International Journal of Engineering Research and Technology. ISSN 0974-3154 Volume 10, Number 1 (2017)

Automated Service Discovery & classification of Web Services with Crawling in Digital Ecosystem

Vaibhav V. Sawant
PG Research Scholar, Dept. of CSE, D. Y. Patil college of Engg. & Tech., Kolhapur (MH), India.

Dr. Vijay R. Ghorpade
Principal, D. Y. Patil college of Engg. & Tech., Kolhapur (MH), India.

Abstract – Semantics based service discovery and classification for similar event will be the feature services of any crawling application in Web. Majority of web services exist without associated semantic description that becomes obstacle for semantic based crawler to function and tedious to classify. The existing service discovery approaches often published keyword matching to find web service practices. As a result many services that are relevant to a specific user service request may not be considered during service discovery. In this paper, such issues are addressed like web service discovery, classification in digital ecosystem and a framework for automated service discovery, classification and categorization of web service for digital ecosystem is implemented. Proposed system performed automated service discovery and classifies web services with improvised service categorization. Clustering is utilized for accurately classifying the web services based on service functionality. Real web traffic is crawled for experimenting & results validate the effective feasibility of the work done by evaluating the performance of crawling technique, vector algorithm, classification technique using various popular metrics from web data mining field such as Harvest rate, Precision, Recall, Fallout rate, F-measure, G-mean.

Keywords: crawler, automated service discovery, ontology, metadata classification, SVD, SVM classification, digital ecosystem.

1. INTRODUCTION

Web search engines are mostly common Information Retrieval (IR) systems. Search engine don't have any laborious task of developing the required technology & thus depends on crawler for IR. Excess of web content includes a high percentage of irrelevant and redundant information which posture unprecedented challenges for the search engines. It is quite more difficult to expect all new services for having associated semantic descriptions. In addition, the descriptions of the vast majority of already existing services do not have clear associated semantics [11]. WWW is enormously increasing with different domain web services getting added daily and is challenge for search engines to for accurate results. First challenge in evolving Semantic Web (SW) is crawling web service automatically & accurate discovery.

A digital ecosystem is a digital environment created for networked organizations that support cooperation, knowledge sharing, development of open and adaptive technologies and evolutionary business models which can further self-organize any digital infrastructure. A service provider publishes a service entity and enters into digital ecosystem environment [15]. Service entities are stored in distributed service knowledge bases. These service entities are stored in the form of service metadata. Before publishing, web service should be annotated by semantic web markup languages such as Resource Descriptions Framework (RDF), Web Ontology Language (OWL) or user defined XML.

Semantic categorization of web services will expedite service discovery by organizing similar entities together. This is not sufficient to improve the selection and matching process as they are syntactic in nature. Thus, the discovery process is constrained by its dependence on human intervention for choosing the right service based on its semantics. A focused crawler or topical crawler is a web crawler that exploits to download only particular web pages that are relevant to a pre-defined topic or set of topics. They attempt to download pages that are analogous to each other. The main issue in focused crawling lies in the context of crawler i.e. predicting the similarity of the text of a given webpage to the input provided as a query before actually downloading the page [6].

In order to resolve these two different issues, a framework for a semantic based focused crawler, with the purpose of automatically discovering, annotating and classifying the web-service in digital environment is proposed [7]. Hence, owing to the huge number of non-semantic service entities, a semantic based crawler that will automate service discovery methodology in digital ecosystem is necessary. Such methodology will grace discovery and category services in entities collection, management and retrieval in the digital service automation [13].

The rest of the paper is presented in below sections – section 2 gives related work, section 3 specifies system architecture, section 4gives implementation details, section 5speaks about experiment setup and result generation. Finally in section 6 work is summarized and concluded.

2. RELATED WORK

Existing work related to focused crawlers mainly articulate on building them using ontology or metadata [2, 16]. Ontology is employed by these crawlers depending upon the search intent in relevant domain knowledge. Thus, semantic focused crawlers can be categorized into two types, based on the search intent. One of them is specifically designed to search relevant ontology in the WWW such as Swoogle & OntoKhoj [12, 14]. They crawl ontology repositories to gather linked data (in RDF, XML or OWL format) existing on the web. The other type of the focused crawlers searches relevant web pages (documents and not ontologies) from the web by utilizing precedent metadata knowledge to arbitrate web page relevance finding its similarity measure between terms. The concepts of topical and focused crawling were first introduced by Udapure [6].

