Undefined 1 (2014) 1–25
IOS Press

Aemoo: Linked Data exploration based on Knowledge Patterns

Andrea Giovanni Nuzzolese [a,*], Valentina Presutti [a], Aldo Gangemi [a,c],
Silvio Peroni [a,b], and Paolo Ciancarini [a,b]
[a] *STLab-ISTC, Consiglio Nazionale delle Ricerche, Rome, Italy*
[b] *Department of Computer Science and Engineering, University of Bologna, Bologna, Italy*
[c] *LIPN, Université Paris 13, Sorbone Cité, UMR CNRS, Paris, France*

Abstract. This paper presents a novel approach to Linked Data exploration that uses Encyclopedic Knowledge Patterns (EKPs) as relevance criteria for selecting, organising, and visualising knowledge. EKP are discovered by mining the linking structure of Wikipedia and evaluated by means of a user-based study, which shows that they are cognitively sound as models for building entity summarisations. A tool named Aemoo is implemented which supports EKP-driven knowledge exploration and integrates data coming from heterogeneous resources, namely static and dynamic knowledge as well as text and Linked Data. Aemoo is evaluated by means of controlled, task-driven user experiments in order to assess its usability, and ability to provide relevant and serendipitous information as compared to two existing tools: Google and RelFinder.

Keywords: Exploratory search, Knowledge exploration, Visual Exploration, Knowledge Patterns, Analysis of Linked Data

1. Introduction

In the Semantic Web vision [6] agents were supposed to leverage the Web knowledge in order to help humans in solving knowledge-intensive tasks. Nowadays, Linked Data is feeding up the Semantic Web by publishing datasets by relying on URIs and RDF. However, it is still difficult to enable homogeneous and contextualised access to Web knowledge, for both humans and machines, because of the heterogeneity of Linked Data and the lack of relevance criteria (aka knowledge boundary [15]) for providing tailored data.

The heterogeneity consists of different data semantics, ontologies and vocabularies used in linked datasets. In fact, Linked Data is composed of datasets from different domains (e.g., life science, geographic, government). Moreover, some of them classify data according to a reference ontology (e.g., DBpedia) and others just provide access to raw RDF data. For example, if we consider the case of aggregating data from different linked datasets we would need a shared instensional meaning over the things described in these datasets in order to properly mash up facts about those things. The scenario is even more complex if we also take into account dynamic data coming from a variety of sources like social streams (e.g., Twitter) and news (e.g., Google News).

Instead, the knowledge boundary problem consists of identifying what data is really meaningful with respect to specific application tasks. Identifying meaningful data involves the need of establishing a clear relevance criteria to be applied as a filter on data. As an example, we can consider an application that leverages Linked Data to provide a summary on some topic. For example, if the topic of the summary is the philosopher Immanuel Kant, such an application should provide users with tailored information concerning, for example, facts about Kant's major works and thoughts and skip more peculiar facts.

Elsewhere [15] we introduced a vision for the Semantic Web based on *Knowledge Patterns* (KPs) as basic unit of mean for addressing both the heterogeneity and the knowledge boundary problems. More

*Corresponding author. E-mail:andrea.nuzzolese@istc.cnr.it.

recently we introduced the Encyclopedic Knowledge Patterns [27] (EKPs), a particular kind of KPs, that are empirically discovered by mining the linking structure of Wikipedia articles. EKPs provide knowledge unit for answering to the following competency question:

> What are the most relevant entity types $(T_1, \ldots, T_n)$ that are involved in the description of an entity of type C?

For example, the EKP for describing philosophers should include types such as book, university and writer because in Wikipedia a philosopher is typically linked to these entity types.

We assume that EKPs are cognitively sound because they emerge from the largest existing multi-domain knowledge source, collaboratively built by humans with an encyclopedic task in mind.

In this paper we want to exploit the EKPs for designing a system aimed at helping humans to address summarisation and discovery tasks. These tasks can be classified as *exploratory search* tasks as they involve different phases, i.e., look-up, learning and investigation [22], that characterise the strategies that humans adopt while exploring the Web. For example, consider a student who is asked to build a concept map about a topic for her homework. She starts by looking-up specific terms in a keyword-based search engine (e.g., Google), then she moves through search results and hyperlinks in order to better investigate the available information about the topic for finally learning new knowledge she can use to address her task.

Hypotheses. The work presented in this paper grounds on tow working hypotheses: (i) EKPs are cognitive sound and provide an unifying view and a relevance criterion for building entity-centric summaries and (ii) they can be exploited effectively for helping humans in exploratory search tasks.

Contribution. The contributions of this paper are:

- a method for extracting EKPs by mining the structure of Linked Data;
- a method for applying EKPs in order to provide entity summarisation and knowledge aggregation;
- an exploratory search system, called **Aemoo** [1], that uses EKPs as criteria for enriching an filtering data;
- an evaluation based on user-study of the cognitive soundness of the EKPs;
- an evaluation of Aemoo by means of controlled, task-driven user experiments in order to assess its usability, and ability to provide relevant and serendipitous information.

[1] `http://www.aemoo.org`

It is worth mentioning that there are state of the art systems that provides semantic mash-up and browsing capabilities, such as [43,18,19]. However, they mostly focus on presenting linked data coming from different sources, and visualising it in interfaces that mirror the linked data structure. Instead, Aemoo organises and filters the retrieved knowledge in order to show only relevant information to users, and providing the motivation of why a certain piece of information is included. The rest of the paper is organised as follows: Section 2 presents a method for extracting EKPs from the Wikipedia; Section 3 presents the solution and the system based on EKPs for knowledge exploration; Section 4 describes the experiments we conducted for evaluating the EKPs and the system; Section 5 discusses evaluation results, limits and possible solutions to improve the system; Section 6 presents the related work; finally, Section 7 summarises the contribution and illustrates possible future work.

2. Encyclopedic Knowledge Patterns

A general formal theory for Knowledge Patterns (KPs) still does not exist. Different independent theories have been developed so far and KPs have been proposed with different names and flavours across different research areas, such as linguistics [12], artificial intelligence [24,4], cognitive sciences [5,14] and more recently in the Semantic Web [15]. According to [15] it is possible to identify a shared meaning for KPs across these different theories, that can be informally summarized as “a structure that is used to organize our knowledge, as well as for interpreting, processing or anticipating information”.

In linguistics KPs were introduced as *frames* by Fillmore in 1968 [12] in his work about *case grammar*. In a case grammar, each verb selects a number of deep cases which form its *case frame*. A case frame describes important aspects of semantic valency, verbs, adjectives and nouns. Fillmore elaborated further the initial theory of case frames and in 1976 he introduced *frame semantics* [13]. According to the author a frame is

"... any system of concepts related in such a way that to understand any one of them you have to understand the whole structure in which it fits; when one of the things in such a structure is introduced into a text, or into a conversation, all of the others are automatically made available." [13]

A frame is comparable to a cognitive schema. It has a prototypical form than can be applied to a variety of concrete cases that fit this prototypical form. According to cognitive science theories [5] humans are able to recognize frames, to apply them several times, in what are called manifestations of a frame, and to learn new frames that became part of they background. Hence, frames (aka KPs) are cognitively relevant, since they are used by humans to successfully interact with their environment, when some information structuring is needed.

In computer science frames were introduced by Minsky, who recognized that frames convey both cognitive and computational value in representing and organizing knowledge. The notion of frame, aka knowledge pattern, was formalized by Minsky [24] as:

"... a remembered framework to be adapted to fit reality by changing details as necessary. A frame is a data-structure for representing a stereotyped situation, like being in a certain kind of living room, or going to a child's birthday party." [24]

In knowledge engineering the term Knowledge Pattern was used by Clark [9]. However, the notion of KP Clark introduces is slightly different from frames as introduced by Fillmore and Minsky. In fact, according to Clark, KPs are first order theories which provide a general schema able to provide terminological grounding and morphisms for enabling mappings among knowledge bases that use different terms for representing the same theory. Though Clark recognizes KPs as general templates denoting recurring theory schemata, his approach is similar to the use of theories and morphisms in the formal specification of software. Moreover, Clark's KPs lack the cognitive value as it is for frames. This makes difficult to use this formalization of KPs for representing contextual relevant knowledge.

More recently Knowledge Patterns have been reproposed in the context of the Semantic Web by Gangemi and Presutti in [15]. Their notion of KPs encompasses those proposed by Fillmore and Minsky and goes further envisioning KPs as the research objects of the Semantic Web as an empirical science.

In [27] we introduced the *Encyclopedic Knoeldge Patterns* (EKPs). EKPs were discovered by mining the structure of Wikipedia articles. EKPs are a special type of knowledge patterns: they express the core elements that are used for describing entities of a certain type with an encyclopedic task in mind. The cognitive soundness of EKPs is bound to an important working hypothesis about the process of knowledge construction realized by the Wikipedia crowds: each article is linked to other articles when *explaining* or *describing* the entity referred to by the article. DBpedia, accordingly with this hypothesis, has rdf-ized a) the subjects referred to by articles as *resources*, b) the wikilinks as relations between those resources, and c) the types of the resources as OWL classes. EKPs grounds on the assumption that wikilink relations in DBpedia, i.e. instances of the `dbpo:wikiPageWikiLink` property[2], convey a rich encyclopedic knowledge that can be formalized as knowledge patterns.

Informally, an EKP is a small ontology that contains a concept S and its relations to the most relevant concepts C_j that can be used to describe S. In [27] we defined a method for extracting EKP by analysing the structure of wikilinks. This method is based on three main steps, i.e.:

- gathering the Knowledge architecture of a dataset;
- EKP discovery;
- OWL2 formalization of EKPs.

These steps are detailed in next paragraphs.

Gathering the Knowledge architecture of a dataset. The goal of this step is to create a model aimed at giving an overview over the structure of wikilinks by exploiting the dataset of wikipedia page links available in DBpedia [3]; For this purpose, we used an OWL vocabulary, called *knowledge architecture*[4] [34], which allows to gather a landscape view on a dataset even with no prior knowledge of its vocabulary. More specifically, this vocabulary enables the representation of the typical paths that connect triples in a target dataset and

[2]Prefixes `dbpo:`, `dbpedia:`, and `ka:` stand for `http:dbpedia.org/ontology/`, `http:dbpedia.org/resource/` and `http://www.ontologydesignpatterns.org/ont/lod-analysis-path.owl`, respectively.

[3]The dataset is named `dbpedia_page_links_en`

[4]`http://www.ontologydesignpatterns.org/ont/lod-analysis-path.owl`

model them through the classes `ka:Path`[5], which defines a path of arbitrary length, and `ka:PathElement`, which defines a basic element of a path. Thus, we used such a vocabulary for identifying *type paths* in the dataset of wikilinks. A type path is as an extension of the notion of *property path*[6] that can be defined in the following way:

Definition 1 (Type Path) *A type path is a property path (limited to length 1 in this work, i.e. a triple pattern), whose occurrences have (i) the same `rdf:type` for their subject nodes, and (ii) the same `rdf:type` for their object nodes. It is denoted here as:*

$$P_{i,j} = [S_i, p, O_j]$$

where S_i is a subject type, p is a property, and O_j is an object type of a triple. In this work, we only extract type paths from RDF triples having `dbpo:wikiPageWikiLink` as predicate [27].

In practice, given a triple `s dbpo:wikiPageWikiLink o`, its type path is the following:

- S_i is set to the most specific type of s
- O_j is set to the most specific type of o
- p is set to the most general type of o

For example, the triple (cf. Fig. 1(a)):

```
dbpedia:Immanuel_Kant
    dbpo:wikiPageWikiLink
        dbpedia:Königsberg
```

would count as an occurrence of the path depicted in Fig. 1(b), which can be summarised as:

```
Path(Kant,Königsberg) =
  [dbpo:Philosopher,
   dbpo:Place,
   dbpo:City]
```

The rationale that allows us to the identify the previous path is the following:

- `dbpo:Philosopher` is the subject type because it is the most specific type of `dbpedia:Immanuel_Kant`, i.e., `dbpo:Philosopher` $\sqsubseteq$ `dbpo:Person` $\sqsubseteq$ `dbpo:Agent`;
- `dbpo:City` is the object type because it is the most specific type of `dbpedia:Königsberg`, i.e., `dbpo:City` $\sqsubseteq$ `dbpo:Settlement` $\sqsubseteq$ `dbpo:PopulatedPlace` $\sqsubseteq$ `dbpo:Place`
- `dbpo:Place` is the property of the path because it is the most general type of `dbpedia:Königsberg`.

