Context-aware Search in Dynamic Repositories of Digital Documents

A. M. Khattak[a], *N. Ahmad*[b], *J. Mustafa*[b], Z. Pervez[c], *K. Latif*[b] *S. Y. Lee*[a]
[a] *Department of Computer Engineering, Kyung Hee University, Korea*
{asad.masood, sylee}@oslab.khu.ac.kr

[b] *School of Electrical Engineering and Computer Science, NUST, Pakistan*
{nabeel.ahmad, jibran.mustafa, khalid.latif}@seecs.nust.edu.pk

[c] *School of Computing, University of the West of Scotland, United Kingdom*
zeeshan.pervez@uws.ac.uk

***Abstract* - Autonomous and Distributed repositories containing digital documents are maintained and managed independently in accordance to organization's business needs. Documents containing same information in different repositories maybe represented differently, making it hard to retrieve desired information. The information explosion necessitates efficient techniques to unearth the lump of information from hay stack of online digital documents with same and heterogeneous structures. Keyword based information retrieval techniques help in improving the recall of user query result, but has a low precision. To improve precision, we adopt semantic information retrieval technique from digital documents using ontology and maintain dynamic and evolving domain ontology to accommodate the retrieved information. We followed searching technique using thematic similarity approach to enhance the precision of search results. We propose a comprehensive architecture for semantic based information retrieval and search. Plain text is read semantically and the extracted metadata is stored for later use to answer user queries. Triple-centric technique is used for maintaining source metadata (in case of system crash) and probing user queries for capturing the context of the keywords. Semantic based information retrieval and annotation technique precision and recall results are very promising. Semantic search using thematic similarity approach proves to have better precision and recall than previous keyword based searching techniques.**

Keywords: Information Storage, Information Retrieval, Knowledge Management Applications, Text Annotation, Ontology Evolution

I. INTRODUCTION

Online digital repositories are increasing in size and they amounted to terabytes of size. The pace of increase in the size is more than the Moore's Law. Searching a digital document and their contents in such huge repositories for given user query is a tedious job, and also consumes lots of time and resources. Traditional keyword-based search engines perform keyword searching in documents by matching keywords specified in user queries. These systems also maintain a word index to accomplish searching [1, 2] (e.g., Google).

The existing systems do not understand the meanings of the keywords (i.e., semantics). Semantic heterogeneity exists among different documents containing same information with different representation, which makes it more difficult to understand semantics of the keywords [3, 4, 5, and 6]. Searching such repositories for documents result in high recall but the precision is very low. To understand the meaning of terms in a digital document, first we need to extract it semantically. After semantic extraction and annotation generation, the concept representation is very important so that matching algorithm can work on it properly. This approach utilizes lexicon and domain ontology in extracting concepts from digital documents and identifies the defined relationships in the domain ontology. Ontology is a formal, explicit specification of a shared conceptualization [7].

Ontology change management is a complicated and multifaceted task incorporating, Ontology Integration, Merging, Versioning, and Evolution that deal with different aspects of ontology change management problem [8]. Domain ontology containing extracted metadata changes/evolves as new information is extracted from the documents. Consequently, an ontology change management solution has to answer a number of questions [9]. First question is posed to the systems' overall working, "how to maintain all the changes in a consistent and coherent manner?" Other questions revolve around the applications of all these logged changes for the purpose of ontology recovery and understanding for the semantics of change.

Search in semantic-based information retrieval techniques is performed by interpreting the meanings of keywords (i.e., semantics) provided by domain ontology. Ontology together with instances of the class constitutes a knowledgebase. This information is stored in Jena based triple store with embedded semantics. Now the semantics of keywords are identified through the relationships between keywords by performing semantic similarity on them [3, 4, 10, 11] and the identified semantics contribute to increase the precision of search results. The proposed framework is scalable in a sense, when new documents are found; information contained in them is extracted and stored in knowledgebase which may require the evolution of knowledgebase. The detailed experimental results show that each module of proposed framework outperforms the discussed systems.

This paper is arranged as follows: Section 2 describes the existing research work in the field of semantic searching. In Section 3 we present our proposed architecture in detail. Experimental results are provided in Section 4. Finally, we conclude our findings in Section 5 and talk about the future directions.

II. Related Work

Various semantic-based techniques have been used to improve precision and recall of search engines. Ontology based system proposed in [10] uses traditional vector space model (VSM), using the term frequency (tf) and inverse document frequency (idf), to carry semantic matching between documents. The system is only restricted to inheritance relationship while inferring the domain ontology. The framework proposed in [3] improves precision by extending the semantic matching to incorporate other relationships by defining weighting scheme to assign different weights to search terms while encountering different relationships between them. The SSR Model in [12] finds similar search terms and assigns them initial weights using tf.idf. Then higher weights are assigned to the search terms located in the same semantic neighborhood. However, we argue that the probability of encountering a keyword in a document is not an adequate measure to determine that the search terms is a suitable representative for any given context. Context-dependent approaches [13] extended VSM using tf.idf method to obtain weighted taxonomy to demonstrate the context. They argue that their weighted taxonomy assigns high similarity score to the search terms in general context and low similarity score for a specific context.