The performance of a focused crawling depends mostly on the richness of links in the specific topic being searched. A focused crawling usually relies on a general Web search engine for providing starting page called as seed.

Lakshmi proposed an ontology-based automatic annotation crawler for an association metric that optimizes the order of visited URLs for web crawlers [8]. Existing approaches to web service matching are either syntactic or semi - semantic matching, e.g., Abhas Paliwal have studied LSI to acquire the semantic associations between short textual Web Services Descriptions Language (WSDL) [11].

3. SYSTEM ARCHITECTURE

To overcome issues as seen in literature survey, a semantic based focused crawler is implemented for automating service discovery in digital web ecosystem. This crawler selects semantically similar web services using domain-knowledge in web ecosystems and improves the service retrieval result & executes at middleware of any search engine. The work is divided into following modules:

I. Implementation of semantics based crawler
II. Metadata generator & classification

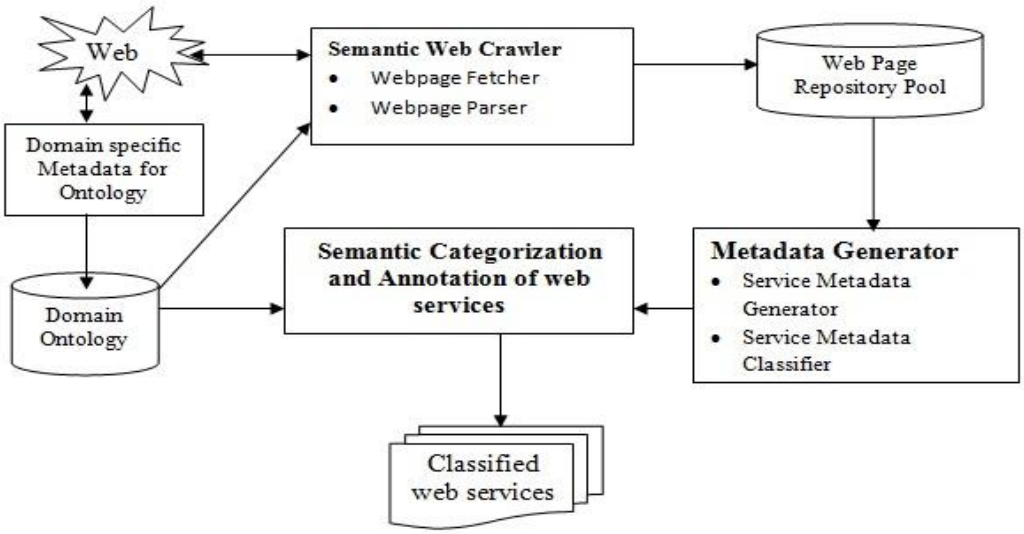


Fig. 1 System Architecture

A hybrid matching approach is implemented that combines various matching methods (e.g. syntactic and semantic) into a composite algorithm. This enables ad hoc composition of several matching approaches based on predefined criteria. The web services within the service registry have predefined categories that are specified by the service providers. As a result, similar services may be listed under different categories. Given the large number of web services and the distribution of similar services in multiple categories in the existing Universal Description Discovery and Integration (UDDI) environment, it is quite difficult to find services that satisfy the desired functionality. The most widely used IR technique constitutes the Vector-Space Model (VSM). VSM considers the syntactic aspect of term association and does not account for the underlying semantic structure [11].

III. Semantic categorization
IV. Performance evaluation

Figure 1 shows detailed architecture of proposed system. This system will work as under:

- Crawler retrieves information regarding service entities from the web, which corresponds to the functionality of service discovery in digital ecosystems.
- Crawler annotates service information with purpose of semantic and stores semantic service information, which corresponds to the functionality of service annotation in digital ecosystems.
- Crawler filters and classifies the annotated service information by means of specific service domain

knowledge, which corresponds to the functionality of service classification in digital ecosystems.

