a function expressing the importance of d with respect to query q, and in relation to p; the function is calculated according to a selected ranking algorithm;
– the set of persons $P(q)$ is sorted in descending order by $Pscore(p, q)$, and a list of top n persons is presented to the user.

In general, the task of ranking researchers is a multi-criteria decision problem. Below we present a ranking algorithm, which takes into account the impact of research, measured by the impact of the publications. It can be roughly presented as follows:[9]

$$score(p, d, q) = rel(d, q) \times (sif(d) + 1) \times role(p, d) \tag{5}$$

where:

– $rel(d, q)$ is a measure of relevance of d with respect to query q; here we rely on the Lucene relevance score, which uses the cosine measure, with boosted values for the fields resulting from the semantic enrichment procedures (Section 5);
– $sif(d)$ is a scientific impact factor of d; it is a linear combination of the impact factor of the journal (for the journal papers) and its citations.

7 Conclusions and future work

The last decade have shown an increased interest among universities in systems concerning research data management and access to publicly funded research data. Both Internet access, and the development of AI methods have resulted in the demand for the building of academic digital libraries, which are not just document-centric repositories, but advanced knowledge bases, equipped with sophisticated functionalities. Bearing this in mind, we have implemented the Ω-Ψ^R platform as the university knowledge base, and installed it at Warsaw University of Technology.

In this paper we presented the AI methods that have been applied within the university knowledge base platform. In particular, we have presented a novel approach to data acquisition and semantic enrichment of the acquired data. Following this, we have presented two important functionalities of the Ω-Ψ^R platform, namely profiling researcher expertise and searching for experts.

In the future we plan to continue research on the issues discussed in the paper. Firstly, we plan to develop unsupervised web harvesting methods, data acquisition and integration. Some web mining tools aimed at discovering knowledge about journals and conferences are already in progress. Secondly, we will continue working on improving the quality of information retrieval, expert profiling and searching for experts. The already built repository of scientific publications, mostly in English, is quite heterogeneous in terms of the covered research areas, and, as such, it provides a good testbed for research on semantic cross-lingual searches, which gives rise to a more symmetric retrieval for English and Polish, i.e., giving similar results for queries regardless of language. Some work in this direction has already begun (Krajewski et al. 2014).

[9]This algorithm causes publications with higher values of tf-idf for the keywords used in q to be scored higher, moreover the journal impact factor and number of citations increase the ranking.

Acknowledgments The authors would like to thank two anonymous referees for their valuable and constructive comments, which have helped us to improve the quality of this paper.

References

Aberer, K., Boyarsky, A., Cudré-Mauroux, P., Demartini, G., & Ruchayskiy, O. (2011). Sciencewise: A web-based interactive semantic platform for scientific collaboration. In *10th International Semantic Web Conference (ISWC 2011-Demo), Bonn, Germany*.

Alani, H., Kim, S., Millard, D., Weal, M., Hall, W., Lewis, P., & Shadbolt, N. (2003). Automatic ontology-based knowledge extraction from web documents. *IEEE Intell. Syst.*, *18*(1), 14–21.

Arasu, A., & Garcia-Molina, H. (2003). Extracting structured data from web pages. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, ACM, New York, NY, USA, SIGMOD '03* (pp. 337–348).

Berman, F. (2008). Got data?: a guide to data preservation in the information age. *Commun. ACM*, *51*(12), 50–56.

Chang, C.H., Hsu, C.N., & Lui, S.C. (2003). Automatic information extraction from semi-structured web pages by pattern discovery. *Decis. Support. Syst.*, *35*(1), 129–147.

Crescenzi, V., Mecca, G., & Merialdo, P. (2001). Roadrunner: Towards automatic data extraction from large web sites. In *Proceedings of the 27th International Conference on Very Large Data Bases, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, VLDB '01* (pp. 109–118).

Davis, P.M., & Connolly, M.J.L. (2007). Institutional repositories: Evaluating the reasons for non-use of cornell university's installation of DSpace. *D-lib Magazine*, *13*(3), 2.

Deng, H., King, I., & Lyu, M.R. (2008). Formal models for expert finding on dblp bibliography data. In *ICDM'08 Eighth IEEE International Conference on Data Mining, 2008* (pp. 163–172): IEEE.

Fernandez, P. (2011). Zotero: information management software 2.0. *Library Hi Tech News*, *28*(4), 5–7.

Han, H., Giles, C.L., Manavoglu, E., Zha, H., Zhang, Z., & Fox, E.A. (2003). Automatic document metadata extraction using support vector machines. In *2003 Joint Conference on Digital Libraries, 2003. Proceedings* (pp. 37–48): IEEE.

Hazan, R., & Andruszkiewicz, P. (2013). Home pages identification and information extraction in researcher profiling. In Bembenik, R., Skonieczny, Ł., Rybinski, H., Kryszkiewicz, M., & Niezgódka, M. (Eds.) *Intelligent Tools for Building a Scientific Information Platform: Advanced Architectures and Solutions, Studies in Computational Intelligence* Vol. 467: Springer.

Kok, S., & Domingos, P. (2005). Learning the structure of markov logic networks. In *ICML* (pp. 441–448).

Koperwas, J., Skonieczny, Ł., Rybinski, H., & Struk, W. (2013). Development of a university knowledge base. In Bembenik, R., Skonieczny, Ł., Rybinski, H., Kryszkiewicz, M., & Niezgódka, M. (Eds.) *Intelligent Tools for Building a Scientific Information Platform: Advanced Architectures and Solutions, Studies in Computational Intelligence* Vol. 467: Springer.