In contrast to VSM, the feature based method computes similarity between concepts by exploring their properties or features. The method employed in [4] uses normalized form of Tversky's model [14] to measure the similarity between concepts, based on their features that include synonyms, semantic neighborhood, attributes and parts. To determine the concept to be a representative of given context requires concentrating on its thematic behavior (i.e., the concept and its relationship in the text).

Systems using semantic distance based approach also exist, mostly used in graph matching such as [11, 15, and 16]. The idea is to define the distance between two concepts with respect to their position in the concept or relationship hierarchy (from their closest common parent). The basic intuition in conceptual graph (CG) matching is to calculate semantic matching by comparing arcs. The arc in CG comparison enables to determine the concept to be a representative of given context by concentrating on its thematic behavior (i.e., the concept and its relationship in the text). We inherent the same thematic behavior while performing semantic matching between the search terms in digital documents and domain ontology using RDF triples. The domain ontology is populated with the triples extracted from the digital documents that are parsed semantically.

III. Searching Digital Documents

In this section we will discuss architecture for a search engine that intelligently searches digital documents from distributed document repositories. The detailed component wise introduction is published in [17], while here we mainly focus on components working and their results. Our assumption is that the documents have already been crawled from distributed repositories. Information contained in these documents is extracted and stored in Jena based triple store where it is searched intelligently. Figure 1 shows the overall framework architecture of our system. The architecture comprising of three main modules: (a) Knowledge Extractor, (b) Ontology Change Management, and (c) Search Module.

A. Knowledge Extractor

Figure 1 (a) shows our knowledge extractor module. In this module, semantically enriched context-aware metadata of a document is generated. The metadata extraction steps are as follows.

Standard Document Format: Since digital documents are unstructured and in heterogeneous data formats. The first step is to transform the unstructured and heterogeneous data formats into one semi-structures format to be processed further, such as, XML, which ultimately improves the machine readability of the documents. After this the standard format is processed component wise as explained below.

Component Identification: Every document comprises of various components, for instance, research documents comprises of: abstract, background, and conclusion. Rhetoric Structure (RS) [18] ontology defines components of scientific publications. Since each component has its importance in document, based on this importance weights are assigned to components.

Term Extraction: In this step, terms are extracted from each component of a document and are then categorized as: 1) Single Word Terms, 2) Compound Word Terms. The single word terms are easily extracted, while compound word terms are extracted using the stop-word based mechanism that consider the words as compound word until a stop-word is reached. TF/IDF [19] of the terms is computed with respect to documents and its components.

Lexical Analyses: Job of this module is to:

- *Parts of Speech Tagging (POS):* POS tagging (a) assigns POS to each term based on thesaurus, and (b) discards unwanted POS except nouns and verbs.
- *Stemming:* is a process for reducing inflected (or sometimes derived) words to their stem, base or root form – generally a written word form. In addition, we maintain a list of suffixes combinations that need special treatment as if they are trimmed off, the context of the concept change. Some of the succeeding element are "ing", 's', "ers". In other word, it helps in identifying concepts in the documents.
- *Concept Clustering:* A document may contain multiple instances of a concept. After stemming process multiple instances of a concept are clustered on the bases of document. The clustering generate a three dimensional graph with its axis as documents, concepts and their respective frequency.

Weight of each concept is calculated with respect to the component in the document and is calculated as:

$$WC_i = \frac{tfidf_i \times (\sum n \times c_p)}{\sum N} \quad \text{---------} \quad (1)$$

Where *WCi* is total weigh of concept, *tfidf* is term frequency / inverse document frequency, n is total no of concepts, *Cp* is weight of the component and N is size of the corpus.

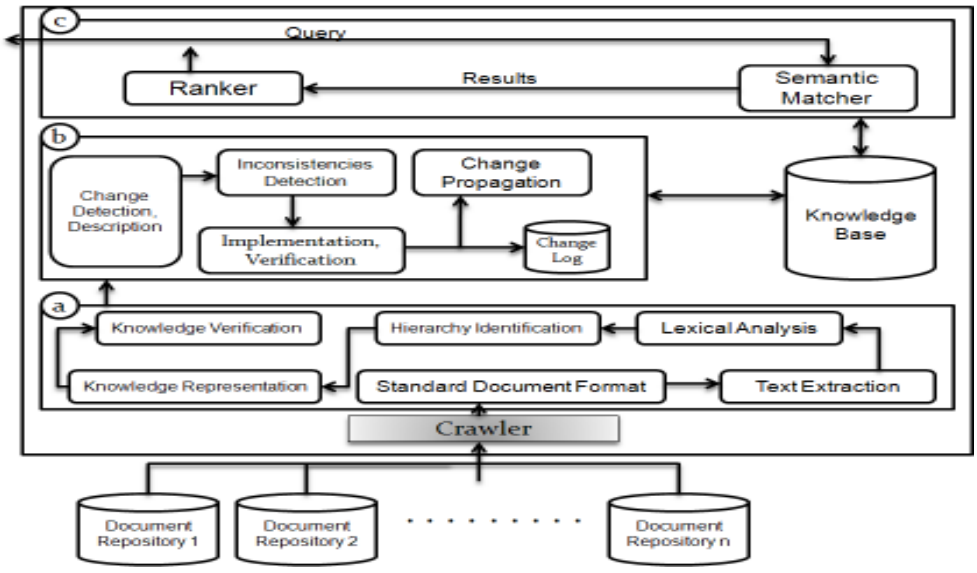

Figure 1, System architecture of intelligent search in digital documents [17].