EKP discovery. Once type paths have been collected and modelled according to the knowledge architecture vocabulary, it is possible to identify recurrent knowledge schemata, aka EKPs, emerging from the dataset. This is the goal of this step, which relies on a path-based definition of EKP that we proposed in [27], namely:

Definition 2 (Encyclopedic Knowledge Patterns) *Let S_i be a DBpedia type, O_j $(j = 1,..,n)$ a list of DBpedia types, $P_{i,j} = [S_i, p, O_j]$ and t a threshold value.*
Given the triples:

```
dbpedia:s
    dbpo:wikiPediaWikiLink dbpedia:o;
dbpedia:s rdf:type dbpo:S_i;
dbpedia:o rdf:type dbpo:O_j;
```

we state that $EKP(S_i)$ is a set of paths, such that

$$P_{i,j} \in EKP(S_i) \iff pathPopularity(P_{i,j}, S_i) \geq t$$

Where:

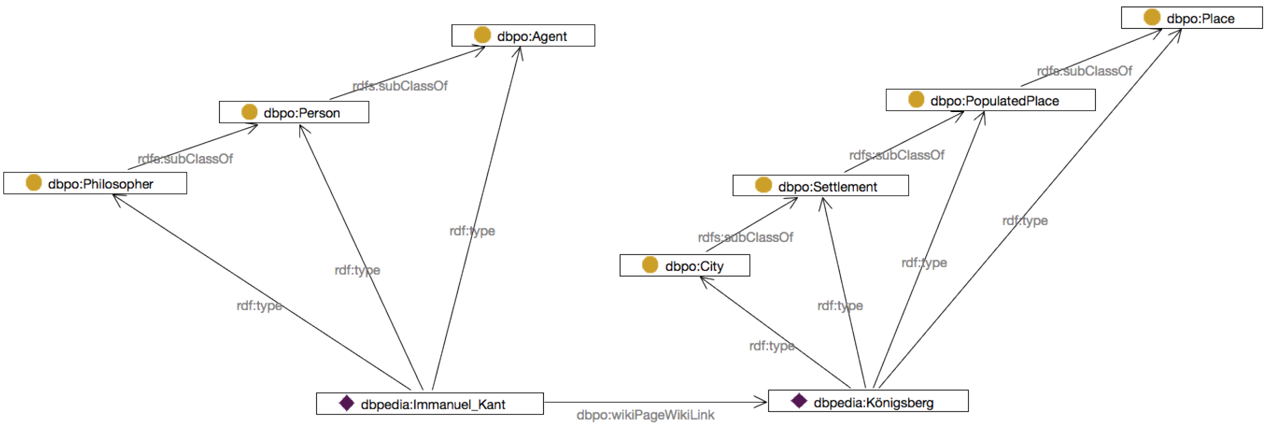
- $pathPopularity(P_{i,j}, S_i)$ is the ratio of how many distinct resources of a certain type participate as subject in a path to the total number of resources of that type. Intuitively, it indicates the popularity of a path for a certain subject type within a dataset. The list of all indicators used is reported in Table 1;
- t is a threshold value [27].

In order to decide a value for t we built a prototypical ranking of the $pathPopularity(P_{i,j}, S_i)$ scores, called $pathPopularity_{DBpedia}$. Then, we computed a K-Means clustering on $pathPopularity_{DBpedia}$ scores to hypothesise a threshold value for t. The clustering generated one large cluster (85% of the 40 ranks) with $pathPopularity_{DBpedia}$ scores below 18.18% and three small clusters with $pathPopularity_{DBpedia}$ scores above 22.67%. Hence, we set $t = 18.18\%$.

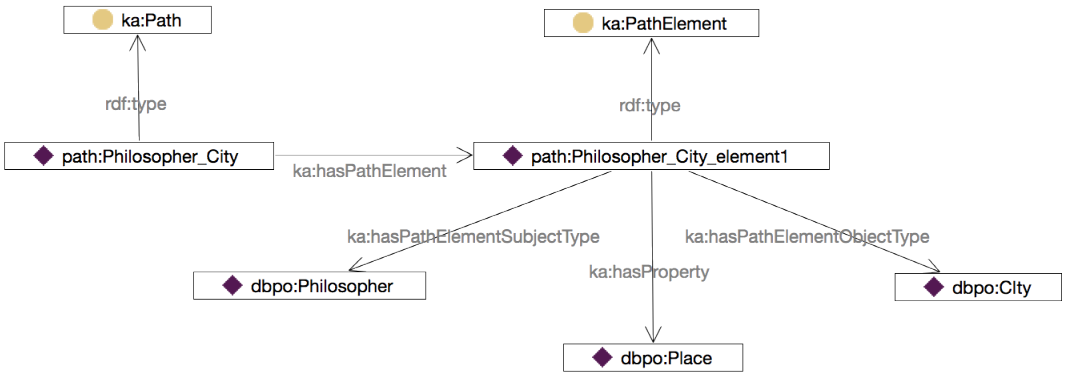
OWL2 formalization of EKPs. Given an $EKP(S_i) = [S_i, p_1, O_1], \ldots, [S_i, p_n, O_n]$, its formalisation in OWL2 is obtained by applying the following refactoring procedure to the dataset resulting from the previous steps:

[5] The prefix `ka:` stands for the namespace `<http://www.ontologydesignpatterns.org/ont/lod-analysis-path.owl#>`

[6] In SPARQL1.1 (http://www.w3.org/TR/sparql11-property-paths/) property paths can have length n, given by their route through the RDF graph.



(a) RDF graph extracted from DBpedia representing the triple `dbpedia:Immanuel_Kant dbpo:wikiPageWikiLink dbpedia:Königsberg` with their types and associated taxonomies.



(b) Type path identified from the triple `dbpedia:Immanuel_Kant dbpo:wikiPageWikiLink dbpedia:Königsberg` and represented according to the *knowledge architecture vocabulary*.

Fig. 1. Example of generation of the knowledge architecture for the wikilink dataset.

Indicator	**Description**
$nRes(C)$	number of resources typed with a certain class C, $\mid \{r_i \texttt{ rdf:type } C\} \mid$
$nSubjectRes(P_{i,j})$	number of distinct resources that participate in a path as subjects, $\mid \{(s_i \texttt{ rdf:type } S_i) \in P_{i,j} = [S_i, p, O_j]\} \mid$
$pathPopularity(P_{i,j}, S_i)$	The ratio of how many distinct resources of a certain type participate as subject in a path to the total number of resources of that type. Intuitively, it indicates the popularity of a path for a certain subject type, $nSubjectRes(P_{i,j} = [S_i, p, O_j])$ divided by $nRes(S_i)$
$nPathOcc(P_{i,j})$	number of occurrences of a path $P_{i,j} = [S_i, p, O_j]$
$nPath(S_i)$	number of distinct paths having a same subject type S_i, e.g. the number of paths having `dbpo:TennisPlayer` as subject type
$AvPathOcc(S_i)$	sum of all $nPathOcc(P_{i,j})$ having a subject type S_i divided by $nPath(S_i)$ e.g. the avarage number of occurrences of paths having `dbpo:Philosopher` as subject type

Table 1

Indicators used for empirical analysis of wikilink paths [27].

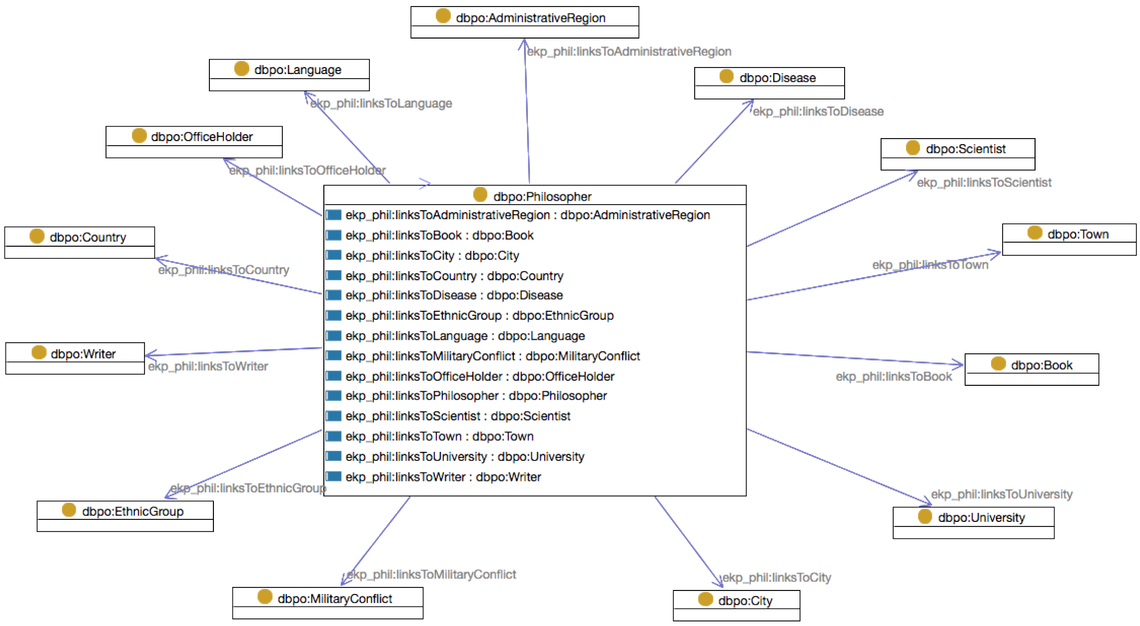


Fig. 2. UML class diagram for the EKP `ekp:Philosopher`. Blue arrows among classes represent universal restrictions.

- the name of the OWL file is `ekp:`[7] followed by the local name of S, e.g., `ekp:Philosopher.owl`. Below we refer to the namespace of a specific EKP through the generic prefix `ekpS:`;
- S_i and O_j $j = 1, \ldots, n$ are refactored as `owl:Class` entities (they keep their original URI);
- p keeps its original URI and is refactored as `owl:ObjectProperty` entities;
- for each O_j we create a sub-property of p, $ekpS$: q whose local name is composed of the string "linksTo" and the same local name as O_j; e.g. ekpS:linksToCity.
- for each $ekpS:q_j$ we add an `owl:allVauesFrom` restriction to S_i on $ekpS:O_j$, with range O_j.

For example, if $Path_{Kant,Knigsberg}$ (cf. Figure 1(b)) is part of an EKP, it gets formalized as follows [8]:

```
...
Class: dbpo:Philosopher
   SubClassOf:
       ekpS:linksToCity
           only dbpo:City
Class: dbpo:City
```

```
ObjectProperty: ekpS:linksToCity
   SubPropertyOf: dbpo:Place
   Domain: dbpo:Philosopher
   Range: dbpo:City
...
```

The UML class diagram of the EKP for `dbpo:Philosopher` formalised as an OWL 2 ontology is depicted in Fig. 2.

According to this method we generated 231 EKPs out of 272 DBPO classes [9] and published them into the ODP repository [10]. More details on the extraction of EKPs can be found in [27].

3. Encyclopedic Knowledge Patterns as relevance criteria for exploratory search

Aemoo is an exploratory search system that implements relevance strategies based on EKP for supporting exploratory search tasks. Additionally, EKPs are used by Aemoo as a unifying view for aggregating knowledge from static (i.e., DBpedia and Wikipedia) as well as dynamic (i.e., Twitter and Google News) sources. A preliminary version of Aemoo was presented in [28].

[7] The prefix `ekp:` stands for the namespace `http://www.ontologydesignpatterns.org/ekp/`.

[8] Prefix `ekptp` stands for `<http://www.ontologydesignpatterns.org/ekp/TennisPlayer.owl\#>`

[9] As at DBpedia version 3.7.

[10] `http://ontologydesignpatterns.org/aemoo/ekp/`

We assume that linking entities is a typical action performed by humans on the Web which reflects the way they organise their knowledge. As EKPs reflect the most frequent links between entity types, our hypothesis is that they can be used for selecting the most relevant entities to be included in an entity-centric summary that can support users in knowledge exploration.

In fact, Aemoo builds entity-centric summaries by applying EKPs as lenses over data, which makes it especially novel. In this way, Aemoo performs both *enrichment* and *filtering* of information. Enrichment and filtering are the two actions that Aemoo performs in order to address the knowledge heterogeneity and boundary problems. In fact, users are guided through their navigation: instead of being presented with a bunch of triples or a big unorganised graph they navigate through *units of knowledge* and move from one to the other without loosing the overview of an entity. An EKP determines a topic context according to its entity type. All relations between resources that emerge from the selected EKP are used as the basis for (i) selecting the information to be aggregated and (ii) visualising it in a *concept map* fashion. We discuss the first point in next section and the visualisation in Section 3.2.

