Empowering Exploratory Search on Linked Movie Open Data with Semantic Technologies

Thi-Nhu NGUYEN
Haiphong University
171, Phan Dang Luu Street
Haiphong, Vietnam
(84) 984 366 869
nhunt@fithpu.edu.vn

Duy-Thanh DINH
Hanoi University of Sience and Technology
1, Dai Co Viet Street
Hanoi, Vietnam
(84) 979 869 992
dinhduythanh51292@gmail.com

Tuan-Dung CAO
Hanoi University of Science and Technology
1, Dai Co Viet street
Hanoi, Vietnam
84-4-38682595
dungct@soict.hust.edu.vn

ABSTRACT

Nowadays, Linked Open Data (LOD) has grown rapidly to become large open datasets defined by RDF standards. Thanks to development of Data Web, the information on LOD is increasingly deeper, larger and easier to link in multi-domains, constituting the Linked Open Data cloud. Recently end-users applications using linked data sources as background knowledge appeared [15]. Thus, the exploitation of information on LOD effectively brings tremendous values as well as challenges. Meanwhile, Exploratory Search (ES) describes information-seeking processes that are opportunistic, iterative, and multi-tactical [7]. Furthermore, systems based on ES capitalize on new technological capabilities and interface paradigms that facilitate an increased level of interaction with information. In this paper, we present a method on searching and recommending information to empower exploratory search with semantic technologies. Our aim is to use algorithms with incorporation of structured semantics in search to give users the best related-semantic results and enhance users' interactions. We have exploited the data within LinkedMDB[1] to support users in finding some information in the movie domain.

Categories and Subject Descriptors

• Theory of computation ~Semantics and reasoning
• Computing methodologies ~Natural language processing
• Software and its engineering ~Software development techniques • *Networks ~Public Internet*

General Terms

Algorithms, Management, Measurement, Documentation, Performance, Design, Experimentation, Human Factors.

SoICT 2015, December 03-04, 2015, Hue City, Viet Nam
© 2015 ACM. ISBN 978-1-4503-3843-1/15/12…$15.00
DOI: http://dx.doi.org/10.1145/2833258.2833283.

[1] http://www. linkedmdb.org

Keywords

Entity Recognition; Exploratory Search; Spreading Activation; Linked Data.

1. INTRODUCTION

The rapid development of the network systems, especially the Internet, has marked the number of users approximately more than three billion in 2014. Consequently, demand for mining and using data effectively is a huge. By helping us to make better decisions, this data is playing an increasingly central role in our lives and driving the emergence of a data economy [11]. In addition, the increasing in the number of published datasets on the Web makes LOD promote the possibilities such as large scale, constantly growing, covering multiple domains.

However, the current searching and exploiting information systems are mostly keyword searching i.e., users type a phrase or a sentence and get the documents containing keywords from the input queries. These systems can show answers quickly and exactly, but they are not adequate. For example, the users do not know exactly what they want and they only have some ambiguous definitions. Therefore, it is difficult to describe keywords to search. Besides, these search engines cannot suggest users some significant documents. Furthermore, these documents are not relevant to the input semantically. To exploit information more effectively, the search systems are required to supply users more information or concepts that they want to search.

In contrast to regular search, Exploratory Search gives a more complete view of the topic [20]. It is a new form of search system, in which the queries are not required to put in the exact key words, they may be a concept or vague information. Results of search process are not constant, they are open-ended. And then, the users can keep on exploring and expanding the queries until coming to the expected results. Exploration demands more time, effort and creativity from the user, but rewards the user with deeper knowledge [12]. This method is particularly beneficial for ill-structured problems and more open-ended goals. Exploratory search systems are the most advanced systems in the field of linked data based exploration [14].

Thus, in this paper we propose a method to empower exploratory search to retrieve the data from LOD effectively with semantic technologies. Firstly, we search entities by the SPARQL language basing on *sliding window* and *query* *expanding* technologies. Then, we give some suggestions related semantically to the user's choice. This exploration continues with *semantic relatedness measures, spreading activation* and *ranking* to give the best results.

Based on this approach, we have developed a system to search and recommend interesting movies to users on LinkedMDB. The users not only find out some limited results, they also exploit more new ones by themselves. They can change or adjust their query to make it become suitable for their need. The results may be not completely accurate to its initial request, but the users will be supplied more relevant knowledge. This will make them feel excited at searching and exploring with our system. For example, a user wants to search a movie involved performance of the main actor and the main actress of "*Titanic*" film but he does not know their names, how does he do? In our system, he only needs to give input "*Titanic*". At first, he gets names of two main actors "*Leonardo DiCaprio, Kate Winslet*". Then, with his exploration, the system will provide *"Revolutionary Road"* film in which these film stars acted. Moreover, the users can change their searching goals because the system not only gives data about the main actors but it also supplies information about the director, editor and characters of that movie.