Hierarchy Identification: After concepts clustering, synonyms for each concept are identified. OntoWordNet is used for this purpose, using this; the broader and narrow concepts are also identified to form a hierarchy.

Knowledge Representation: The identification of narrow and broader concepts helps in specifying the super-class and sub-class relationship. When the relationships are identified, a semantic structure is generated and expressed in form of triple.

Knowledge Verification: After knowledge representation in the form of triples, then the consistency and quality of the semantic knowledge has to be verified. The SemRef and SemEval [18] are used to ensure the quality of the semantic metadata.

We utilize the syntactic and semantic measures to increase the quality of the metadata. The advantage over tradition metadata is that we use the Rhetoric Structure for correlating the concept – concept relationship with the document component which improves the preciseness and accuracy.

B. *Change Management*

Ontology is *formal description of shared conceptualization of a domain of discourse*. They evolve with the passage of time, when the perspective under which the domain is viewed has changed [20]. The evolution process deals with the growth of the ontology i. e., *modifying or upgrading ontology when there is certain need for change or there comes a change in the domain knowledge.*

Ontology change management module as given in Figure 1 (b), have two basic operations:

- *Ontology Enrichment:* When the knowledge extractor produces totally new changes (concepts and slots), which are not present in domain ontology (knowledgebase), and then these are added to the domain ontology. So we enrich our ontology to accommodate the new changes and also populate our ontology for its instances.
- *Ontology Population:* When we get new concept(s) i.e., simple or aggregated, which is already present in the ontology. Then only instance of this concept(s) is introduced and the knowledgebase is populated.

After receiving the extracted metadata from the knowledge base, this module enriches and populates the knowledgebase. Input for this process is domain ontology, and the emerging concepts (local ontology of digital documents) received from Knowledge Extraction module. Working of this module is:

Change Detection and Description: The first step in the process is to detect new changes. Schema and individual level differences are detected effectively, as reported in [21]. In case, the concept in focus is totally new, then H-Match algorithm [22] is used. Its Semantic Affinity measure provides the contextual matching facility through set of four models: surface, shallow, deep, and intensive. It takes the new concepts for addition and domain ontology as input and returns the best matching concept in the ontology. The taxonomic position identification for the concept is given in [23].

After this, every identified change is represented in a proper and consistent format, where these changes may be atomic or composite. Changes are first assembled in a sequence, and then this sequence is followed for implementing the changes on domain ontology. Process of atomic changes is followed that consider all composite changes as an ordered sequence of atomic changes as well. Finally, the changes are represented using Change History Ontology (CHO) [9]. We use the same representation for logging the ontology changes in the Change History Log (CHL).

If, there is no new changes received from the Knowledge

Extractor, then this module forward the extracted information to change propagation module, which populate the instances in knowledgebase.

Inconsistencies Detection: Here ontology changes are resolved systematically to ensure that consistency of the ontology is not lost. Ontology may become inconsistent because of the new changes. Types of inconsistencies can be, 1) syntactic: when undefined entity at ontology or instance level is used, 2) semantic: when meaning of entity is changed due to performed changes. To keep ontology consistent, deduced changes are introduced. A complete request of, required changes and the deduced changes are made. KAON API [24] to identify alternative deduced changes. These changes are presented to the ontology engineer and then ontology engineer selects changes from the available alternatives.

Change Implementation & Verification: All the induced and deduced changes, which make a complete change request, are applied to the ontology. This module is designed to manage three characteristics. 1) When a change is applied then it should complete in isolation, must be atomic, durable, and consistent. 2) After every change implementation, change verification is made to verify that the required changes have been committed to the ontology. 3) After every change implementation the change must be logged in the change log, to keep track of all the implemented changes in an ordered manner. This helps in undoing changes by reversing the logged changes on ontology.

Change Propagation: It has two basic operations. 1) If there are no new changes in the local ontology, verified by Change Detection and Description module, then the instances of local ontology are simply populated in the knowledgebase. 2) If new changes were requested and also implemented through the above steps, then these changes are propagated to the domain ontology and it evolves to a new state. The instances of local ontology are populated in the knowledgebase according to the evolved domain ontology.