3.1. Knowledge enrichment and filtering

Knowledge enrichment and filtering is achieved by Aemoo by applying a method composed of the following steps:

- identity resolution of a topic (provided by means of user queries) with respect to Linked Data entities;
- selection of an EKP corresponding to an entity type;
- filtering and enrichment of static data according to a selected EKP;
- filtering and enrichment of dynamic data;
- aggregation of peculiar knowledge;

In next paragraphs we detail these steps.

Identity resolution. Identity resolution is performed in two alternative ways:

- with a semi-automatic approach that leverages a DBpedia index based on the Entityhub of Apache Stanbol[11]. The Entityhub is a component of Stanbol that relies on Apache Solr[12] for building a customised entity-centric index for a linked dataset. This approach is semi-automatic because the system returns a list of possible entities that match the topic provided by an user according to their `rdfs:label`. Hence, the final entity resolution is up to the user;
- with a completely automatic approach based on a Named Entity Resolution (NER) system. Aemoo uses the Enhancer component of Apache Stanbol as NER system. In this case the identity of the topic provided by an user is resolved with respect to the DBpedia entity with the highest NER confidence.

EKP selection. An EKP provides an effective unit of knowledge for answering to questions like: "What are the most relevant entity types $(T_1, \ldots, T_n)$ that are involved in the description of an entity of type C? ". Thus, given a certain entity, Aemoo needs to (i) identify its type and (ii) select its corresponding EKP. For this purpose, given a subject entity, Aemoo uses an index, which provides associations between EKPs and DBPO types. The index was built during the extraction of EKPs from Wikipedia. The procedure applied to select an EKP by using such an index is the following:

- selection of the most specific DBPO type for a subject entity. This allows to avoid multi-typing and to be compliant with the method used for generating type paths (cf. Section 2);
- looking up the index in order to fetch the EKP associated with the DBPO type coming from the previous step. If no association is available, Aemoo traverses the DBPO taxonomy of superclasses iteratively until an association is found. If no association is found again, then a policy is applied. The policy consists of selecting the EKP for `owl:Thing`, i.e., `ekp:Thing`[13], that was gathered by analysing wikilinks having as subjects those entities typed only as `owl:Thing`. It is important to remark that many entities might be typed with classes defined in other ontologies (e.g., YAGO [39]), but we are only interested in DBPO types. Hence, if no DBPO type is available for an entity we select `ekp:Thing` as its EKP.

As an example, the type `dbpo:Philosopher` would be selected for the entity `dbpedia:Immanuel_Kant`

[11] `http://stanbol.apache.org`
[12] http://lucene.apache.org/solr/

[13] `http://ontologydesignpatterns.org/ekp/owl/Thing.owl`

and the EKP `ekp:Philosopher`[14] fetched from the index accordingly. Fig. 2 shows the UML class diagram for this EKP in which it is possible to see what are the typical relations emerging from wikilinks among entites typed as `dbpo:Philosopher` and other entity types.

Filtering and enrichment of static data. Aemoo applies EKPs as lenses over data for building contextualised views for organising and filtering the knowledge to be presented. This is the result of using EKPs as building blocks for automatically derive SPARQL constructs to query DBpedia. It is worth remarking that if an EKP changes then the results returned by Aemoo change accordingly. This means that the filtering and the organisation of data is completely pattern based. For example, the following SPARQL query is generated by using the `ekp:Philosopher`

```
1.  CONSTRUCT {
2     ...
3.    ?entity a ?type .
4.    dbpedia:Immanuel_Kant ?ekp_property ?entity
5.    ...
6.  }
7.  WHERE{
8.    GRAPH <dbpedia_page_links_en> {
9.      dbpedia:Immanuel_Kant
10.       dbpo:wikiPageWikiLink ?entity .
11.   }
12.   GRAPH <dbpedia_instance_types_en> {
13.     ?entity a ?type .
14.   }
15.   GRAPH <ekp_philosopher> {
16      ...
17.     ?ekp_property
18.       rdfs:domain dbpo:Philosopher .
19.     ?ekp_property
20.       rdfs:range ?type
21        ...
22.   }
23. }
```

Where, `dbpedia_page_links_en`, `dbpedia_instance_types_en` and `ekp_philosopher` identify three different datasets in our SPARQL endpoint[15], i.e., the DBpedia dataset of Wikipedia links, the DBpedia dataset of instance types and the dataset identifying the `ekp:Philosopher` respectively. In the query above the rationale for the knowledge boundary (that allows to filter only the wikilinks that are relevant with respect to the EKP) is the following:

– the subject entity is associated with its page links (i.e., the triple pattern `dbpedia:Immanuel_Kant dbpo:wikiPageWikiLink ?entity` at lines 9 and 10);
– each linked entity is bound to its type (i.e., the triple pattern `?entity a ?type` at line 13);
– the entities participating in a wikilink are filtered with respect to their types accordingly to the EKP. Namely, by using the notion defined for type paths (cd. Definion 1), we have:
  * $S_i =$`dbpo:Philosopher`;
  * $q =$`?ekp_property`;
  * $O_j =$`?type`;

  where, `?type` is a variable bound to the object type and `?ekp_property` is a variable bound to the property defined locally to an EKP for representing a wikilink (e.g., `ekpS:linksToCity`) obtained during the formalisation step of the method described in the previous section. Hence, the query filters only those wikilinks that are intensionally compliant with the `ekp:Philosopher`. This is done by constraining the wikilinks with the `rdfs:domain` and `rdfs:range` provided by their formalisation in the EKPs (code between lines 17 and 20).

Queries like the previous one produce entity-centric RDF models that allow Aemoo to identify the entities that are relevant to include in a description of a certain subject entity. For example, consider `dbpedia:Immanuel_Kant` as subject entity, then its entity-centric model is the following:

```
dbpedia:Immanuel_Kant a dbpo:Philosopher ;
    rdfs:label "Immanuel Kant"@en ;
    ekpS:linksToCity
        dbpo:Königsberg ;
    ...
    ekpS:linksToLanguage
        dbpo:Latin ;
    ...
    ekpS:linksToBook
        dbpo:Critique_of_Pure_Reason .

dbpo:Königsberg a dbpo:City ;
    rdfs:label "Königsberg"@en .

dbpo:Latin a dbpo:Language ;
    rdfs:label "Latin"@en .

dbpo:Critique_of_Pure_Reason a dbpo:Book ;
    rdfs:label "Critique of Pure Reason"@en .
```

Additionally, Aemoo enriches the previous model by also identifying possible semantic relations that can explain a specific link type. These relations are not ex-

[14] http://ontologydesignpatterns.org/ekp/owl/Philosopher.owl
[15] http://wit.istc.cnr.it:8894/sparql

haustive as they are obtained by only looking at their frequency for linking entities in DBpedia having the same type of the subject and the object entity. These semantic relations are added to the previous RDF model by means of skos:relatedMatch axioms. For example:

```
ekp_phil:linksToCity a owl:ObjectProperty ;
    skos:relatedMatch
        dbpo:birtPlace, dbpprop:placeOfBirth .
```

These two triples provide two possible matches found on DBpedia for ekpS:linksToCity, namely: dbpo:birtPlace and dbpprop:placeOfBirth

After having collected Linked Data from DBpedia, the system enrich it by aggregating textual information from Wikipedia that provides linguistic evidences of the relations between a subject entity and the entities just filtered. Intuitively, these evidences are the text windows (having maximum size of 50 words) surrounding wikilinks in the Wikipedia article corresponding to a subject entity. They are fetched only for those wikilinks filtered during the previous step and added to an entity-centric model by means of OWL2 triple annotations. For example, consider the triple

```
dbpedia:Immanuel_Kant
    ekpS:linksToCity
        dbpedia:Königsberg
```

Its annotation would be the following:

```
 [] a owl:Axiom ;
    owl:annotatedSource
        dbpedia:Immanuel_Kant ;
    owl:annotatedProperty
        ekpS:linksToCity ;
    owl:annotatedTarget
        dbpedia:Königsberg ;
    grounding:hasLinguisticEvidence:
        aemoo:wiki-sentence .

aemoo:wiki-sentence a doco:Sentence ;
    frbr:partOf wikipedia:Immanuel_Kant
    c4o:hasContent
        "Immanuel Kant was born in 1724
        in Königsberg, Prussia (since 1946
        the city of Kaliningrad,
        Kaliningrad Oblast, Russia), as
        the fourth of nine children."@en .

wikipedia:Immanuel_Kant a fabio:WikipediaEntry .
```

Where the triple is annotated with the property grounding:hasLinguisticEvidence[16] and the linguistic evidence is identified by aemoo:wiki-sentence[17]. The linguistic evidence is (i) typed as a doco:Sentence[18], is (ii) declared to be extracted from Wikipedia with the property frbr:partOf[19] and (iii) associated with the text by means of the property c4o:hasContent[20] [10]

Filtering and enrichment of dynamic data. Besides static data source, Aemoo uses also two dynamic data sources (that can be easily extended): Twitter and Google News. In fact, the entity-centric model built by Aemoo can be extended in order to include (i) the current stream of Twitter messages and (ii) available articles provided by Google News. Tweets and news are aggregated by means the identification of entities that co-occur with mentions of the subject. For example, consider the following tweet:

> "Lots of people love to read Kant here in Rome."

In such a tweet, the named entities "Kant" and "Rome" would be resolved by means of named-entity resolution as co-occurrences of dbpedia:Immanuel_Kant and dbpedia:Rome. These, in turn, are typed as ekp:Philosopher and ekp:City respectively. Thus, it is very simple to extend the entity-centric model by aggregating new (heterogeneous) data. According to the representation schema previously described, the entity-centric model would be extended in the following way:

```
 [] a owl:Axiom ;
    owl:annotatedSource
        dbpedia:Immanuel_Kant ;
    owl:annotatedProperty
        ekpS:linksToCity ;
    owl:annotatedTarget
        dbpedia:Rome ;
    grounding:hasLinguisticEvidence
        twitter:tweet_id .

twitter:tweet_id a fabio:Tweet ;
    c4o:hasContent
        "Lots of people love to
        read Kant here in Rome."@en .
```

Where the triple is annotated with the property grounding:hasLinguisticEvidence and the linguistic evidence, in this case, is identified by twitter:tweed_id[21]. The tweet is (i) typed as fabio:Tweet [32][22]

[16]The prefix grounding: identifies the ontology http://ontologydesignpatterns.org/cp/owl/grounding.owl.

[17]The namespace and are generated by Aemoo internally.

[18]The prefix doco: identifies the ontology http://purl.org/spar/doco.

[19]The prefix frbr: identifies the ontology http://purl.org/vocab/frbr/core#.

[20]The prefix c4o: identifies the ontology http://purl.org/spar/c4o [10].

[21]The tweet_id has to be replaced with an actual tweed identifier available from Twitter.

[22]The prefix fabio: identifies the ontology http://purl.org/spar/fabio [32].

and (ii) associated with the text by means of the property `c4o:hasContent`.

It is worth clarifying that not all co-occurrences are selected, but only those whose types are compliant with the intensional schema provided by an EKP. In fact, consider the tweet

"Just bought another Kant's book on Amazon."

The entity `dbpedia:Amazon` typed as `dbpo:Organisation` would not be added to the entity-centric model associated with `dbpedia:Immanuel_Kant`, because, according to the relevance criterion provided by the EKP `ekp:Philosopher`, no relevant relation exists between entities typed as `dbpo:Philosopher` and entities typed as `dbpo:Organisation`.

Aggregation of peculiar knowledge. EKPs are also used for identifying "curiosities" about a topic. Aemoo uses long-tail links for building a different entity-centric model, which includes peculiar facts instead of core knowledge. Long-tail links are those links that are normally taken out by an EKP lens because of their low $pathPopularity$ (cf. Definition 2). Entity-centric models based on curiosities are built in the same way of entity-centric models based on core facts. The main difference is that, for curiosity, the relevance criterion provided by an EKP is inverted. This means that less relevant relations (in terms of $pathPopularity$) between entities are taken into account instead of more relevant. For example, `dbpedia:Amazon` would be part of the curiosity-based entity-centric model associated with `dbpedia:Immanuel_Kant`. We assume that peculiar facts are useful in in order to better characterise a certain entity.