The remainder of the paper is structured as follows: Section 2 shows an entity search approach and system overview; section 3 presents exploratory search; implementation is presented in section 4; section 5 discusses some of the technologies used in actualizing the system and previous works done on the same field of study; finally, a conclusion is given in Section 6 with some further suggested studies.

2. SYSTEM OVERVIEW

By using algorithms based on exploratory search with semantic technologies, the main goal of our system is to support users in searching and recommending information about movie domain. To do this, the user has to provide an input for the system first. Queries of users can be random keywords, phases or sentences formatted in natural language. These queries should be relevant to the movie field, i.e. actor, director or film to return results more accurate. Requests are sent to the *query analysis and entity search* component. This component has three steps in handling the queries, including Sliding Window, Expanding Query and Ranking Results by using SPARQL. The first step starts with the determination of key entities. They are entities which can be potentially mapped to instances or classes of ontology of LinkedMDB. Then, the system will return an accurate result set. The second step's aim is to complement these results based on attention to semantic aspect of the query. Finally, our system returns ranked results to the users in categories such as TOP and RECOMMEND.

Furthermore, from the above list, the users can choose more entities to make new queries. The results are returned by algorithms such as spreading activation and semantic relatedness measure. During exploratory search, the users can change or adjust their queries to restart information searching. To return the most complete results and best recommendation, our system accesses the information on LinkedMDB, Freebase[2], IMDB[3] and DBpedia[4]. In particular, LinkedMDB is a LOD dataset about movies in the system. It has ex-link with Freebase and IMDB. In which, Freebase is the data store of Google, it contains twenty three millions entities and topics about human, location and items. It stores the information of entities on LinkedMDB. This makes users have a more exact view about what they want to search. Our system retrieves information from IMDB that stores the details about famous movies and their evaluations. This supports in ranking the results of Exploratory Search. And DBpedia is a big LOD, it allows users to query the relationships and properties related to Wikipedia.

The system overview architecture is provided in Figure 1 below.

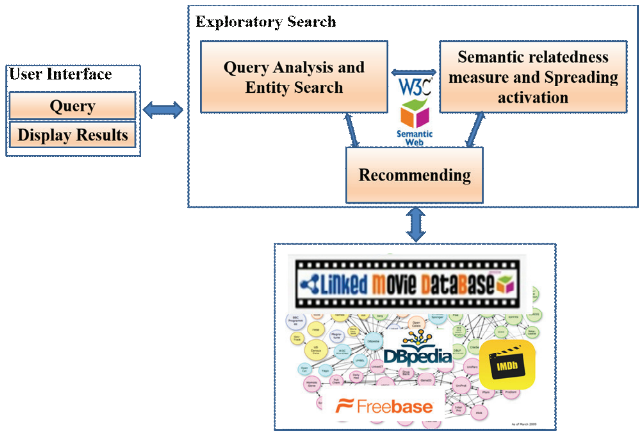


Figure 1. Exploratory search system architecture.

3. EXPLORATORY SEARCH

3.1 Query Analysis and Entity Search

The query processing will help users get the initial assumptions about what they want to search. The key entities are presented in natural language query such as "*actor of the film Annie Hall"* or keywords as *Leonardo DiCaprio, Kate Winslet.* The users can type a random text or even they only have some ideas about keywords to make a query in the movie domain.

In this section, we will show how the original request of user is separated into queries and executed them on LOD. We use "*sliding window"* and "*query expanding"* to do this.

3.1.1 Sliding Window

Queries are analyzed from beginning to end and this analysis is divided to many steps. With each step, a sliding window is created and it includes a part of query. The initial size of window is max and slides at the first word of input. The size of substrings in query is equal to the window size. If there are results when executing a query, they are put on a list and the size of window will be reduced by 1. Then the query continues to be implemented. When the window size reduces to 0, the window is moved to next word of input. This process is repeated until all of substrings slid.

Choosing the size of window should base on reality and it is not too small because the entities whose size is larger than

[2] https://www.freebase.com

[3] http://www.imdb.com

[4] http://wiki.dbpedia.org

window will be not slid over. For example, if size of window is 2 and the user types "Alice in Wonderland", it means they want to find this film. According to this algorithm, we have substrings as "Alice in","Alice", "in Wonderland", "in", "Wonderland", in which "Alice in Wonderland" does not exit, so this movie will be not found out. Obiviously, if the choosen size is too small, some *exact* entities will be lost. However, the larger size is the more substring is separated. And this makes the volume of queries, calculations and response time larger. Thus, we use the following formula to compute the number of the substrings:

$$N = \frac{k(2 \times a - k + 1)}{2} \qquad (1)$$

Where, N is the number of substrings, k is the size of sliding window and a is the length of input string.