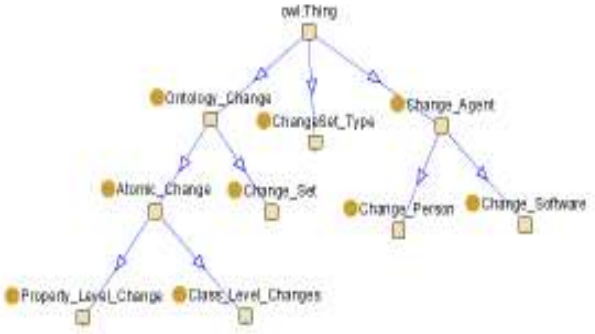

Figure 2. Snapshot representing core classes of Change History Ontology (CHO)

Change History Log (CHL): It is a repository that keeps track of all the changes made to the domain ontology. It stores every change after it is being implemented in change implementation phase. CHL is also required for reversibility purpose when an ontology engineer want to undo or redo some of the changes then this log is accessed and changes are simply reverted. The log uses Jena based triple store and the change description is provided by Change History Ontology (CHO) shown in Figure 2, to preserve the changes for later use. The baseline for CHO is the Log Ontology presented by Yaozhong David Liang [25]. We have modeled quite a few extensions to Log Ontology and come up with CHO as shown in Figure 2. Some of the extensions include: 1) Capturing such provenance information as the change author, reason, timestamp. 2) We introduced a class *OntologyChange*. It has further subclasses including *AtomicChange* and *ChangeSet*. The *AtomicChange* tackles with all types of changes that can be applied to ontology at its class and property, and Instance level, which are addition, deletion, updating, and renaming. 3) *ChangeSet* holds information about the changes that whether it is an instant or composite and stretched change over a defined time interval. *ChangeSet* also helps in properly maintaining the sequences of the changes applied. With *ChangeSet*, all the changes of some defined time interval are organized and managed together, which help us to undo/redo the changes.

C. Document Searching

In this module (shown in Figure 1 (c)), we employ RDF triples instead of keywords searching in order to concentrate on the context of the search term. A user submits search query in the form of RDF triple(s), which is answered using knowledgebase. The user submitted query is parsed into Spar-QL query. The search query is analyzed by the semantic matcher, to generate one or more standard Spar-QL queries. The searching is performed using the extended queries to identify the relevant documents for the user query. The relevance score of documents is computed by Ranker to sort the identified documents. This module has two main components shown in Figure 1 (c): Semantic Matcher and Ranker.

Semantic Matcher: In Search Module, the semantic matcher is the main component which is responsible for query answering/searching. In first step, the RDF query is augmented with synonym [3, 12, 16], semantic neighborhood [4] and other relationships such as hyponym [4, 12, 26] (i.e., Is-A relationship) and Meronym (i.e., Part-of) [4, 12]. The equivalent Spar-QL queries are generated for augmented RDF triples to perform searching. We have used different semantic properties to relate different concepts.

- *synOf:* The *synOf* property states that different individuals are the same (i.e., equivalence relationship). It is used to deal with synonyms, acronyms, and lexical variant heterogeneity issues. For example *<author> synOf <writer>*.
- *neighborOf:* The *neighborOf* property is used to explore the semantic neighborhood of concept's relationships. The semantic neighborhood (n) of a concept c is the set of the set of Cs whose distance d to the concept c is less than or equal to a non-negative integer r, called radius of the semantic neighborhood. The semantic neighborhood with radius r =1, represents subclass, super class and part-whole relationships.

$$n(c,r) = \{C_s\} \ \ such that \ \forall \ d(c,C_s) \le r \qquad (2)$$

In second step, similarity between query and source RDF triple is computed using distance based approach [11, 15, and 16].

Semantic Similarity: Interpreting the keywords with respect to the context requires obtaining similarity between concepts and relationships (i.e., thematic similarity). We focus on thematic similarity by matching RDF triples to concentrate on both aspects together.

- *Concepts Similarity:* Concepts similarity is measured by calculating the distance between them [11, 15, 16]. The distance is calculated between different concepts from concepts position in the hierarchy. The position of a concept in a hierarchy defined in [16] is as follows.

$$milestone \quad (n) = \frac{1/2}{k^{\,l(n)}} \tag{3}$$

Where 'k' is a predefined factor larger than 'one' that indicates the rate at which the value decreases along the hierarchy, and $l(n)$ is the depth of the node 'n' in hierarchy. For the root of a hierarchy, $l(root) = 0$. For any two concepts in the hierarchy, have closest common parent (*ccp*). The distance between two concepts c_1, c_2 and their *ccp* will be determined by their closest common parent as follows:

$$d_c(c_1, c_2) = d_c(c_1, ccp) + d_c(c_2, ccp) \tag{4}$$

$$d_c(c_1, ccp) = milestone(ccp) - milestone(c_1) \tag{5}$$

Thus, the similarity calculation between two concepts, c_1 and c_2, as follows:

$$sim_c(c_1, c_2) = 1 - d_c(c_1, c_2) \tag{6}$$

There are some exceptions that if the concept c_1 and concept c_2 are synonym or acronym of each other, the distance will be set to zero, i.e., the similarity between these two concepts will be 'one'. We consider synonym and acronym relation between concepts are at the same level.