3.2. Knowledge visualisation

The entity-centric model built by Aemoo is visualised in a concept map fashion having:

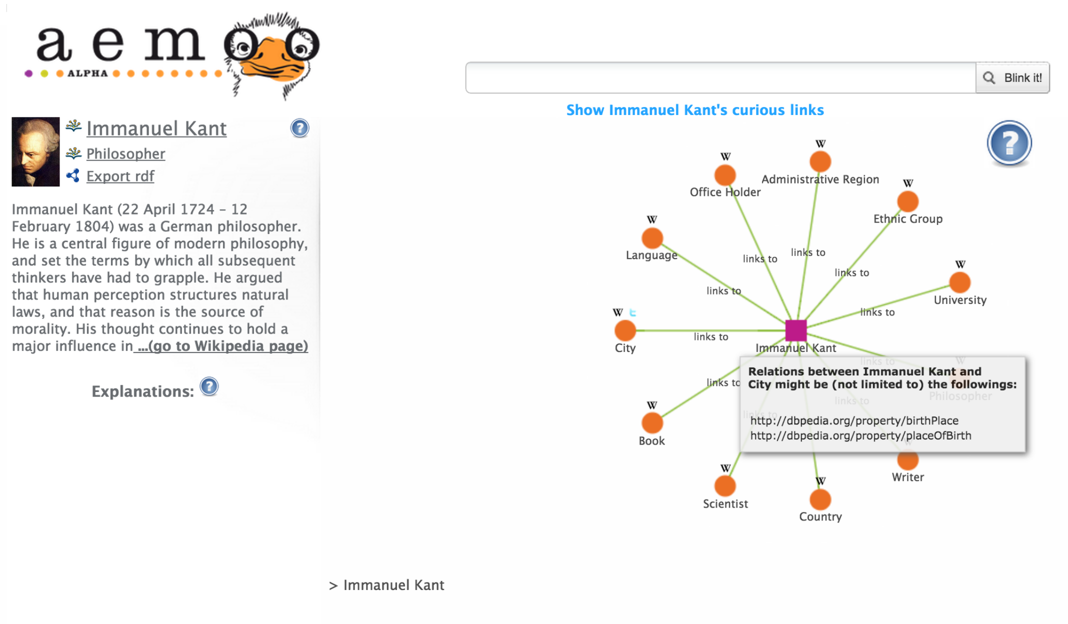
- the subject entity in the center represented as squared node;
- circular nodes around the subject entity that represent sets of resources of a certain type. Namely, they are the types of the resources linked to the subject entity that Aemoo aggregated and enriched from the various sources. We refer to them as *set nodes*. As previously described, such types are the ones that a user would intuitively expect to see in a summary description of an entity according to its type, as we described in Section 2.

Fig. 3(a) shows an example of a concept map relatively to Immanuel Kant as subject entity. Aemoo splits the interface into two parts: (i) in the center-right side it visualises the concept map and (ii) in the left side additional information about the subject entity. This additional information consists of an entity label with its thumbnail, DBPO type and abstract. An a user can click on an entity label or on its DBPO type in order to be redirected to their corresponding pages in DBpedia. Similarly, an user can open the Wikipedia page associated with the subject entity by clicking on the link "(go to Wikipedia page)" that Aemoo shows at the end of an abstract (cf. Fig. 3(a)).

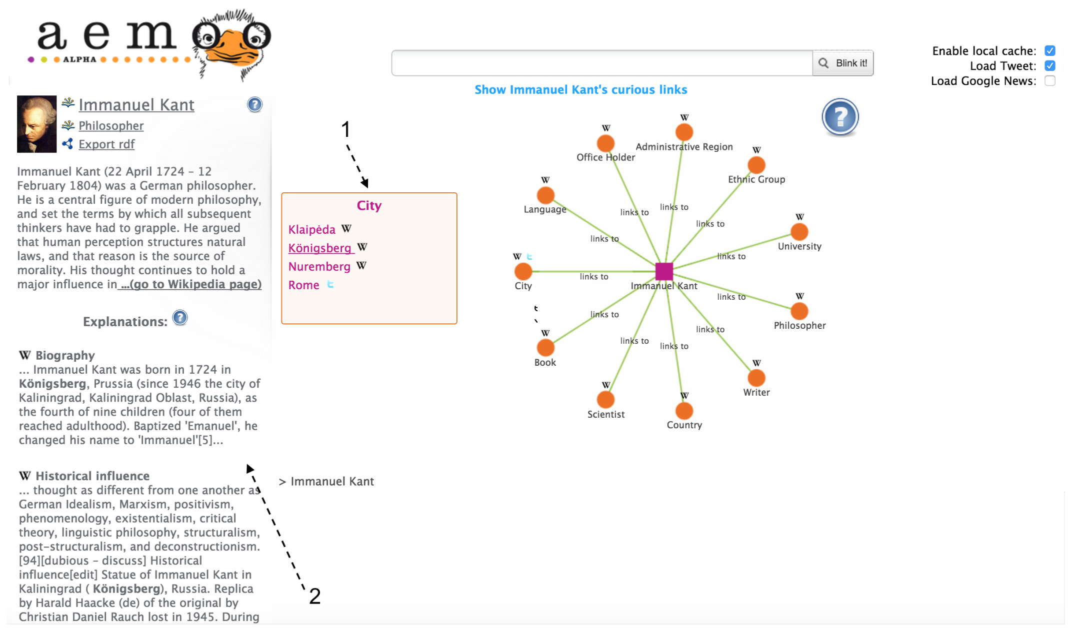
The concept map is completely interactive. In fact:

- a square box (cf. the arrow with id 1 in Fig. 3(b)) is visualised by mouse hovering on any set node. Such a box contains the list of the entities belonging to a set node. The list is paginated in order to present 10 entities per page. We refer to these boxes as entity boxes. Each entity in an entity box is depicted along with an icon that indicates the provenance (i.e., Wikipedia, Twitter or Google News). These icons are also shown for set nodes in order to summarise the provenance about their contained entities.
- a tooltip appears by mouse hovering on any arch connecting a subject entity to a set node. Such a tooltip shows a list of possible semantic relations that can explain that specific link type, according to their frequencies in DBpedia. The semantic relations are represented in an entity-centric model as `skos:relatedMatch` axioms. For example, Fig. 3(a) shows a tooltip with respect to the link between Immanuel Kant and the set node City. Such a tooltip provides as possible relations `dbpo:birthPlace` and `dbpprop:placeOfBirth`[23];
- a list of explanations appears in left-bottom of the interface (cf. the arrow with id 2 in Fig. 3(b)) by hovering on any entity in an entity box. These explanations come from the linguistic evidences represented as triple annotations in an entity-centric model, as previously described. Explanations can be used by humans for mining relations among entities provided by a concept map. For example, in Fig. 3(b) are shown all the linguistic evidences aimed at explaining the relations be-

[23]The prefix `dbpprop` stands for the namespace `http://dbpedia.org/property/`.



(a) Initial summary page for the query: *Immanuel Kant*.



(b) Exploring knowledge between *Immanuel Kant* and *Königsberg*. On the left side there is a list of explanation snippets that provide evidences of the relation with respect to the enabled sources.

Fig. 3. Aemoo user interface.

tween Immanuel Kant and Königsberg. An user may easily discover that Immanuel Kant was born in Königsberg and that there is a statue of Kant in such a city. Explanations have also associated icons, which indicate their provenance;

– any entity in an entity box is clickable. This enables browsing capabilities. In fact, an user can change the focus (i.e., the subject entity) of the concept map at any time by selecting with a click any possible entity in an entity box. When the

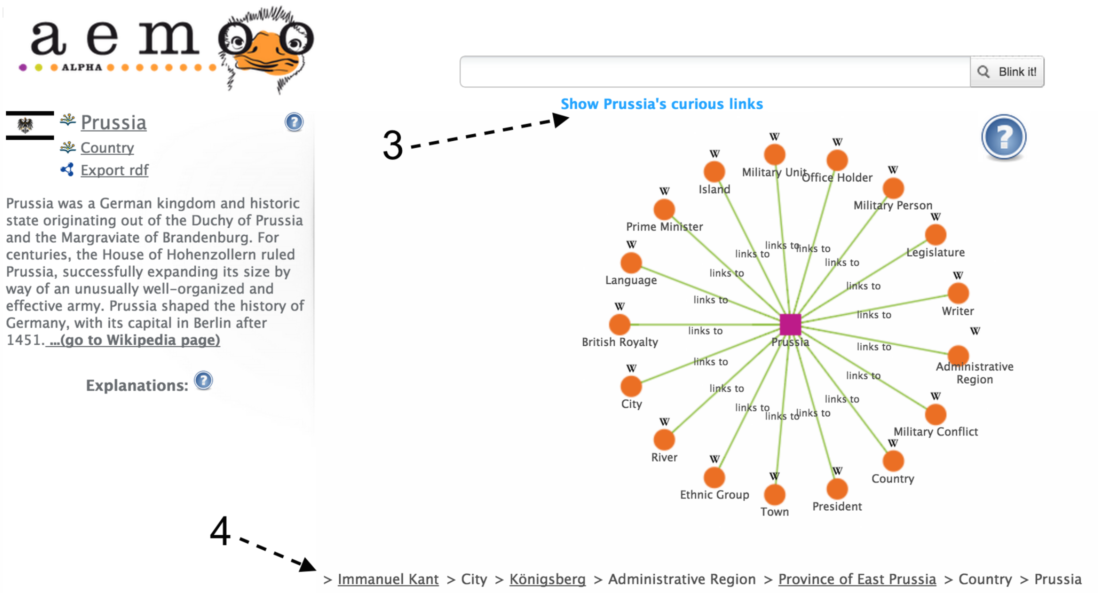


Fig. 4. Aemoo: breadcrumb

focus changes the concept map is rearranged according to the new subject entity and its type by applying the appropriate EKP. Fig. 4 shows the situation after some exploration steps with Prussia as subject entity. In center-bottom of the interface there is the exploratory history (cf. the arrow with id 4 in Fig. 4), named breadcrumb. The breadcrumb allows an user to retrace his exploratory steps at any time.

The sources to be used for populating the concept map can be chosen by users through a set of checkboxes that appears in the top-right corner of the interface (cf. Fig. 3(b)).

A blue link located in top-center of the interface under the search bar allows users to switch to the curiosities about a subject entity. When ckicking on this link the knowledge is again arranged in a concept map fashion, and enriched with news and tweets just as it happens for the previous summary, but this time the set nodes are selected with a different criterion: they are types of resources that are unusual to be included in the description of a country, hence possibly peculiar.

3.3. *Design and implementation details*

Aemoo is released as a web application: it consists of a server side component implemented as a Java-based REST service, and a client side component based on HTML and JavaScript.

The overview of the architecture of the server side, including also the components for EKP extraction, is depicted in Fig.5.

The architecture is designed by using the Component-based [16] and the REST [11] architectural styles.

On one hand, the Knowledge Pattern (KP) extractor is composed of the following components:

- KP Extraction Coordinator who takes care about the coordination of the overall extraction process;
- Property Path Identifier that is responsible for the identification of type paths (cf. Definition 1);
- Property Path Storage that manages the storage of identified paths;
- Property Path Analyzer that draws boundaries around paths in order to formalise KPs;
- KP Repository Manager who is responsible for the storage, indexing and fetching of KPs.

On the other, Aemoo is composed of the following components:

- Aemoo Coordinator who coordinates all the activities;
- Identity Resolver who is in charge of resolving an user query with respect to entity in Linked Data;
- KP Selector that selects an appropriate KP according to the entity identified;
- Knowledge Filter who takes care of applying a KP on row RDF data;

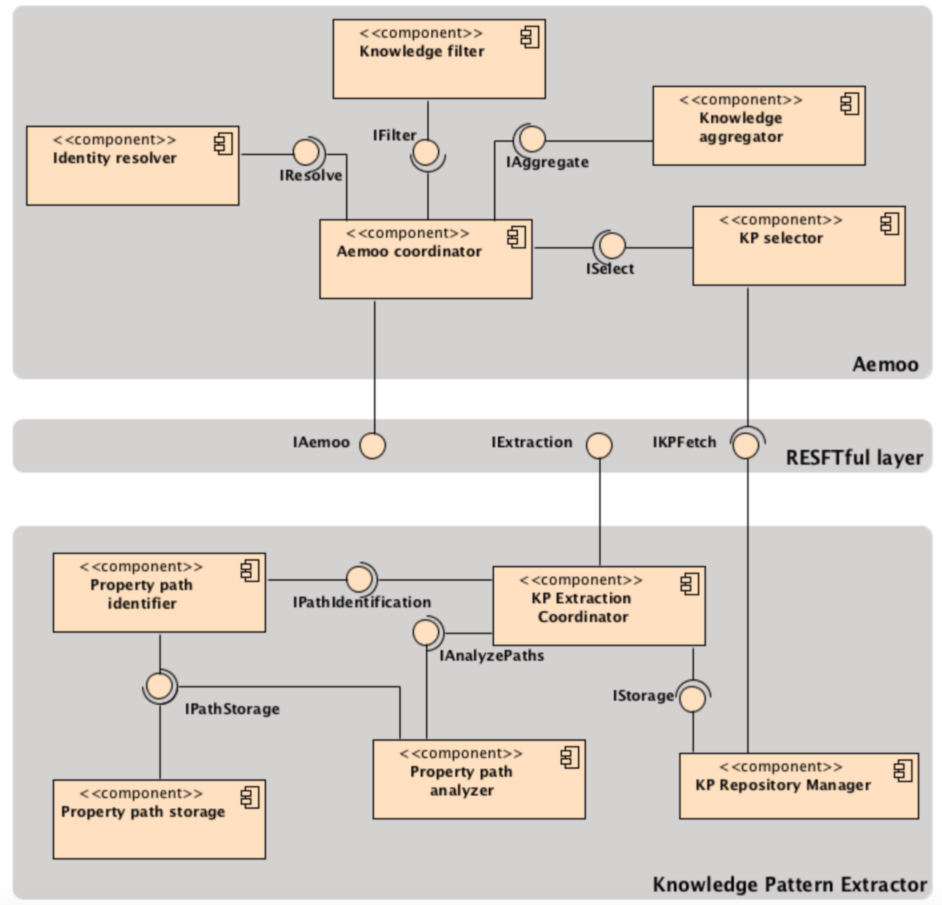


Fig. 5. Overview of the architecture of Aemoo and the EKP extractor.