The system is applied in practice, thus it not only returns accurate results but it also pays attention to the response time. The actual testing process on LinkedMDB shows the window size is 3 will best meet the above requirements. Let's consider the following example "*actor of the film Annie Hall*" (a query from the user). Table 1 describes sliding windows in size 3. Table 2 indicates the operation of "*sliding window*", shows selected words and how to search. Table 3 illustrates the used SPARQL queries.

Table 1: Sliding Windows

actor	**Of**	**The**	**film**	**Annie**	**Hall**
actor	*Of*	*The*			
	Of	*The*	*film*		
		The	*film*	*Annie*	
			film	*Annie*	*Hall*

Table 2: The operation of "sliding window"

No	**Sliding window**	**Result/Action**
1	*actor of the*	Not found, size reduced
2	*actor of*	Not found, size reduced
3	*actor*	Found, pushed to the result list, size reduced
4	*of the film*	Not found, size reduced
...	...	...
...	*Annie Hall*	Found, pushed to the result list, size reduced
...	...	...

Table 3: SPARQL Queries in Use

SPARQL Query	**Goals**
*SELECT * WHERE {* *{?s dc:title <searchText>} UNION* *{?s movie:actor_name 'searchText'}UNION* *{?s movie:director_name* *'searchText'} UNION* .... *}*	Search entities like *name* exactly

*SELECT * WHERE {*
 ?s rdfs:label ?o . FILTER regex(?o, 'searchText','i')
 }
	Search entities which have keywords in *name*

3.1.2 Query Expanding

The query is not only realized by keywords, it also need be understood semantically, for example "*actor of, writer of*". As a result, *"sliding window"* will not meet the requirements of the user completely. Because we want users to have the experience with Exploratory Search method, we improve the search by expanding the queries. When they include the keywords as *film, actor, writer*, we apply *movie:type* query to get the respective entities. The system scans automatically all of classes in LOD (i.e. actor, film, producer, editor, film art, director, etc.) to map the exact entities.

Nevertheless, the user can still add the related entities to search more.

For the mentioned query above, after implementing *"sliding window"*, the "*Annie Hall"* movie is one of the results, and then the system will search actors of this film with respective URIs. We use the following query:

```
SELECT * WHERE {
   <URI> rdf:type movie:actor
}
```

3.1.3 Ranking the results

The results are a set of URIs after analyzing the query and searching the entity. This dataset is not sorted and lacks the descriptions, it therefore needs to be handled.

The results are divided into two categories:

- *Top:* includes the exact entities and the one in expanding search. Besides, it has the options as *film, actor, director* to make the queries. In addition, the user can search by using the single word filtered from the initial input. For example, *"Annie Hall", "Annie", "Hall"*.
- *Recommend:* The entities are found by expanding query with the *keywords in name*.

For the above mentioned example, output will be classified by keywords. In this query, a set of keywords is S= *{"actor", "film", "Annie Hall", "Annie", "Hall"}*. Each word has a set of entities (URI) on LOD. For example, with "*Annie":*

E*("Annie")={"Annie (Film)", "Annie (Musical Drama)", "Annie Hall (Film)", "Annie Charrier (Editor)", ...}*. These entities are stored as shown in Fig. 2 to sort the results.

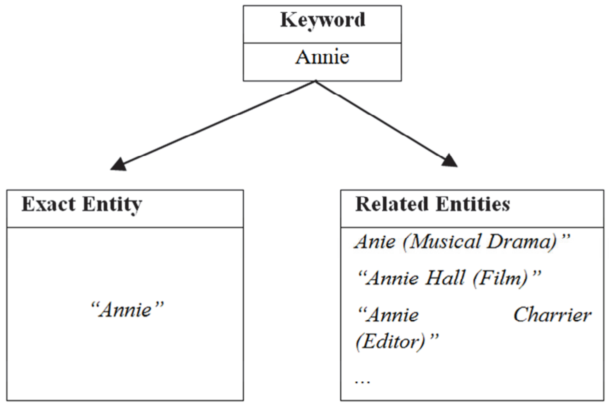


Figure 2. The returned results.

Each entity on LOD contains the information saved on Freebase and IMDB. Accessing these pages need the respective API.