- *Relationship Similarity:* Likewise, the similarity between two relationships is defined as follows:

$$sim_r(r_1, r_2) = 1 - d_r(r_1, r_2) \tag{7}$$

The distance between two relations is also calculated by their respective positions in the relation hierarchy. The only difference is that the relation hierarchy is constructed manually by us. There are some exceptions that if relations r_1 and r_2 are *synonym* or *acronym* of each other than the distance will be set to *zero*, consequently the similarity between these two relations will be 'one'.
- *RDF Triple Similarity:* The user query and data source RDF triples are matched to find their similarity. The final triple similarity matching formula by combining (6) (for concepts similarity) and (7) (for relations similarity) is as follows:

$$sim(q,s) = \prod_{i=0}^{n} \prod_{j=0}^{m} \begin{matrix} sim_{sub}(q^i_{sub}, s^j_{sub}) \\ sim_r(q^i_r, s^j_r) \\ sim_{obj}(q^i_{obj}, s^j_{obj}) \end{matrix} \tag{8}$$

Where q_{sub} , q_{obj} and S_{sub} , S_{obj} are matched concepts while q_r and S_r matched relation of query RDF triple q and source RDF triple S respectively. *Sim(q, s)* is the overall similarity between query *(q)* and source (*S*) RDF triples. Here i and j represent i^{th} and j^{th} subject or object or relation of query and source RDF triples respectively.

Semantic Reasoning: To infer knowledge from existing metadata (knowledgebase), different rules have been defined in the respective rule bases. Some of the rules are: *inverseOf* and *transitiveOf*.

- *InverseOf Rule:* The *inverseOf* rule defines the relation taken '*backwards*'. If two concepts c_1 is related to c_2 with relation R then c_2 will be related to c_1 with R^{-1}. Figure 3, shows the N3 representation of *inverseOf* rule in semantic web rule language (SWRL).

```
:Def-inverseOf @swrl("(?x neighborOf ?y) -> (?y  neighborOf ?x)").
```

Figure 3. A snippet of' inverseOf rule

- *TransitiveOf Rule:* The *transitiveOf* rule defines if c_1 is related to c_2 and c_2 is related to c_3 with relation R then there exist relation R between c_1 and c_3. Figure 4 shows the N3 representation of *transitiveOf* rule in SWRL.

```
:Def-transitiveOf @swrl("(?x neighborOf ?y) (?x synOf ?w) (?y synOf ?u) ->
(?w neighborOf ?u)").
```

Figure 4. A snippet of transitiveOf rule

Documents Ranking (R(d)): Identified relevant documents through the Semantic Matcher are passed to the Ranker, which sort them according to their relevance to the users query. The ranking algorithm of the Ranker combines two factors: 1) the RDF triple score is calculated and 2) its relevance to a document indicated by W_i using equation (6). The documents relevance *R (d)* can be calculated as:

$$R(d) = \sum_{i=0}^{n} sim(q_i, s_i) \times W_i \tag{9}$$

Where n be the total number of triples in a document. The documents are ranked according to their relevance score and returned to the user.

IV. Implementation and Results

To retrieve context-aware information from digital documents in the repositories, we have implemented all the three modules of our framework architecture and verified their results. There is no such system available that claim all the features our system provides so that we can compare our proposed system with. But there are individual components been developed by different researchers, so we compare our system component wise. We also discuss the results of our

three modules with different data sets that we used to validate their functionality.

A user defined data set has been prepared for the evaluation of the Knowledge Extractor module because standard data set is not available for system testing and evaluation in the specified domain. The statistics of the data set are shown in Table 1. The system evaluation has been divided into three units: 1) section detection and segregation, 2) compound word extraction and, 3) semantic annotation generation. Here we discuss the semantic annotation generation results (end triples generated by Knowledge Extractor module) against existing systems.

TABLE 1. DATA SET SPECIFICATIONS

No of documents	20
Average document size	5 – 7 pages
Average sections per document (physical)	5-7
No of corpses	2 (10 document each)

In semantic annotation generation, Semantic Annotation Platforms (SAP's) generally follow the standard evaluation criteria of annotation precision, annotation recall, and annotation f-measure details given in [27]. Here, the proposed system is compared with the existing systems such as Onto–O–Mat: PANKOW [28], SemTag [29], AeroDML [30], and KIM [31]. The results comparison of proposed system with the above mentioned systems is shown in Figure 5, while their detail working comparison is given in Table 2. From the graph, it is evident that AeroDAML system has more precision than our system Semantic Annotations Engine (SAE). But, our system outperforms AeroDAML in annotation recall and annotation f-measure. The main reason of AeroDAML having more annotations precision is using bootstrap ontology for the extraction of concepts from the digital documents. It associates concepts based on user knowledge not on representation in document.