– Knowledge Aggregator that aggregates knowledge form other sources with respect to the KP selected.

All components are implemented as Java OSGi [1, 2,3] bundles, components and services and some of them, i.e., KP Extraction Coordinator, KP Repository Manager, Aemoo Coordinator, provide RESTful interfaces.

The client side interacts with the other components via REST interfaces through AJAX. Additionally, it handles the graphical visualization of Aemoo through the JavaScript InfoVis Toolkit[24], a Javascript library that supports the creation of interactive data visualizations for the Web.

[24]http://thejit.org/

4. Evaluation

In Section 1 we hypothesised the (i) cognitive soundness of EKPs for providing intuitive entity-centric summaries. We also hypothesised that (ii) EKPs can be exploited effectively for helping humans in exploratory search tasks. In next section we describe the experimental setup used for demonstrating the first hypothesis, while in Section 4.2 we describe the experimental setup used for demonstrating the second one.

4.1. Experimental setup for evaluating EKPs

In order to demonstrate our first hypothesis we carried out an user-based study aimed at making emerge EKPs from human consensus in order to compare these EKPs with those empirically emerging from DBpedia. We asked to 17 subjects of different culture and language to indicate the most relevant types of things (object types) that could be used to describe a certain type of things (subject types).

DBPO class type	nRes(S)	$nPath(S_i)$	$AvPathOcc(S_i)$
Language	3,246	99	29.27
Philosopher	1,009	112	18.29
Writer	10,102	172	15.30
Ambassador	286	85	15.58
Legislature	453	83	25.11
Album	99,047	172	11.71
Radio Station	16,310	151	7.31
Administrative Region	31,386	185	11.30
Country	2,234	169	35.16
Insect	37,742	98	9.16
Disease	5,215	153	12.10
Aircraft	6,420	126	10.32

Table 2

DBPO classes used in the user-study and their related figures [27].

We provided subjects with a list of 12 subject types spanning different domains. These types are those presented in Table 2, where for each of them we also indicate the number of resources ($nRes(S)$), the number of paths they participate in as subject types ($nPath(S_i)$) and the average number of occurrences of their associated paths ($AvPathOcc(S_i)$). The information about $nRes(S)$, $nPath(S_i)$ and $AvPathOcc(S_i)$ was not given to subjects.

Then, we provided subjects with a list of object types for each subject type of the previous list of 12. This second list was obtained by selecting, for each subject type S_i, the object types O_j so that $pathPopularity(P_{i,j}, S_i) >= 18.18$, where 18.18 is the value we associated to our threshold t (cf. Section 2).

The subjects were asked to assign a 5-point relevance score, ranging from 1 (completely irrelevant) to 5 (very relevant), to each object type by taking into account to what extent it can be used to describe the target subject type.

Finally, we defined a mapping between $pathPopularity$ values (ranging from 0 to 100) and the 5-point scale rates that users provided. The mapping (cf. Table 3) was needed to perform the comparison between ranking values provided by humans and those of $pathPopularity_{DBpedia}$.

In Section 5.1 we provides the outcomes of this experimental setup.

4.2. Experimental setup for evaluating Aemoo

In order to demonstrate our second hypothesis (i.e., the EKPs are effective in supporting humans during

$pathPopularity_{DBpedia}$ interval	Relevance score
[18, 100]	5
[11, 18[	4
]2, 11[	3
]1, 2]	2
[0, 1]	1

Table 3

Mapping between $pathPopularity_{DBpedia}$ intervals and the relevance score scale [27].

exploratory search tasks) we carried out an user-based study whose aim is twofold:

– evaluate the system usability of Aemoo;
– provide evidence about the effectiveness of Aemoo to support humans in exploratory search tasks.

For this purpose, we defined three different tasks involving the different phases, i.e., look-up, learning and investigation [22], that characterise the strategies that humans adopt while exploring the Web. The three tasks were focused on:

– summarisation;
– finding of related entities;
– relation identification.

The first task was more exploratory and unbounded in terms of goals than the other two. Anyway, they were all designed in order to require to the subjects an effort in learning, investigation an look-up.

In next paragraphs we describe in details these tasks.

Task 1 - summarisation: the subjects were asked to identify all the elements and their relations that could be included in a summary about a certain topic, record them, and score them based on two criteria: relevance and unexpectedness. The scores had to be assigned on a 5-point scale ranging from 1 (minimum score of relevance/unexpectedness) to 5 (maximum score of relevance/unexpectedness). The topic we selected for the summary was “Alan Turing”. In order to provide such a summary, the subjects were asked to explore the information by starting from any possible concept they found useful and to record any pair of elements they found relevant for building the summary. Basically, these pairs had to be recorded in the form of *subject-object* plus a description about the relation occuring between the two elements. For example, having “Paris" as general topic, the information that Paris is located at the heart of the Île-de-France region would be registered as follows:

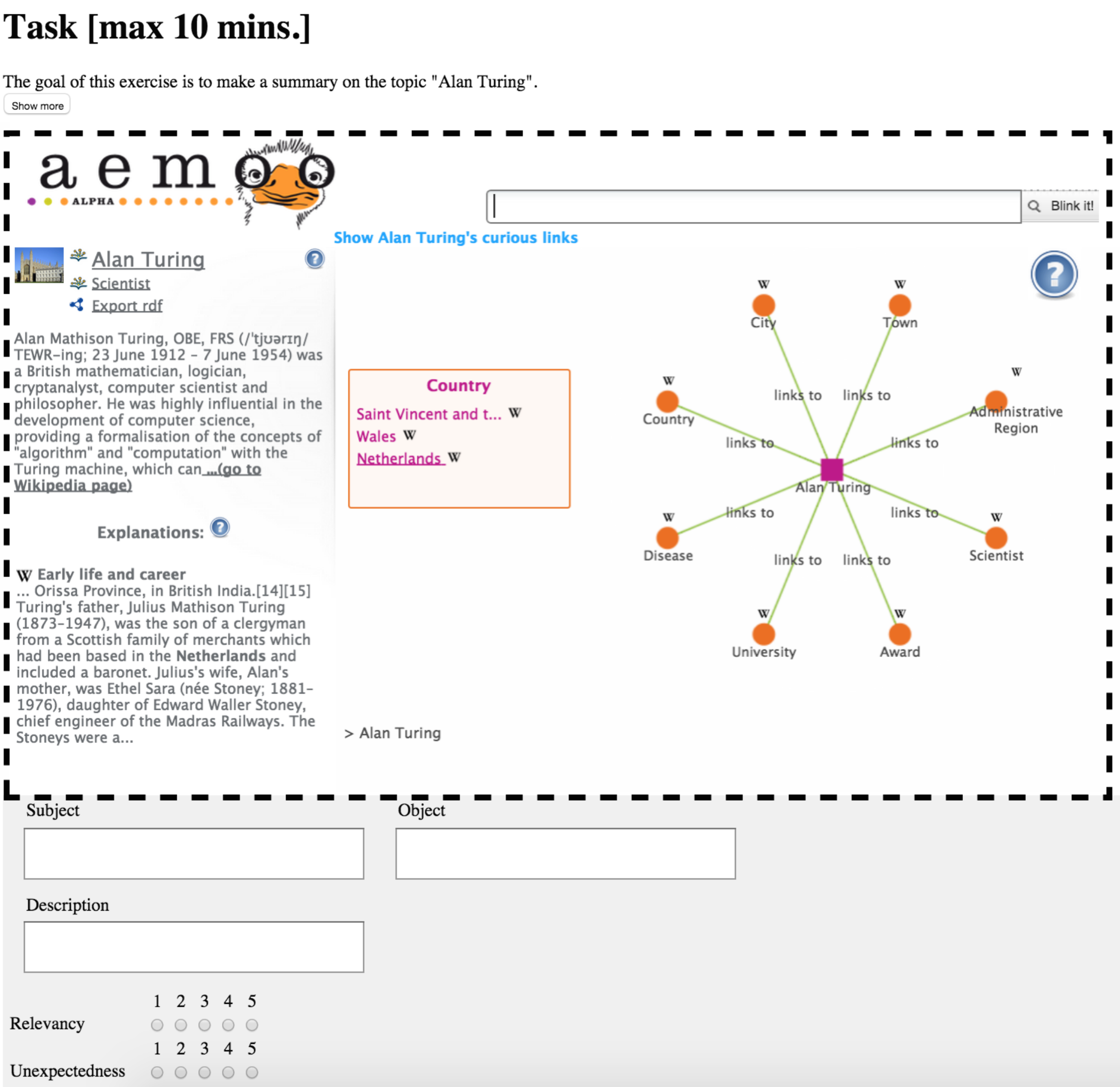


Fig. 6. The interface of the Web-based tool designed and implemented for carrying on the user-study.

- subject=*Paris*
- object=*Île-de-France*
- description=*Paris is located at the heart of the Île-de-France region*

Task 2 - finding of related entities: the subjects were asked to find as many objects of a certain type as possible related to a given topic. Namely, the question was: *What are the places related to "Snow White"?*. For addressing this task, the subjects had to list all the elements related to the topic according to the typing constraint. For example, having "Steve Jobs" as topic, the information about the companies related to Steve Jobs would be recorded as a list composed of *Apple*, *NeXT*, *Pixar*, etc.

Task 3 - relation finding: the subjects were asked to find one or more relations between two topics and to further provide explanations, i.e., evidence, about such relation. The question we asked to the subject was: *Why "Snow White" is related to "Charlize Theron?"*. Subjects had to report the list of relations they were able to find. For example, having "Steve Jobs" and "Apple" as subject and object topics respectively, the possible relations would be recorded as *CEO*, *co-founder*, *chairman*, etc.

The outcomes from all the three tasks were used for measuring the system usability. The outcomes from Task 1, about the relevance and unexpectedness scores, were used for measuring the ability of Aemoo to support humans by providing them with relevant results. We only used Task 1 for this purpose because it was designed to cope with a wider range of activities required by exploratory search. In fact, it less constrained and more open in terms of the goals to achieve.

The three tasks were proposed to 32 subjects, with different background and skills, divided into 5 groups. Each group performed the three tasks twice on two different systems, i.e., Aemoo and one between Google Search or RelFinder [18]. Subjects did the tasks twice in order to give us some feedback for possibly improving Aemoo. Instead, Google and RelFinder were introduced in the experiments as basis for comparison. We selected these two systems because: (i) Google is one of the most-used search engines on the World Wide Web and it is also used by humans for exploratory search, for example, by iteratively refining the query during a search; (ii) RelFinder is one of the most widespread tool in the Semantic Web community supporting exploratory search.

We designed the user study in order to have a balanced number of subjects starting the experiment by using one of the three tools as first. This means that the number of subjects that started the experiment by using Aemoo as firts tool is comparable with the number of subjects that started the experiment by using RelFinder or Google as first tool. This choice was aimed at breaking down a possible bias deriving from the fact that each subject was asked to complete the same tasks by using two different systems sequentially. Namely, we wanted to prevent that the answers provided by a certain subject during the first iteration affected the answer provided by the same subject during her second iteration. Concerning the time constraint imposed during the user-study, it is worth reporting that the subjects had to complete each task in maximum 10 minutes.