For Freebase:

"https://www.googleapis.com/freebase/v1/topic+<ID>+<key API>"

For IMDB:

"http://www.omdbapi.com/?i=+<ID>"

3.2 Semantic relatedness measure and Spreading Activation

3.2.1 Semantic relatedness measure

The main goal of Exploratory Search is to propose the suitable concepts that relate semantically to the queries. The quality and the accuracy of the results not only depend on similarity of two concepts but they are also impacted by measuring the semantic relatedness a lot. It can be clearly seen from 02 entities of Johhny Depp and Titanic as examples on LinkedMDB that we hardly clarify their relationship if we only focus on their similarities. Because they are not the same category and their vocabulary are different ("Johnny Depp" –Actor and "Titanic" – Film). However, this relationship can be determined by using the semantic relatedness measures. For example, "Johnny Depp" and "Leonardo DiCaprio" are the same actors of "What's Eating Gilbert Grape" film and "Leonardo DiCaprio" is the main actor of "Titanic". This section describes the concepts of the semantic relatedness and calculation formulas used in this study.

There are many approaches proposed to measure semantic relatedness in this field. To meet requirements of our aim, we use a combination of two approaches. Those are Wikipedia Link-based Measure (WLM) (formula 2) by Milne & Witten [13] and Normalized Google Distance (NGD) (formula 3) [13], [5] with the following formulas.

If s and t are the source and target articles, then the weight w of the link $s \to t$ is:

$$w(s \to t) = \log\left(\frac{|W|}{|T|}\right) \quad , if\ s \in T\ , 0\ otherwise \quad (2)$$

where, s and t are the source and target articles. W is the set of all articles in Wikipedia. T is the set of all articles that link to t. It is clearly that the weight of a link is the inverse probability of any link being made to the target, or 0 if the link does not exist [9]. The fact that two articles both link to *science* is much less significant than if they both link to a specific topic such as *atmospheric thermodynamics*.

And,

$$r(a,b) = \frac{\log(\max(|A|,|B|)) - \log(|A \cap B|)}{\log(|M|) - \log(\min(|A|,|B|))} \quad (3)$$

Where, a and b are the two articles of interest, A and B are the sets of all articles that link to a and b respectively. M is the set of all articles on LOD. We use both of these two measures in this work.

3.2.2 Spreading activation

Spreading activation has its origins associated with modeling the human semantic memory in cognitive psychology and has a history of applications in cognitive psychology, artificial intelligence and, more recently, on information retrieval [1]. The recent emergence of the Linked Data Web is likely to motivate new investigations in spreading activation techniques. This method uses ontology and some technologies to find the concepts that relate to the input. The main idea is exploring the relationships of the ontologies, where, the relationships have weight and labels or direction as shown in figure 3.

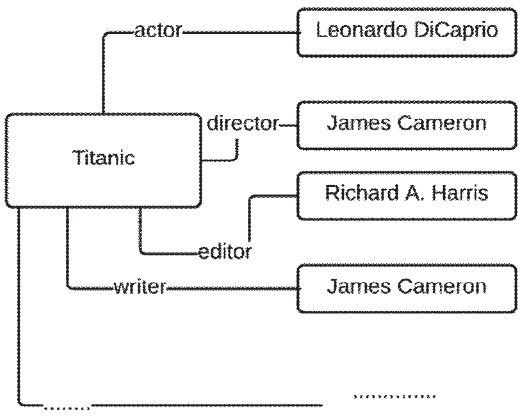


Figure 3. The relationships and properties of "Titanic" film.

First, spreading activation receives a set of concepts from the previous process. From these concepts, the system will return a set of related ones by propagation according to the relationships in ontology. This process is repeated till a stop condition is reached. The active concepts are weighted and added to the initial query.

Basing on the above definition and using the semantic relatedness, we propose a spreading activation method to suggest the related entities to the users. They decide to continue the spread by selecting the next query. In addition, they can choose an input again to achieve the desired results after each spreading step. For example in Fig.4, the purpose of the user is to look for the films acted by main actors of both "Titanic" and "Russell Crowe". Input order is the weighted spread, the first entity has the max weight and the next one decreases to the end. The algorithm determinates all of the entities are properties of *"Titanic"* film that their weights are largest.

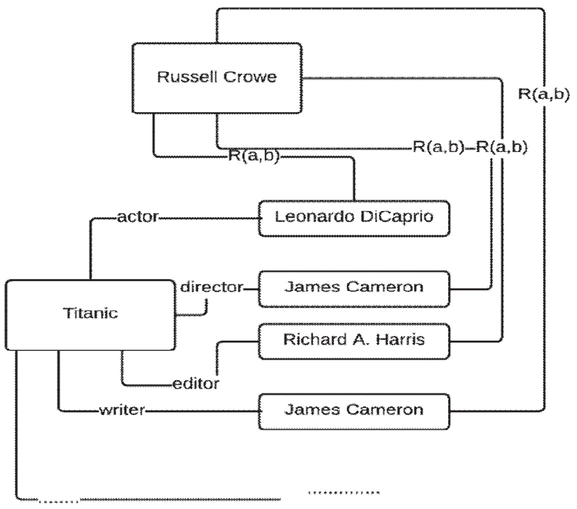


Figure 4. The semantic relatedness measure.