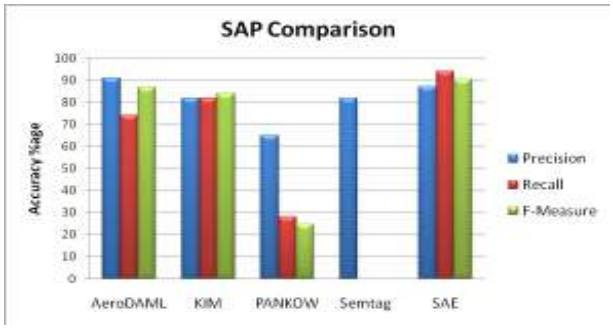

Figure 5. SAP Comparison

To validate whether the ontology change management work appropriately for the overall mechanism, we have tested it for ontology enrichment and population. To evaluate the change capturing capability of our change management module, we developed as prototype Protégé plug-in, we compared it with another plug-in i.e., *ChangesTab* of Protégé. Both the plug-ins (i.e., *ChangesTab* and *Change Tracer*) were enabled in Protégé and changes were made to ontology (*Documentation* Ontology). 35 random changes were made to the *Documentation* ontology covering all the four different categories (i.e. Change in Hierarchy, Change in Class, Change in Property, and Other Changes). Out of these 35 changes, *ChangesTab* of Protégé was able to capture 26 changes while our plug-in i.e., *Change Tracer* captured 31 changes. The graph representing these results is given in Figure 6; where y-axis represents the number of changes captured and x-axis represents the number of changes made. The experiment was repeated several times and *Change Tracer* showed better results than *ChangesTab*.

TABLE 2. COMPARISON OF EXISTING SYSTEMS AGAINST SAE

Platform	Method	Machine Learning	Manual Rules	Bootstrap Ontology
AeroDAML [30]	Rule	N	N	Wordnet
KIM [31]	Rule	N	Y	KIMO
Onto –o – Mat: PANKOW [28]	Pattern Discovery	N	N	User
SemTag [29]	Rule	N	N	Tag
SAE	***Rule***	***Y***	***Y***	***No***

Results for the recovery of crashed or changed ontology is provided, where validation and verification of the outcome of the recovery process is an essential and critical aspect. There has to be a mechanism to prove the hypothesis that the output ontology, after applying the recovery process on top of the Change History Ontology (CHO) [9], is correct. In order to quantitatively measure the performance of the recovery algorithm, an evaluation measure has been used and we have published it in [32]. The Ontology Metadata Vocabulary (OMV) is used by the community for better understanding of the ontologies for the purpose of properly sharing and exchanging the information among organizations. To achieve this goal, this standard is set and agreed by the community for sharing and reusing of ontologies. OMV has different versions available online containing different set of concepts, properties, and restrictions. We have tested our recovery algorithm on three different versions of OMV[1] (i.e., omv-0.6.owl, omv-0.7.owl, and omv-0.91.owl).

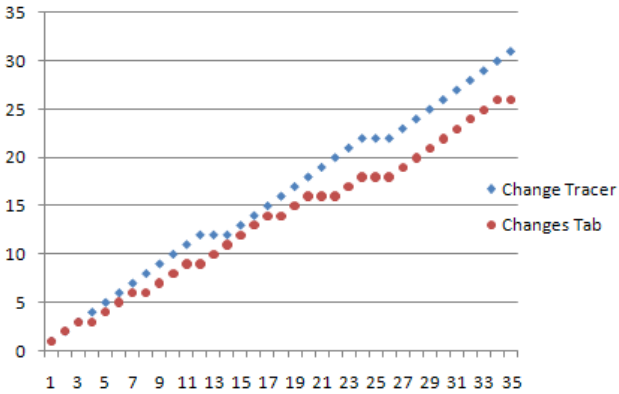

Figure 6. Comparison of Change Tracer against ChangesTab of Protégé

[1] http://ontoware.org/frs/?group_id=39

Table 3 shows details about the types and number of changes among different versions. These changes are captured and stored in CHL with the help of *Change Tracer* [32]. Using these logged changes we applied the roll back and forward procedure given in [32], which resulted in recovered versions. The recovered versions were compared with the original versions using *Prompt* Tab in Protégé and also checked manually. After the comparisons, all the recovered versions were found exact with the original versions.

TABLE 3. NUMBER AND TYPES OF CHANGES AMONG DIFFERENT VERSIONS OF OMV ONTOLOGY

Ontology Versions	**OMV.owl & OMV-0.7.owl**	**OMV-0.7.owl & OMV-0.91.owl**
Total Changes	**38**	**189**
Change in Hierarchy	18	71
Change in Classes	6	34
Change in Properties	25	123

The prototype system for semantic search takes different queries as shown in Figure 7. Query_1 is simple, so precision of all the systems on this query is quite high, whereas Query_2 is not simple so the precision of VSM is quite low. The precision of intelligent search is better than VSM but expanding the context-aware search in semantic neighborhood shows improved results as compared to other approaches with respect to precision and recall. The proposed intelligent searching approach has better precision than VSM and simple semantic-based searching. The improvement in recall is also observed in our approach by expanding concepts in semantic neighborhood.

Query_1: *Show all IEEE conference paper by 'levi' ?*
RDF Query Pattern:
(?p :writtenBy :Levi)
(?P :hasType :Conference)
(?p :hasPublicationOrganization :IEEE).

Query_2: *Find papers about the use of ontologies in Data Integration in the year 2005?*
RDF Query Pattern:
(?p :hasContent :ontologies)
(?p :isAbout :Data Integration)
(?p :hasPublicationYear :2005).
….