In order to support subjects during the user-study we implemented a Web-based tool, namely AemooEval [25]. AemooEval was implemented according to the following design requirements:

- a subject group had to be associated with an evaluation session;
- an evaluation session had to be customised to allow the usage of two tools out of three;
- a subject had to use both tools available during a session;
- a subject had first to complete the three tasks designed by using one of the two tools, then she could proceed by repeating the same tasks by using the other tool;
- a subject could not skip a task before starting a new one;
- AemooEval had to automatically split the subjects into two balanced sub-groups. Then, AemooEval automatically had to provide the two sub-groups with two different tools to use as first system to evaluate;
- AemooEval had to present within the same interface: (i) the task to address; (ii) a description about how to complete the task; (iii) a running instance of the system to use for the evaluation; (iv) an input form for recording answers.
- AemooEval had to record the solutions proposed by a subject for a task along with the time needed for completing the task.

It is worth mentioning that, for the user-study, we constrained the search domain to Wikipedia by:

- properly configuring on our server an instance of RelFinder[26] to find relations only on DBpedia
- limiting Google's space search to the english version of Wikipedia only. This was performed by asking and supervising subjects to set the `site` keyword to the value `en.wikipedia.org` for querying Google.

The reason behind this choice is simple: Aemoo is designed to perform exploratory search on Wikipedia by applying EKPs as filtering lenses over data. Hence, we found fair to compare the systems on a same domain.

Figure 4.2 shows the user interface of AemooEval with respect to the execution of the first task, i.e., "The goal of this exercise is to make a summary on the topic *Alan Turing*". The "Show more" button allowed to display a noninvasive text snippet that could be shown or hidden in order to have a detailed description about how to complete the task in terms of what to do and to record. AemooEval put the system to use for the accomplishing the tasks inside the dotted frame, which is embedded into the interface by means of an HTML iframe tag. In the bottom of the Web interface (cf. Fig. 4.2) AemooEval provides an input form the subjects had to use in order to record their solutions to tasks.

[25] `http://wit.istc.cnr.it/sweng/cgi-bin/`

[26] This instance was presented to subjects by AemooEval.

The user-study was supervised by an administrator, who was in charge to support the subjects during the experiments. Additionally, the administrator provided subjects with:

– an introductory description of the user-study;
– a brief tutorial about how to use the Web-based tool for the evaluation according to the three tasks proposed;
– a brief tutorial on the systems that were objects of the experiments, i.e., Aemoo, RelFinder and Google.

Twice (one time per tool) at the end of the three tasks, the subjects were asked to answer to ten five-point Likert scale questions and to five free-text answering questions.

The five-point Likert scale questions are reported in Table 4 and were aimed at recording data to use for the usability analysis by means of the System Usability Scale (SUS) [7]. The SUS is a well-known metrics used for the perception of the usability of a system. It has the advantage of being technology independent and it is reliable even with a very small sample size [37]. Furthermore, it provides two-factor orthogonal structure, which can be used to score the scale on independent Usability and Learnability dimensions [37].

Nr.	Question
1	I think that I would like to use the system frequently
2	I found the system unnecessarily complex
3	I thought the system was easy to use
4	I think that I would need the support of a technical person to be able to use the system
5	I found the various functions in the system were well integrated
6	I thought there was too much inconsistency in the system
7	I would imagine that most people would learn to use the system very quickly
8	I found the system very awkward to use
9	I felt very confident using the system
10	I needed to learn a lot of things before I could get going with the system

Table 4

Five-point Likert scale questions.

The five free-text answering questions are reported in Table 5 and where aimed at receiving feedback (pros and cons) from the subjects to use to compute a qualitative analysis based on a Grounded theory [38]. Grounded theory is a method often used in Social Science to extract relevant concepts from unstructured corpora of natural language resources (e.g., texts, interviews, or questionnaires).

Nr.	Question
1	How effectively did the system support you in answering to the previous tasks?
2	What were the most useful features of the system that helped help you to perform your tasks?
3	What were the main weaknesses that the system exhibited in supporting your tasks?
4	Would you suggest any additional features that would have helped you to accomplish your tasks?
5	Did you (need to) open and read Wikipedia pages when using the system? If yes, please explain the motivation.

Table 5

Free-text answering questions.

5. Results and discussion

In next subsection we describe the main outcomes of the two experiments described in the previous section.

5.1. Cognitive soundness of EKPs

In order to use the EKPs emerging from humans for the comparison with those emerging empirically from DBpedia we firstly checked the inter-rater agreement among subjects by using the Kendall's coefficient of concordance (W). Kendall's W assesses the agreement among raters on a scale ranging from 0 (no agreement) to 1 (complete agreement). Additionally we computed the Cronbach's coefficient of internal consistency (α), which is used as an estimate of the reliability of a psychometric test for a sample of raters. Cronbach's α can be considered unacceptable for $\alpha < 0.5$ and excellent for $\alpha \geq 0.9$. In Table 6 we report results about the agreement and the reliability among the 17 subjects for the list of 12 subject types.

On average we obtained good concordance, with $W = 0.68$, and excellent internal consistency, with $\alpha = 0.93$.

We then computed the Spearman's rank correlation coefficient (ρ) between users's assigned scores, and $pathPopularity_{DBpedia}$ based scores. Spearman's ρ assesses the correlation on a scale ranging from -1 (no

DBPO class	Agreement	Reliability
Language	0.836	0.976
Philosopher	0.551	0.865
Writer	0.749	0.958
Ambassador	0.543	0.915
Legislature	0.612	0.888
Album	0.800	0.969
Radio Station	0.680	0.912
Administrative Region	0.692	0.946
Country	0.645	0.896
Insect	0.583	0.929
Disease	0.823	0.957
Aircraft	0.677	0.931

Table 6

Inter-rater agreement computed with Kendall's W and reliability test computed with Cronbach's alpha [27]

DBPO class	Correl. users / DBpedia
Language	0.893
Philosopher	0.661
Writer	0.748
Ambassador	0.655
Legislature	0.716
Album	0.871
Radio Station	0.772
Administrative Region	0.874
Country	0.665
Insect	0.624
Disease	0.824
Aircraft	0.664

Table 7

Spearman's rank correlation coefficient (ρ) between users's assigned score, and $pathPopularity_{DBpedia}$ based score [27].

agreement) to 1 (complete agreement). Table 7 shows Spearman's ρ values.

On average we obtained very good correlation, with $\rho = 0.75$.

Discussion The good inter-rater agreement and the high reliability suggests large consensus among subjects of our experiment. The multicultural connotation of subjects is an added value with respect to these outcomes. Additionally, the high correlation between users' and EKP rankings suggests that: (i) the value for t we hypothesised (cf. Section 2) is a stable criterion for boundary creation; (ii) EKPs provide good summaries according to an entity type. Hence, the hypothesis of cognitive soundness of the EKPs is supported by these findings.

5.2. Usability and effectiveness of Aemoo

We classified all the solutions provided by subject in order to have a set of correct ones. Then we measured them as function of time. Fig. 7 reports the number of correct solutions per minute provided by subjects for each of the three tasks and on average. Results show that, on avarage, Aemoo performs better than RelFinder and Google Search, but, apparently, it is due to its efficiency in Task 2. In fact, Aemoo outperforms the other tools in the second task, while, in the first task, RelFinder performs slightly better than Aemoo and Google and, finally, in the third task Google performs slightly better than Aemoo and RelFinder.

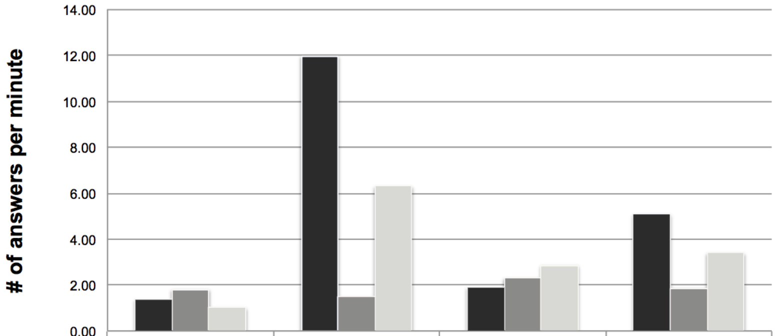
Additionally, we computed the SUS for understanding to what extent Aemoo is perceived by subjects useful and easy to learn.

Figure 8 depicts the comparison of the SUS results for the three tools in the cases they are used as first tool, as second tool and on average. SUS values are weighted on a scale between 0 and 100. Values between brackets represent standard deviations and they are also reported into the chart as vertical error bars in black.

We did not observed any relevant difference of SUS values between Aemoo and Google. This is also supported by the analysis of the p-value aimed at measuring the statical significance level. In fact, we recorded $p > 0.1$ between Aemoo and Google in terms of SUS values, which means that there is no presumption of statistical significance. The lack of statistical significance was observed in all the possible configurations (i.e., "Used ad first tool", "Used ad first tool" and "Average").

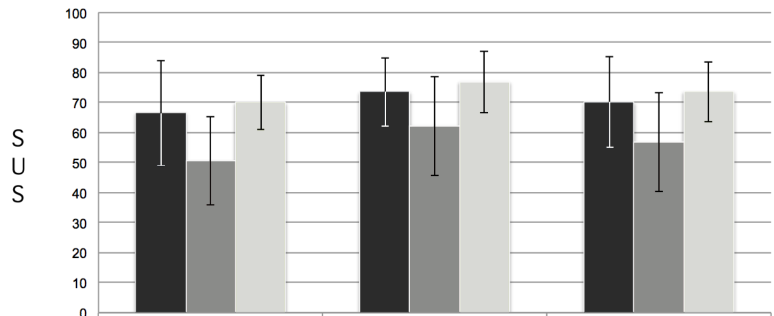
Instead, we observed a strong or good statistical significance in the difference of SUS values between Aemoo and RelFinder. In fact, when subjects used Aemoo as first tool we recorded $0.01 < p \leq 0.05$, which means strong significance. When subjects used Aemoo as second tool we recorded $0.05 < p \leq 0.1$, which means good significance. Considering the average we recorded $0.05 < p \leq 0.1$.

In addition to the main SUS scale, we were also interested in examining the sub-scales of pure Usability and pure Learnability of the three systems that have been proposed some years ago by [21]. Figure 9 shows values and the standard deviations for these two orthogonal structures of the SUS. Namely, Figure 9(a)) depicts Learnability scores and their related standard deviations and Figure 9(b) reports Usability scores and



	Task 1	Task 2	Task 3	Average
Aemoo	1.37	11.99	1.90	5.08
RelFinder	1.81	1.49	2.30	1.87
Google	1.01	6.31	2.86	3.39

Fig. 7. Number of correct answers per minute for each task and tool.



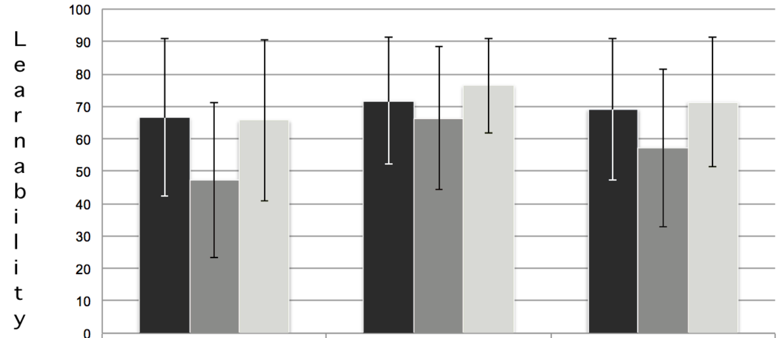
	Used as first tool	Used as second tool	Average
Aemoo	66.58 (17.58)	73.55 (11.53)	70.06 (15.08)
Relfinder	50.55 (14.72)	62.25 (16.56)	56.71 (16.41)
Google	70 (8.96)	76.94 (10.29)	73.67 (10.04)

Fig. 8. SUS scores and standard deviation values for Aemoo, RelFinder and Google. Standard deviation values are expressed between brackets and shown as black vertical lines in the chart.

their related standard deviations. Again there is no statical significance in the difference of values between Aemoo and Google both in terms of Learnability and Usability, regardless of the order of usage. Concerning the comparison between Aemoo and RelFinder we did not observed any statical significance in the difference of Learnability values, regardless of the order of usage. However, we observed a strong statistical significance ($0.01 < p \leq 0.05$) in the difference of Usability values between the two tools whether Aemoo was used as first tool or not. If we consider the average, such a difference reached a good statistical significance ($0.05 < p \leq 0.1$).