And then, it measures semantic relatedness of each entity with the next pivot element, for example *"Russell Crowe"* actor. This returns an order to help the user to choose again and adjust the pivot element list for the next spread time.

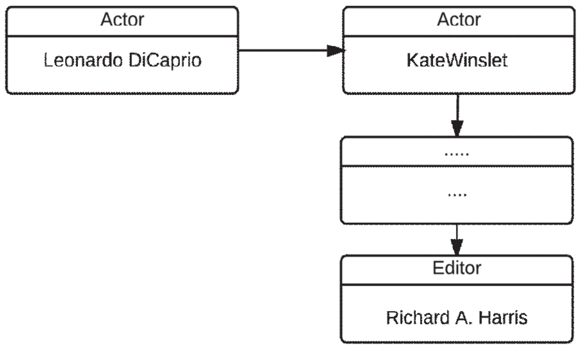


Figure 5. The sorted results by using the semantic relatedness measure.

Figure 5 illustrates that the first proposed entity is *"Leonardo DiCaprio"* actor. If the user adds this element to the query and performs searching again, the *"Body of Lies"* film is found and it means the user achieves his goal.

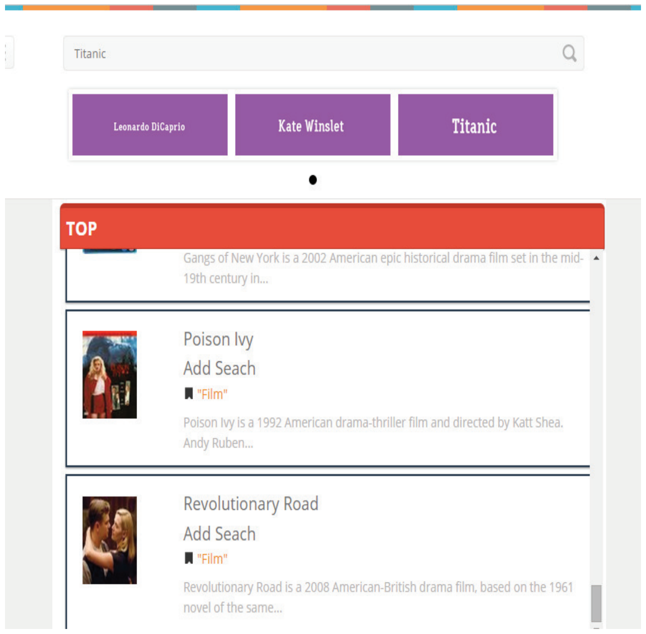
4. IMPLEMENTATION

To satisfy the requirements of the Exploratory Search, we build a searching and recommending system about movies. We exploit data on LinkedMDB which is the large Linked Data of Semantic Web about movie domain. Searching on LinkedMDB only returns URIs, thus we query the information from other LODs such as DBpedia, IMDB, Freebase. This will supply to the user the helpful information and make our system become more realistic. Executing many queries on many different LODs, we need to handle asynchronous. In this work we develop an application on Web to process pure data with ajax/javascript. This makes the user have better experiences in interaction and speed.

4.1 Search Results

Our system is an exploratory search engine to help the user explore interest of the movie domain. It is set up and run on *http://exploratorysearch.esy.es/* website. Searching is started after user makes a query. The system returns the related entities basing on the algorithms (section 3.1). Each returned entity has Add EX Search property and the user can select to further query. After adding an entity and pressing the *EX Search* button, a set of the most semantic relatedness entities are returned (section 3.2).

In above mentioned example, a user wants to search another movie with performance of both the main actor and the main actress of Titanic but he does not know their names. He only needs to query "*Titanic*", our system will return names of two main actors "*Leonardo DiCaprio, Kate Winslet*". If he adds these two entities to explore, he will find out the *"Revolutionary Road"* film he wants. Figure 6 shows the results page of Exploratory Search.



Gangs of New York is a 2002 American epic historical drama film set in the mid-19th century in...

Poison Ivy

"Film"

Poison Ivy is a 1992 American drama-thriller film and directed by Katt Shea. Andy Ruben...

Revolutionary Road

"Film"

Revolutionary Road is a 2008 American-British drama film, based on the 1961 novel of the same...

Figure 6. The results of Exploratory Search.