Koperwas, J., Skonieczny, Ł., Kozłowski, M., Andruszkiewicz, P., Rybinski, H., & Struk, W. (2014a). AI platform for building university research knowledge base. In Andreasen, T., Christiansen, H., Cubero, J.C., & Raś, Z.W. (Eds.) *Foundations of Intelligent Systems, Lecture Notes in Computer Science*, (Vol. 8502 pp. 405–414): Springer.

Koperwas, J., Skonieczny, Ł., Kozłowski, M., Rybinski, H., & Struk, W. (2014b). University knowledge base: Two years of experience. In Bembenik, R., Skonieczny, Ł., Rybiński, H., Kryszkiewicz, M., & Niezgódka, M. (Eds.) *Intelligent Tools for Building a Scientific Information Platform: From Research to Implementation, Studies in Computational Intelligence*, Vol. 541: Springer International Publishing.

Kozlowski, M., & Rybinski, H. (2014). Sns: A novel word sense induction method. In *Rough Sets and Intelligent Systems Paradigms* (pp. 258–268).

Krafft, D.B., Cappadona, N.A., Caruso, J.C.R., Devare, M., Lowe, B.J., & Collaboration, V. (2010). Vivo: Enabling national networking of scientists.

Krajewski, R., Rybinski, H., & Kozlowski, M. (2014). A seed based method for dictionary translation. In *Foundations of Intelligent Systems, LNCS*, (Vol. 8502 pp. 415–424): Springer.

Lafferty, J.D., McCallum, A., & Pereira, F.C.N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, ICML '01* (pp. 282–289).

Leidig, J.P., & Fox, E.A. (2014). Intelligent digital libraries and tailored services. *J. Intell. Inf. Syst.*, *43*(1), 463–480.

Losiewicz, P., Oard, D., & Kostoff, R. (2000). Textual data mining to support science and technology management. *J. Intell. Inf. Syst.*, *15*(2), 99–119.

Mazurek, C., & Werla, M. (2005). Distributed services architecture in dLibra digital library framework. In *8th International Workshop of the DELOS Network of Excellence on Digital Libraries on Future Digital Library Management Systems*, Vol. 29.

Medelyan, O., Milne, D., Legg, C., & Witten, I.H. (2009). Mining meaning from wikipedia. *Int. J. Hum. Comput. Stud.*, *9*(67), 716–754.

Milne, D., & Witten, I.H. (2008). Learning to link with wikipedia. In *Proceedings of the 17th ACM conference on Information and knowledge management* (pp. 509–518).

Moreira, C., & Wichert, A. (2013). Finding academic experts on a multisensor approach using shannon's entropy. *Expert Systems with Applications*, *40*(14), 5740–5754.

Navigli, R., & Vannella, D. (2013). Semeval-2013 task 11: Word sense induction and disambiguation within an end-user application. In *Proc. of 7th International Workshop on Semantic Evaluation (SemEval), in the Second Joint Conference on Lexical and Computational Semantics*.

Omelczuk, A., & Andruszkiewicz, P. (2015). Agent-based web resource acquisition system for scientific knowledge base. In Onieva, E., Santos, I., Osaba, E., Quintiàn, H., & Corchado, E. (Eds.) *Hybrid Artificial Intelligent Systems - 10th International Conference, HAIS 2015, Bilbao, Spain, June 22-24, 2015, Proceedings, Springer, Lecture Notes in Computer Science*, (Vol. 9121 pp. 38–49).

Richardson, M., & Domingos, P. (2006). Markov logic networks. *Mach. Learn.*, *62*(1–2), 107–136.

Rose, S., Engel, D., Cramer, N., & Cowley, W. (2010). Automatic keyword extraction from individual documents. In *Text Mining* (pp. 1–20). Wiley.

Salo, D. (2008). Innkeeper at the Roach Motel. *Library Trends*, *57*, 98–123.

Tang, J., Yao, L., Zhang, D., & Zhang, J. (2010). A combination approach to web user profiling. *ACM Trans Knowl Discov Data*, *5*(1), 2:1–2:44.

Witten, I.H., Paynter, G.W., Frank, E., Gutwin, C., & Nevill-Manning, C.G. (1999). Kea: Practical automatic keyphrase extraction. In *Proceedings of the fourth ACM conference on Digital libraries* (pp. 254–255).

Wu, C.J., Chung, J.M., Lu, C.Y., Lee, H.M., & Ho, J.M. (2011). Using web-mining for academic measurement and scholar recommendation in expert finding system. In *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Volume 01, IEEE Computer Society* (pp. 288–291).

Wu, J., Williams, K., Chen, H.H., Khabsa, M., Caragea, C., Ororbia, A., Jordan, D., & Giles, C.L. (2014). Citeseerx: Ai in a digital library search engine. In *The Twenty-Sixth Annual Conference on Innovative Applications of Artificial Intelligence, IAAI*, Vol. 14.

Yu, K., Guan, G., & Zhou, M. (2005). Resume information extraction with cascaded hybrid model. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics* (pp. 499–506).

Zaiane, O.R., Chen, J., & Goebel, R. (2007). Dbconnect: mining research community on dblp data. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis, ACM* (pp. 74–81).

Zeng, J., Cheung, W.K., Li, C.h., & Liu, J. (2010). Coauthor network topic models with application to expert finding. In *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, (Vol. 1 pp. 366–373): IEEE.