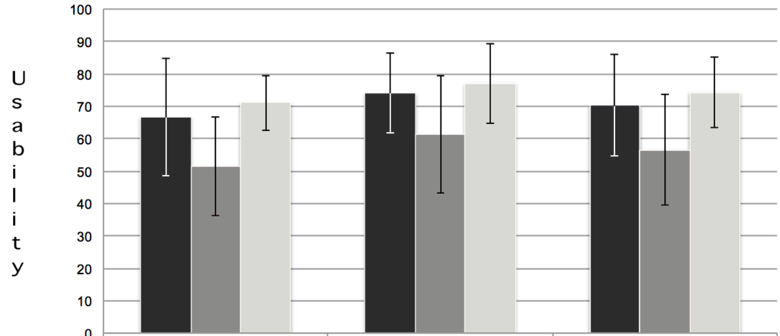
Then, we used the free-text answering questionnaires (cf. Table 5) to compute a Grounded theory [38]. We proceeded first with open coding and consequently with the so-called axial coding. The open coding is aimed at extracting actual relevant sentences - called codes - from the answers. The axial coding is the rephrasing of the original codes so as to have semantic connections emerge from them and generate concepts. We finally analysed the respective frequency of each emerged concept (defined as the number of codes which contributed to the concept's existence) so as to consider the most important issues arising from the answers. Fig. 5.2 shows the results of the most mentioned codes (ordinate values report the number of mentions normalised on a scale between 0 and 1).

Finally, we analysed the five-point scale values of relevance and unexpectedness that users provided to their findings relatively to Task 1. (cf. Section 4.2). Fig. 4.2 shows the aggregation (standard deviations are



	Used as first tool	Used as second tool	Average
Aemoo	66.45 (24.31)	71.71 (19.47)	69.08 (21.89)
Relfinder	47.22 (24.02)	66.25 (22.08)	57.24 (24.4)
Google	65.62 (24.77)	76.39 (14.58)	71.32 (20.13)

(a) Learnability.



	First tool	Second tool	Average
Aemoo	66.61 (18.07)	74.01 (12.41)	70.31 (15.75)
Relfinder	51.39 (15.16)	61.25 (18.11)	56.58 (17.08)
Google	71.09 (8.48)	77.08 (12.3)	74.26 (10.8)

(b) Usability.

Fig. 9. Learnability and Usability values and standard deviations. Standard deviation values are expressed between brackets and shown as black vertical lines in the chart.

reported between brackets) of these value regardless of the usage order of the systems during the experiments. We recorded better relevance (r) and unexpectedness (u) for Aemoo (r=4.11, u=2.92) than for RelFinder (r=3.97, u=2.64) and Google (r=4.05, u=2.2).

Discussion. Accuracy is the typical measure used to evaluate systems in information retrieval. Basically, more the precision and recall are high the more the system is accurate. Unfortunately, exploratory search tasks are typically associated with undefined and uncertain goals [46]. For example, there are a myriad of correct answers that an user can find for providing a summary about the topic "Alan Turing" (cf. Task 1). It is near impossible to classify all correct and wrong answers in order to use precision and recall appropriately. Hence, we performed: (i) the analysis of the time required by subjects for completing their tasks, (ii) the SUS, (ii) a grounded and (iv) an analysis of relevance and unexpectedness of results.

The time needed by subjects for completing the tasks provides just an idea about the effectiveness of the different tools for supporting subjects in exploratory search. In fact, time related values might be biased by a variety of factors that include, for example, users' expertise either about a certain topic or about using the system. As a matter of fact, the most users we interviewed declared to use Google Search daily.

Instead, with the SUS we obtained good results. We were mainly interested in comparing Aemoo with RelFinder, which is one of the most accepted ex-

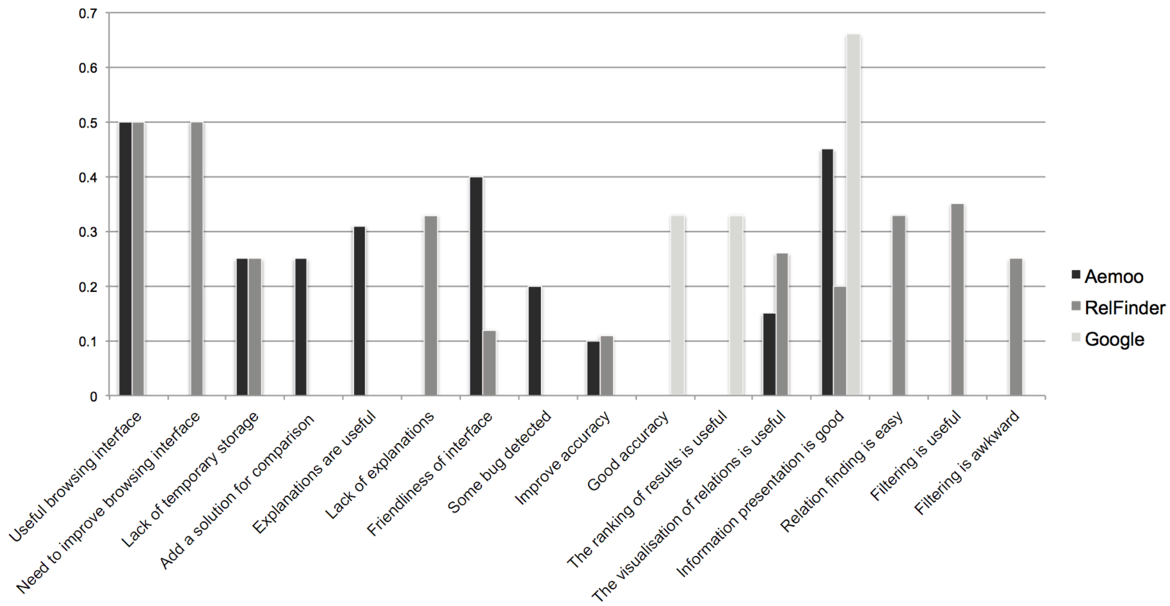
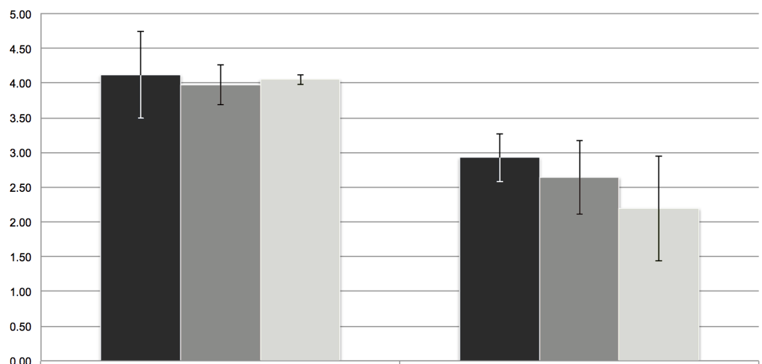


Fig. 10. A chart of the most mentioned pros and cons in the free-text answering questionnaires..



	Relevance	Unexpectedness
Aemoo	4.11 (0.62)	2.92 (0.34)
RelFinder	3.97 (0.28)	2.64 (0.53)
Google	4.05 (0.07)	2.20 (0.76)

Fig. 11. Relevance and unexpectedness values recorded by subjects. The values are reported along with standard deviations between brackets for each of the three tools.

ploratory search system in the Semantic Web community. Nevertheless, we included Google in the evaluation in order to scale the comparison between Aemoo and RelFinder with respect to one of the most used search engine. Aemoo surpassed on average the target of 68, which is required by the SUS to demonstrate a good level of usability [37], while RelFinder did not. Additionally Aemoo was perceived, with good statistical significance, more usable than RelFinder. This provides evidence about the system usability of Aemoo.

A grounded analysis (cf. Fig. 5.2) gave us some indicators about the main pros and cons. Concerning Aemoo, the main pros were its browsing interface, the explanations and the good presentation of information and relations. The main cons that we need to take into account for our future work include the need of a mechanism to perform comparison and the support to temporary storage (e.g., a basket). We also noticed that we miss an automatic relation finding mechanism that is, instead, provided by RelFinder. A comparable number of subjects judged useful and awkward to use at the same time the faceted browsing of RelFinder based on filters. This suggests that some filtering mechanism should be provided in a transparent wasy to the users. We believe that Aemoo addresses this requirement by leveraging the EKPs. Some subjects reported feedback about information presentation. In this respect, Google emerged among the three systems as the best to present information. Probably it is true, but we cannot ignore that, being one of the most used search engine in the world, there is an high-level of familiarity with its interface. If we consider only the systems that subjects were unfamiliar with, i.e., Aemoo and RelFinder, the former received an higher number of positive comments concerning information presentation. All these findings suggest that the visualisation of data provided by Aemoo is effective.

Finally Aemoo obtained on average the highest (including also Google) scores of relevance and unexpectedness. Hence, we claim that Aemoo returns *serendipitous* results. Serendipity can be informally defined as beneficial discovery that happens in an unexpected way and has been recently referred to as *unexpected relevance* [40]. The intuition is simple: more a result is at same time relevant and unexpected the more it is serendipitous. Aemoo shows the best ratio of relevance and unexpectedness among the three systems and this provides evidence about is ability to support humans in exploratory search by providing them with relevant results.

Thus we can conclude that our initial hypothesis (cf. Section 1) that EKPs are effective in supporting humans during exploratory search tasks is valid.

6. Related work

Many existing solutions for exploring Linked Data are based on semantic mash-up or browsing applications. Examples are [43,18,19] that leverage the semantic relations asserted in the linked datasets without applying any criterion for defining a boundary to provide users with tailored and contextualized knowledge. For example, RelFinder [18] provides the visualisation of existing relations between two of more DBpedia entities. These relations can be simple or can include more complex paths. The visualisation of relations can be filtered thanks to a faceted menu according to their length, the entity types and property names. Unfortunately, no automatic filtering is provided.

However, an increasing number of work has been focused on research tackling the knowledge boundary and heterogeneity problems for summarising, recommending or browsing Linked Data. Many of them present novel approaches, however, to best of our knowledge, none of them is based on a solution that leverage ontology patterns (i.e., EKPs) as a means for addressing previously mentioned problems.

Entity summarisation. In [41] is presented a diversity-aware algorithm, called DIVERSUM, that generates graphical entity summaries extracted from Wikipedia. The algorithm selects triples related to entity by both measuring their relevance and diversity. It is aimed at providing the most important triples about an entity with the higher coverage of diverse information. RELIN [8] computes entity summaries by using a variant of the random surfer model, which is based on two kinds of actions, i.e., relational move and informational jump that follow non-uniform probability distributions. It leverages the relatedness and informativeness of description elements for ranking entity triples. SUMMARUM [42] is an entity summarisation system that uses the PageRank algorithm. The PageRank is computed in order to assign relevance scores to the triples having as subject a certain DBpedia entity.

Recommending Linked Data. MORE [26] leverages DBpedia, Freebase and LinkedMDB in order to recommend movies. It computes similarities between movies thanks to an adaptation of the Vector Space Model (VSM). Seevl [31] is a recommendation system that provides personalised access and exploration of a knowledge based about music facts, which is created by exploiting DBpedia. The core of the system is an algorithm, called DBrec, which computes the relatedness among entities of the knowledge base by looking at shared relations, both incoming and outcoming. The authors in [20] present a recommendation system for retrieving music related to a point of interest (POI). The system exploits a spreading activation algorithm in order to weight the relatedness between musicians and POIs in DBpedia.

Browsing Linked Data The Discovery Hub [23] is an exploratory search system for DBpedia. It relies on spreading activation, which is a well known method inspired by cognitive science. Basically it starts by weighting an origin entity and consequently propagating the weights to its neighbors. The propagation is constrained by taking into account the entity types and by integrating a triple-based similarity measure. The Lookup Explore Discovery [25] (LED) is a system that exploits users' query in order to create tag clouds aimed at suggesting related knowledge to users during exploration search tasks. The tag clouds are created by means of named-entity recognition and linking to DBpedia resources. Factic [44] is a faceted exploration browser for the Semantic Web built upon the opening-midgame-endgame paradigm. Yovisto [45] is a platform that provides exploratory capabilities specialised in academic lecture recordings and conference talks. It proposes a ranked list of related topics during exploration based on content popularity rather than similarity-based methods. Additionally, it enables multi-faceted browsing. Rexplore [30] is a scholarly data exploration system, which facilitates the identification of research trends and "interesting" connections between researchers. It is based on a knowledge base automatically generated by using Klink [29], an algorithm which exploits machine learning methods, stochastic techniques and external knowledge (i.e., Wikipedia, Google Scholar and EventSeer) to infer semantic relationships between keywords in the scholarly domain. mSpace [17] a multi-column faceted spatial browser for multimedia data which shows a subset of the data at the time, called "a slice", and arranging them in a hierarchy of columns in accordance with user-defined priorities. Visor [33] facilitates the navigation process by introducing a multi-pivot paradigm, which allows users to identify key elements in the data space, called pivots.