Besides, the user can adjust the search target immediately by selecting other entities such as director, editor. For example, from "Titanic" query, the user can know that *James Cameron* is the director of this film. If he wants to search a film that *James Cameron* is director and *Leonardo DiCaprio* is actor, it is possible.

The goal of this engine is not only to supply the users the links to the results but it is also helpful to the users who understand thoroughly about the interest and want to retrieve surprising and unexpected information. This is not searching, it is digging.

4.2 Top/Recommend

Once searched, our system gives suggestions to user in The recommending function based on the semantic relatedness measure of results. And they are displayed on the TOP and RECOMMEND pages. At *Top*, the search results related to the remaining ones semantically are displayed immediately. This allows reducing the waiting time. The *Recommend* page displays ranked results after exploratory search. Figure 7 shows example results of Top and Recommend.

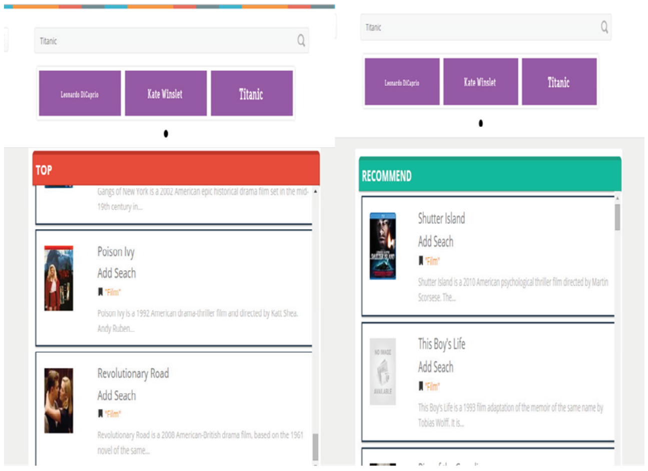
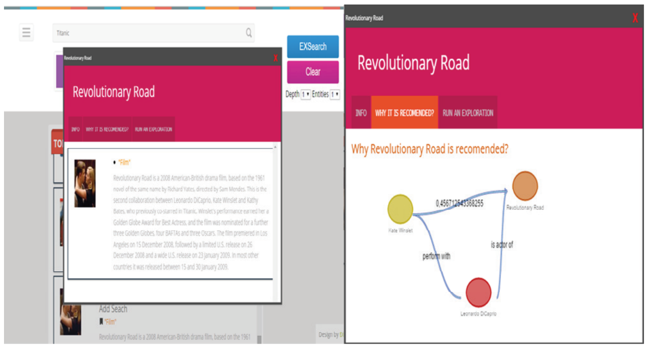


Figure 7. Top and recommend pages.

Further, in order to answer the question why it (an entity) is recommended, we give detail information about it and a suggestive graph, see figure 8.



a) Detail Information b) Suggestive Graph

Figure 8. Detail information and Suggest graph of recommended film.

4.3 Evaluation

Our system focuses on recommending the suitable information for user's exploration. There are many evaluation methods, but in this study, we use the hybridization of three kinds of index: precision, recall and F-Measure index. These indexes are often used in the semantic recommend systems.

Precision is the fraction of retrieved documents that are relevant to the query. *Recall* in information retrieval is the fraction of the documents related to the query that is successfully retrieved. A measure that combines precision and recall is the harmonic mean of precision and recall, the traditional *F-measure* or balanced *F-score*.

$$precision = \frac{(\{relevantdocuments\} \cap \{retrieveddocuments\})}{(\{retrieveddocuments\})}$$

$$recall = \frac{(\{relevantdocuments\} \cap \{retrieveddocuments\})}{(\{relevantdocuments\})}$$

$$F = \frac{2 \times precision \times recall}{precision + recall}$$

The below tables are calculated by the actual results. Results in Table 4 and Table 5 are averaged by two parts: *Top* and *Recommend*.

Table 4. Results of measure after query analysis and entity search

Query Analysis and Entity Search	Avg. Precision	Avg. Recall	Avg. F-Measure
	76%	82%	78.88%

Table 5. Results of measure after Exploratory Search

Spreading Activation	Deep	Avg. Precision	Avg. Recall	Avg. F-Measure
	1	81%	77%	78.95%
	2	87%	84%	85.47%

Additionally, in order to having a more objective result, we get assessments of users by using Google Analysis to collect on http://exploratorysearch.esy.es. Collected data is showed in following figures.