Figure 7. Different Query examples in RDF

Figure 8, shows the comparison graph between the results of VSM, context-aware search and context-aware search with semantic neighborhood (our system). The experiment is carried out with a collection of manually classified 633 documents, related to master's research thesis, each containing 50-100 RDF triples. The research publication ontology, containing 719 concepts, is used in the experiment as the domain knowledgebase. Table 4 shows the comparison between the other principal approaches used by the existing systems and our proposed approach. The comparison shows that our approach significantly improves the precision by introducing context awareness but recall may vary (i.e., either low or high).

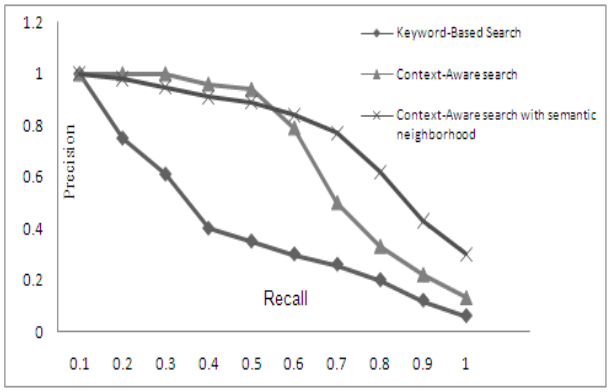

Figure 8. Precision-Recall illustration of VSM, Context Aware & Context-Aware Semantic with semantic neighborhood Approach

TABLE 4. COMPARISON BETWEEN PRINCIPAL MEASURE AND OUR APPROACH

IR Systems	**Similarity Method**	**IR Type**	**Precision***	**Recall***	**Context Awareness**
DOSE [10]	VSM	Keyword-Based	Low	High	No
IR Framework [3]	VSM	Semantic-RDF Based	Average	High	No
SSRM [12]	Hybrid	Semantic-Keyword Based	Average	Average	No
Context-Dependent [13]	VSM	Semantic-Keyword Based	Average	High	Yes
Our Approach	Hybrid	Semantic-RDF Based	High	Average	Yes
**Precision & Recall : High >0.65, Average>=0.35 and <=0.65, Low<0.35*					

V. CONCLUSION

In this research articles, we presented architecture for context-aware search engine for digital repositories, based on the semantic metadata retrieved from digital documents. RDF triples are used instead of keywords matching technique. The subject, property and object of RDF triple enabled the search module to concentrate on the combination of concepts and their relationship similarity at the same time. We maintain knowledgebase to store metadata using domain ontology. In semantic matcher, user submitted query is expanded with *synonym* and *semantic neighborhood*. It is then rewritten for generated search terms. The proposed architecture resulted in improved precision of search results.

Currently, we are focusing on metadata retrieval from digital documents, knowledgebase maintenance, and semantic search module for achieving good precision. In future we are going to extend the architecture, so that it can handle other data structures like graphs, tables, instead of only text from research articles.

ACKNOWLEDGMENT

This research was supported by the MSIP (Ministry of Science, ICT & Future Planning), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency)" (NIPA-2013-(H0301-13-2001)).

REFERENCES

[1] X. Li, F. Bian, H. Zhang, C. Diot, R. Govindan, G. Iannaccone. "MIND: A Distributed Multi-Dimensional Indexing System for Network Monitoring". IEEE Infocom-06 Barcelona, April 2006.

[2] A. Ntoulas, G. Chao, J. Cho, "The Infocious Web Search Engine: Improving Web Searching Through Linguistic Analysis", International World Wide Web Conference Committee (IW3C2) ACM, Chiba Japan, May 2005.

[3] W. D. Fang, L. Zhang, Y. X. Wang, S. B. Dong, "Towards a Semantic Search Engine Based on Ontologies", IEEE Preceedings of the Fourth International Conference on Machine Learning and Cybernetics Guangzhou China, pp. 1913- 1918, 18-21 August 2005.

[4] M. A. Rodriguez, M. J. Egenhofer, "Determining Semantic Similarity among Entity Classes from Different Ontologies", Knowledge and Data Engineering, IEEE Transactions, vol. 15, Issue 2, pp. 442- 456, April 2003.

[5] C. Y. Lee, V. W. Soo, "Ontology based information retrieval and extraction", 3rd International Conference on Information Technology: Research and Education IEEE, pp. 265-269, 27-30, June 2005.

[6] M. Uschold, M. Gruninger, "Ontologies and Semantics for Seamless Connectivity", SIGMOND, vol. 33, December 2004.

[7] T. Gruber, "A translation approach to portable ontology specifications. Knowledge Acquisition", pp. 199-220, 1993.

[8] S. Castano, A. Ferrara, G. Hess, "Discovery-Driven Ontology Evolution". The Semantic Web Applications and Perspectives (SWAP), 3rd Italian Semantic Web Workshop, PISA, Italy, 18-20 December, 2006.