7. Conclusions and future work

This paper presents a novel approach for Linked Data exploration that uses Encyclopedic Knowledge Patterns (EKPs) as relevance criteria for selecting, organising, and visualising knowledge. The EKPs were discovered by means of a method aimed at mining the linking structure of Wikipedia. Based on this approach, a system called Aemoo was implemented for supporting EKP-driven exploration as well as integration of data coming from heterogeneous resources, namely static (i.e., DBpedia and Wikipeida) and dynamic knowledge (i.e., Twitter and Google News).

The works grounds on two working hypotheses: (i) the cognitive soundness of EKPs and (ii) their effectiveness in summarising and aggregating knowledge to address exploratory search tasks.

The first hypothesis was validated by means of a user-study aimed at making emerge EKPs from human consensus in order to compare these EKPs with those extracted from Wikipedia. Results showed high values of correlation combined with good inter-rater agreement and very high reliability of human raters.

The second hypothesis was validated by means of controlled, task-driven user experiments in order to assess its usability, and ability to provide relevant and serendipitous information as compared to two existing tools: Google and RelFinder

Currently, we are working on several extensions. Examples include:

- improving the automatic interpretation of hypertext links by hybridizing NLP with Semantic Web techniques. In this respect we have recently obtained [35,36] very good results by designing a novel Open Knowledge Extraction (OKE) approach and its implementation, called Legalo [27], that performs unsupervised, open domain, and abstractive knowledge extraction from text for producing directly usable machine readable information;
- providing visual analytics interfaces that compare different entities having the same type;
- providing different views on the same entity by allowing users to change the applied lens, i.e., EKP;
- adding a basket functionality, allowing users to save the summary data of their exploration in RDF;
- integrating the EKP-based approach with user profiles for boundary creation.

References

[1] Alliance, T. O., September 2009. OSGi Service Platform Release 4 Version 4.2, Compendium Specification. Committee specification, Open Services Gateway initiative (OSGi).
[2] Alliance, T. O., September 2009. OSGi Service Platform Release 4 Version 4.2, Core Specification. Committee specification, Open Services Gateway initiative (OSGi).
[3] Alliance, T. O., March 2010. OSGi Service Platform Release 4 Version 4.2, Enterprise Specification. Committee specification, Open Services Gateway initiative (OSGi).
[4] Barker, K., Porter, B., Clark, P., 2001. A library of generic concepts for composing knowledge bases. In: Proceedings of the 1st international conference on Knowledge capture. ACM, pp. 14–21.
[5] Barsalou, L. W., 1999. Perceptual symbol systems. Behavioral and brain sciences 22 (04), 577–660.
[6] Berners-Lee, T., Hendler, J., Lassila, O., May 2001. The Semantic Web. Scientific American 284 (5), 34–43.

[27] http://wit.istc.cnr.it/stlab-tools/legalo

[7] Brooke, J., 1996. SUS: A quick and dirty usability scale. Usability evaluation in industry, 189–194.

[8] Cheng, G., Tran, T., Qu, Y., 2011. Relin: Relatedness and informativeness-based centrality for entity summarization. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N. F., Blomqvist, E. (Eds.), International Semantic Web Conference (1). Vol. 7031 of Lecture Notes in Computer Science. Springer, pp. 114–129.

[9] Clark, P., Thompson, J., Porter, B., 2000. Knowledge Patterns. In: Cohn, A. G., Giunchiglia, F., Selman, B. (Eds.), KR2000: Principles of Knowledge Representation and Reasoning. Morgan Kaufmann, San Francisco, pp. 591–600.

[10] Di Iorio, A., Nuzzolese, A. G., Peroni, S., Shotton, D., Vitali, F., 2014. Describing bibliographic references in rdf. In: Proceedings of 4th Workshop on Semantic Publishing (SePublica 2014). CEUR-WS.

[11] Fielding, R. T., 2000. REST: architectural styles and the design of network-based software architectures. Doctoral dissertation, University of California, Irvine.

[12] Fillmore, C., 1968. The case for the case. In: Bach, E., Harms, R. (Eds.), Universals in Linguistic Theory. Rinehart and Winston, New York.

[13] Fillmore, C. J., 1976. Frame semantics and the nature of language*. Annals of the New York Academy of Sciences 280 (1), 20–32.

[14] Gallese, V., Metzinger, T., 2003. Motor ontology: the representational reality of goals, actions and selves. Philosophical Psychology 16 (3), 365–388.

[15] Gangemi, A., Presutti, V., 2010. Towards a Pattern Science for the Semantic Web. Semantic Web 1 (1-2), 61–68.

[16] Garlan, D., Monroe, R. T., Wile, D., 2000. Acme: Architectural description of component-based systems. In: Leavens, G. T., Sitaraman, M. (Eds.), Foundations of Component-Based Systems. Cambridge University Press, pp. 47–68.

[17] Gibbins, N., Harris, S., Schraefel, M., 2003. Applying mSpace Interfaces to the Semantic Web. In: World Wide Web Conference 2004.

[18] Heim, P., Hellmann, S., Lehmann, J., Lohmann, S., Stegemann, T., 2009. RelFinder: Revealing Relationships in RDF Knowledge Bases. In: Proceedings of the 3rd International Conference on Semantic and Media Technologies (SAMT). Vol. 5887 of Lecture Notes in Computer Science. Springer, pp. 182–187.

[19] Heim, P., Ziegler, J., Lohmann, S., 2008. gFacet: A browser for the web of data. In: Auer, S., Dietzold, S., Lohmann, S., Ziegler, J. (Eds.), Proceedings of the International Workshop on Interacting with Multimedia Content in the Social Semantic Web (IMC-SSW'08). CEUR-WS, pp. 49–58.

[20] Kaminskas, M., Fernández-Tobías, I., Ricci, F., Cantador, I., 2012. Knowledge-based music retrieval for places of interest. In: Liem, C. C. S., Müller, M., Tjoa, S. K., Tzanetakis, G. (Eds.), MIRUM. pp. 19–24.

[21] Lewis, J. R., Sauro, J., 2009. The Factor Structure of the System Usability Scale. In: Kurosu, M. (Ed.), HCI (10). Vol. 5619 of Lecture Notes in Computer Science. Springer, pp. 94–103.

[22] Marchionini, G., Apr. 2006. Exploratory search: from finding to understanding. Commun. ACM 49 (4), 41–46.

[23] Marie, N., Gandon, F., Ribière, M., Rodio, F., 2013. Discovery hub: on-the-fly linked data exploratory search. In: Sabou, M., Blomqvist, E., Noia, T. D., Sack, H., Pellegrini, T. (Eds.), I-SEMANTICS. ACM, pp. 17–24.

[24] Minsky, M., 1975. A Framework for Representing Knowledge. In: Winston, P. (Ed.), The Psychology of Computer Vision. McGraw-Hill.

[25] Mirizzi, R., Di Noia, T., 2010. From exploratory search to web search and back. In: Proceedings of the 3rd workshop on Ph.D. students in information and knowledge management. PIKM '10. ACM, New York, NY, USA, pp. 39–46.

[26] Noia, T. D., Mirizzi, R., Ostuni, V. C., Romito, D., Zanker, M., 2012. Linked open data to support content-based recommender systems. In: Presutti, V., Pinto, H. S. (Eds.), I-SEMANTICS. ACM, pp. 1–8.

[27] Nuzzolese, A. G., Gangemi, A., Presutti, V., Ciancarini, P., 2011. Encyclopedic Knowledge Patterns from Wikipedia Links. In: Aroyo, L., Noy, N., Welty, C. (Eds.), Proceedings fo the 10th International Semantic Web Conference (ISWC 2011). Springer, pp. 520–536.

[28] Nuzzolese, A. G., Presutti, V., Gangemi, A., Musetti, A., Ciancarini, P., 2013. Aemoo: Exploring knowledge on the web. In: Proceedings of the 5th Annual ACM Web Science Conference. ACM, pp. 272–275.

[29] Osborne, F., Motta, E., 2012. Mining Semantic Relations between Research Areas. In: Cudré-Mauroux, P., Heflin, J., Sirin, E., Tudorache, T., Euzenat, J., Hauswirth, M., Parreira, J. X., Hendler, J., Schreiber, G., Bernstein, A., Blomqvist, E. (Eds.), International Semantic Web Conference (1). Vol. 7649 of Lecture Notes in Computer Science. Springer, pp. 410–426.

[30] Osborne, F., Motta, E., 2013. Exploring research trends with rexplore. D-Lib Magazine 19 (9/10).

[31] Passant, A., 2010. dbrec - Music Recommendations Using DBpedia. In: Proceedings of the 9th International Semantic Web Conference (ISWC 2010). Springer, pp. 209–224.

[32] Peroni, S., Shotton, D., 2012. Fabio and cito: Ontologies for describing bibliographic resources and citations. J. Web Sem. 17, 33–43.
URL `http://dblp.uni-trier.de/db/journals/ws/ws17.html#PeroniS12`

[33] Popov, I. O., Schraefel, M. M. C., Hall, W., Shadbolt, N., 2011. Connecting the dots: A multi-pivot approach to data exploration. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N. F., Blomqvist, E. (Eds.), International Semantic Web Conference (1). Vol. 7031 of Lecture Notes in Computer Science. Springer, pp. 553–568.

[34] Presutti, V., Aroyo, L., Adamou, R., Gangemi, A., Schreiber, G., 2011. Extracting core knowledge from linked data. In: In COLD2011. Citeseer.

[35] Presutti, V., Consoli, S., Nuzzolese, A. G., Reforgiato Recupero, D., Gangemi, A., Bannour, I., Zargayouna, H., 2014. Uncovering the semantics of wikipedia pagelinks. In: Janowicz, K., Schlobach, S., Lambrix, P., Hyvönen, E. (Eds.), Knowledge Engineering and Knowledge Management. Vol. 8876 of Lecture Notes in Computer Science. Springer International Publishing, pp. 413–428.

[36] Presutti, V., Nuzzolese, A. G., Consoli, S., Recupero, D. R., Gangemi, A., 2014. From hyperlinks to Semantic Web properties using Open Knowledge Extraction. Submitted to Semantic Web Journal.

[37] Sauro, J., 2011. A practical guide to the system usability scale: Background, benchmarks & best practices. Measuring Usability LCC.

[38] Strauss, A., Corbin, J., 1998. Basics of Qualitative Research: Techniques and Procedures for developing Grounded Theory.

Sage Publications Inc.

[39] Suchanek, F., Kasneci, G., Weikum, G., 2008. Yago - A Large Ontology from Wikipedia and WordNet. Elsevier Journal of Web Semantics 6 (3), 203–217.

[40] Sun, T., Zhang, M., Mei, Q., 2013. Unexpected relevance: An empirical study of serendipity in retweets. In: Kiciman, E., Ellison, N. B., Hogan, B., Resnick, P., Soboroff, I. (Eds.), ICWSM. The AAAI Press.

[41] Sydow, M., Pikula, M., Schenkel, R., 2013. The notion of diversity in graphical entity summarisation on semantic knowledge graphs. J. Intell. Inf. Syst. 41 (2), 109–149.

[42] Thalhammer, A., Rettinger, A., ???? Browsing DBpedia Entities with Summaries. In: Proceedings of the 11th Extended Semantic Web Conference 2014 (ESWC 2014), Poster & Demo.

[43] Tummarello, G., Cyganiak, R., Catasta, M., Danielczyk, S., Delbru, R., Decker, S., 2010. Sig.ma: Live views on the web of data. Web Semantics: Science, Services and Agents on the World Wide Web 8 (4), 355 – 364, semantic Web Challenge 2009, User Interaction in Semantic Web research.

[44] Tvarozek, M., Bieliková, M., 2010. Factic: Personalized Exploratory Search in the Semantic Web. In: Benatallah, B., Casati, F., Kappel, G., Rossi, G. (Eds.), ICWE. Vol. 6189 of Lecture Notes in Computer Science. Springer, pp. 527–530.

[45] Waitelonis, J., Sack, H., 2012. Towards exploratory video search using linked data. Multimedia Tools Appl. 59 (2), 645–672.

[46] White, R. W., Kules, B., Bederson, B. B., 2005. Exploratory search interfaces: categorization, clustering and beyond: report on the xsi 2005 workshop at the human-computer interaction laboratory, university of maryland. SIGIR Forum 39 (2), 52–56.