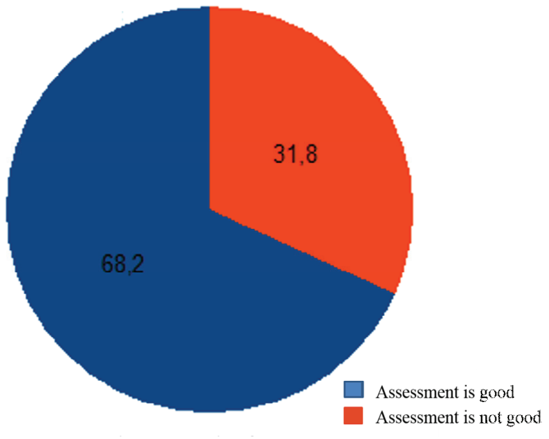


Figure 9. Rating from users.

Figure 9 shows assessment of users about system, where *good* is approximate 70 percentages and not good is about 30 percentages. The information of activities of users and ratio among parts of searching are illustrated in Figure 10 and 11.

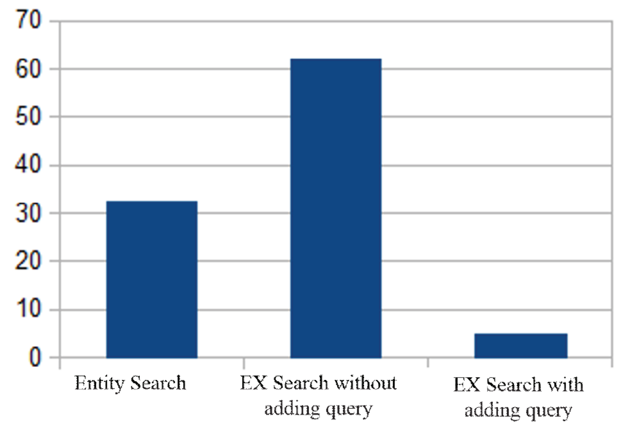


Figure 10. The ratio among parts of searching from users.

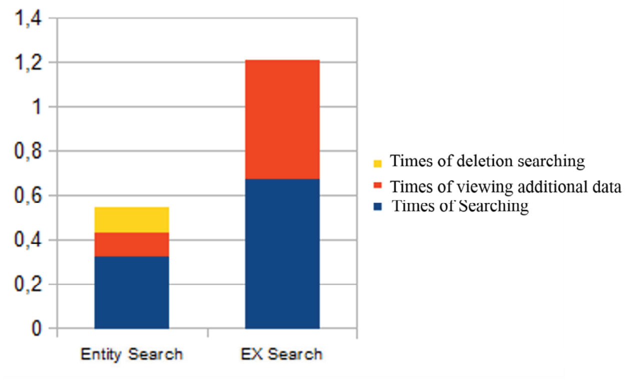


Figure 11. The information about activities of users.

5. RELATED WORK

Exploratory Search is a method concerned by many researchers with a lot of researches under different approaches. Yovisto[5] (2009) is an academic videos search engine that provides a linked data based exploratory search feature. The exploratory search feature retrieves topic suggestions that are semantically related to the users' query. When the user enters a query, Yovisto retrieves a set of semantically interrelated suggestions. These suggestions are computed over DBpedia by using a property ranking process based on a set of heuristics.

Discovery Hub[6] [15] implements a linked data exploration framework based on a semantic-sensitive traversal algorithm coupled to a live graph sampling technique. Discovery Hub offers faceted browsing and multiple result explanations features. Several advanced query modes are supported by the framework including the specification of facets of interest and disinterest about the topic explored and the injection of randomness during the data processing to retrieve more surprising results.

inWalk[7] is a web application for high level linked data exploration [18]. It allows the users to interact with a graph of clusters named inCloud. An inCloud is a set of linked data based clusters, computed according to a manually declared list of properties. The clusters are related to each other by a semantic proximity link. The current inWalk web application is implemented on a Freebase subset related to athletes and celebrities. InWalk aims at giving high-level overview on a domain and to ease its appropriation by the users.

[5] http://www.yovisto.com

[6] http://discoveryhub.co/

[7] http://islab.di.unimi.it/inWalkProject/

MORE [17] (More than Movie Recommendation) is a movie recommender system. The recommendation is mainly based on the computation of a (semantic) similarity among movies that users might be interested in. The movies are identified by resources described in two LOD datasets: DBpedia and LinkedMDB.
Seevl [6] is a DBpedia-based band recommender for Youtube based on the DBrec algorithm. The DBrec algorithm ranks the similar bands according to shared direct and indirect properties. The ranking is processed offline and stored in RDF.

6. CONCLUSIONS

In this article, we introduced an approach to search information on the LOD datasets. We developed a system to help the user finding films at *http://exploratorysearch.esy.es/* website. After an exploratory turn, the related, hidden and helpful entities are returned to help the users to clarify what they want. In addition, the system helps them explore more new and surprising information, too. Our user's evaluations demonstrated the effectiveness with high accuracy and the more complete data. This makes ES become appropriate to the user's interests.