[9] A. M. Khattak, K. Latif, and S. Y. Lee, "Change Management in Evolving Web Ontologies", Knowledge-based Systems, ISSN: 0950-707051, Available online, June 6, 2012

[10] D. Bonino, F. Corno, L. Farinetti, A. Bosca, "Ontology Driven Semantic Search", WSEASTransaction on Information Science and Application, Issue 6, vol. 1, pp. 1597-1605, December 2004.

[11] S. Khan and F. Marvon, "Identifying Relevant Sources in Query Reformulation". In the proceedings of the 8th International Conference on Information Integration and Web-based Applications & Services (iiWAS2006), Yogyakarta Indonesia, December 2006.

[12] G. Varelas, E. Voutsakis, P. Raftopoulou, "Semantic Similarity Methods in WordNet and their Application to Information Retrieval on the Web", 7th ACM international workshop on Web information and data management November 5, 2005.

[13] E. Sayed, A. Hacid, H. Zighed, Djamel, "A New Context-Aware Measure for Semantic Distance Using a Taxonomy and a Text Corpus", IEEE International Conference on Information Reuse and Integration, 2007 (IRI-07), pp. 279-284, 13-15, August 2007.

[14] A. Tversky, "Features of similarity", Psychological Review, Vol. 84(4): pp. 327-352, 1977.

[15] M. Montes-y-Gomez, A. Lopez, A. Gelbukh, "Information Retrieval with Conceptual Graph Matching", DEXA-2000: 11th International Conference and Workshop on Database and Expert Systems Applications, Greenwich, England, September, 2000.

[16] J. Zhong, H. Zhu, J. Li, Y. Yu, "Conceptual Graph Matching for Semantic Search", Proceedings of the 10th International Conference on Conceptual Structures: Integration and Interfaces table of contents, Springer-Verlag London, pp. 92-196, 2002.

[17] A. M. Khattak, J. Mustafa, N. Ahmed, K. Latif, S. Khan: "Intelligent Search in Digital Documents". IEEE/WIC/ACM International Conference on Web Intelligence, pp-558-561, December, 2008.

[18] T. Groza, S. Handschuh, K. Moller, S. Decker. "SALT: Semantically Annotated LATEX for Scientific Publication", ESWC, 2007.

[19] M. Shepherd. "Extracting Meaningful Metadata". DRTC Workshop on Semantics, 2003.

[20] N. F. Noy and M. Klein, "Ontology evolution: Not the same as schema evolution", Knowledge and Information System, vol. 6, no. 4, pp. 428–440, 2004.

[21] M. Tury, M. Bielikova. "An Approach to Detection Ontology Changes". First international workshop on adaptation and evolution in web systems engineering (AEWSE), 2006.

[22] S. Castano, A. Ferrara, and S. Montanelli. "Matching ontologies in open networked systems". Techniques and applications, Journal on Data Semantics (JoDS), vol. V, pp. 25-63, 2006.

[23] S. Castano, A. Ferrara, and S. Montanelli, "Evolving open and independent ontologies," Journal of Metadata, Semantics and Ontologies (IJMSO), vol. 1, No.4 pp. 235 - 249, 2006.

[24] T. Gabel, Y. Sure, and J. Voelker. "KAON – ontology management infrastructure". D3.1.1.a, SEKT Project: Semantically Enabled Knowledge Technologies, March 2004.

[25] Y. D. Liang, "Enabling Active Ontology Change Management within Semantic Web-based Applications". Mini PhD Thesis, University of Southampton, 2006.

[26] J. Smith, and D. Smith, "Database Abstractions: Aggregation and Generalization," ACM Trans. Database Systems, vol. 2, pp. 105-133, 1977.

[27] N.F. Noy. "Semantic Integration: A survey of Ontology-Based Approaches". SIGMOD record, Volume 33, pp 65-70, December 2004.

[28] P. Cimiano, S. Handschuh, and S. Staab, "Towards the Self-Annotating Web", in Thirteenth International Conference on World Wide Web, pp 462-471, 2004.

[29] S. Dill, N. Gibson, D. Gruhl, R. Guha, A. Jhingran, T. Kanungo, S. Rajagopalan, A. Tomkins, J. A. Tomlin, and J. Y. Zien, "SemTag and Seeker: Bootstrapping the semantic web via automated semantic annotation" in Twelfth International World Wide Web Conference, pp 178-186, Budapest, 2003.

[30] P. Kogut, and W. Holmes, "AeroDAML: Applying Information Extraction to Generate DAML Annotations from Web Pages", in First International Conference on Knowledge Capture, 2001.

[31] B. Popov, A. Kiryakov, A. Kirilov, D. Manov, D. Ognyanoff, and M. Goranov, "KIM – Semantic Annotation Platform" in 2nd International Semantic Web Conference (ISWC2003), pp 834- 849, Florida, 2003.

[32] A. M. Khattak, K. Latif, M. Han, S. Y. Lee, Y. K. Lee, H. I. Kim: "Change Tracer: Tracking Changes in Web Ontologies", International Conference on Tools with Artificial Intelligence (ICTAI), pp-449-456, November, 2009.