In this paper, we only exploited the data within LinkedMDB to collect information about movies, actors, directors, etc. For further studies, we will address more concerns including exploring the other LOD such as DBpedia to complete the knowledgebase as well as the relationships and handling the bridging relations to bring more amazing results.

7. REFERENCES

[1] A. Passant,. 2010. dbrec – music recommendations using dbpedia. ISWC 2010

[2] André Freitas, João Gabriel Oliveira, Edward Curry, Seán O'Riain, and João Carlos Pereira da Silva, "Treo: Combining Entity-Search, Spreading Activation and Semantic Relatedness for Querying Linked Data", Digital Enterprise Research Institute (DERI). http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.417.8446

[3] Bastian Quilitz, Ulf Leser, "Querying Distributed RDF Data Sources with SPARQL", The Semantic Web: Research and Applications Lecture Notes in Computer Science Volume 5021, 2008, pp 524-538

[4] C Bizer, T Heath, K Idehen and T Berners-Lee, "Linked data on the web", publish in Proceedings of the 17th international conference on World Wide Web, pp 1265-1266

[5] Cilibrasi, R.L. and Vitanyi, P.M.B.: The Google Similarity Distance, IEEE Transactions on Knowledge and Data Engineering, vol. 19, no. 3, 370-383 (2007).

[6] Crestani, F. Application of Spreading Activation Techniques in Information Re- trieval, Artificial Intelligence Review, vol. 11, no. 6, pp. 453-453 (1997)

[7] D. F. Huynh and D. Karger. Parallax and companion: Set-based browsing for the data web. In WWW Conference. ACM. Citeseer, 2009.

[8] E., Markovitch, S.: Computing semantic relatedness using Wikipediabased explicit semantic analysis. International Joint Conference On Artificial Intelligence (2007).

[9] Fox C. (1990). A stop list for general text. ACM-SIGIR Forum.

[10] Jeen Broekstra, Arjohn Kampman, Frank van Harmelen, "Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema", The Semantic Web — ISWC 2002 - Lecture Notes in Computer Science Volume 2342, 2002, pp 54-68.

[11] Jim Ericson. Net expectations - what a web data service economy implies for business. Information Management Magazine, Jan/Feb, 2010

[12] Marchionini, G. (2006). Exploratory search: from finding to understanding. Communications of ACM 49(4), pp. 41–46. http://dl.acm.org/citation.cfm?id=1121979.

[13] Milne, D. and Witten, I.H. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In Proceedings of the first AAAI Workshop on Wikipedia and Artificial Intelligence (WIKIAI'08), Chicago, I.L. (2008)

[14] Nicolas Marie, Fabien Gandon. Survey of linked data based exploration systems. IESD 2014 - Intelligent Exploitation of Semantic Data, Oct 2014, Riva Del Garda, Italy. 2015

[15] Nicolas Marie, Fabien Gandon and Myriam Ribière, Florentin Rodio, "Discovery Hub: on-the-fly linked data exploratory search", I-Semantics 2013, Sep 2013, Graz, Austria. http://dl.acm.org/citation.cfm?id=2506185.

[16] Prateek Jain, Pascal Hitzler, Amit P. Sheth, Kunal Verma, Peter Z. Yeh, "Ontology Alignment for Linked Open Data", The Semantic Web – ISWC 2010 - Lecture Notes in Computer Science Volume 6496, 2010, pp 402-417. http://dl.acm.org/citation.cfm?id=1940308

[17] Roberto Mirizzi, Tommaso Di Noia, Vito Claudio Ostuni, Azzurra Ragone. Linked Open Data for content-based recommender systems. Politecnico di Bari – Via Orabona, 4, 70125 Bari, Italy

[18] S. Castano, A. Ferrara, and S. Montanelli. inwalk: Interactive and thematic walks inside the web of data. In EDBT, pages 628–631, 2014.

[19] Sherzod Hakimov, Salih Atilay Oto, Erdogan Dogdu, "Named Entity Recognition and Disambiguation using Linked Data and Graph-based Centrality Scoring", SWIM 2012, May 20, 2012, Scottsdale, Arizona, USA, pp 2-4. http://dl.acm.org/citation.cfm?id=2237871.

[20] Vania Dimitrova, Lydia Lau, Dhavalkumar Thakker, Fan Yang-Turner, Dimoklis Despotakis. Exploring Exploratory Search: A User Study with Linked Semantic Data. 2013. http://dl.acm.org/citation.cfm?id=2462199