3.1 Implementation of Semantic web crawler

In this module, characteristics of domain based ontology and syntax based ontology are clubbed together and with use of depth first algorithm for crawling, website data is collected in XML files [5]. Efficient focused crawlers [9] continue visiting web pages until specified number of pages, information snippets have been downloaded or until local resources (such as storage) gets exhausted but within boundary of same domain. Sequential working of this crawling is as follows:

3.1.1 First seed URL of website along with its boundary limit such as number of pages & depth of crawl are initiated by user in GUI of application server. This depth (q) is sent via policy center to multithreaded crawler for web page crawling as shown in figure 2.

3.1.2 Webpage fetcher will start to obtain web pages in queue as it receives the URL. After crawling the desired amount of web pages, they are sent to repository pool as depicted in figure 2.

3.1.3 When the policy centre receives the URLs from the webpage fetcher, it will determine whether they are within the crawling boundary & redundancy by comparing each document domain name. After to it, Policy Centre will allot doc id to the URL of webpage by their visiting priorities and sends them to the pool. Once all web pages have been traversed, all its embedded tags are removed and page will be saved in repository pool in plain texts. This pool consists of XML documents which include information snippets enclosed inside text tag and its parent URI.

3.2 Metadata generation using semantic web crawler

Metadata generator module reinforces in creating service metadata from collected information and stores into the service metadata. Metadata will be generated in ontology concepts. Classification will be done based on predefined domains under category concepts. For e.g. Art & Humanities category have different domains like History, Dance, Languages, Films and Architecture etc. After identifying the category and referring these domains, implemented tool classify webpages.

For development of this module, ontology is generated for new as well as already collected repository documents. Lovins stemmer algorithm [17] is incorporated for pre-processing stemming step, standard stopwords libraries [18] are used for stopwords removal and further coding part is carried in Java programming using Waikato Environment for Knowledge Analysis (WEKA) core repositories. Metadata generation is necessary because information extracted from web pages is rule based and to fetch terms that appear in multiple documents in same domain may increase size of dataset. Hence to reduce the terms frequency, TF matrix is generated and metadata is classified that improves service categorization of terms according to their domain. This module worked as under:

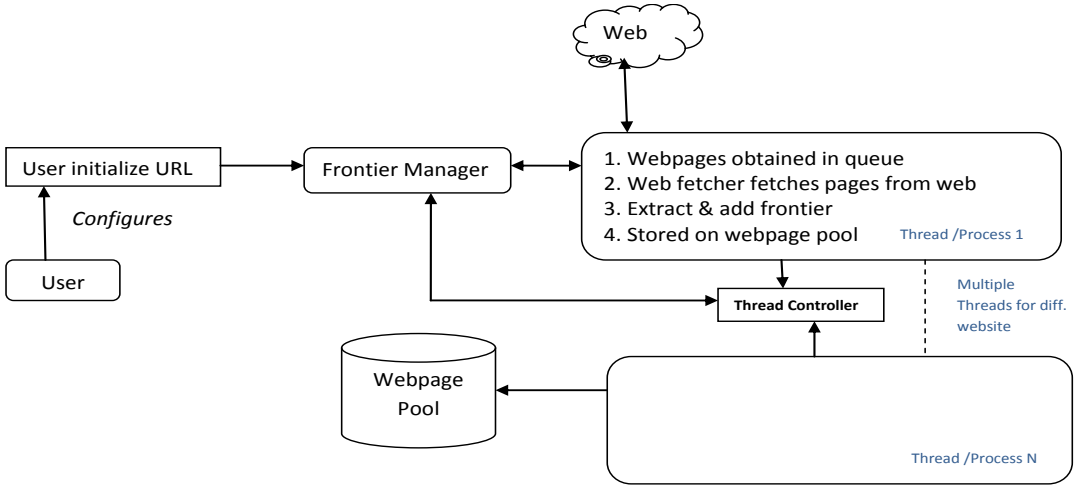


Fig. 2 Working of multithreaded web crawler

3.2.1 Ontology generation is first important step in this module because it helps to classify webpages unlike syntactic crawler were not aware of it. For this documents are collected from web repository.

3.2.2 The documents which are collected from repository are send to service metadata generator. Service metadata will be created after pre-processing. Metadata is stored into the service metadata local database..

3.2.3 Metadata classifier will compute the similarities between the generated metadata and for each bottom-level ontology concept for compatible ontology. If a similarity is above a threshold value, the corresponding concept can be regarded as being relevant to the metadata. Then, service metadata generator will associate metadata with the concepts. If similarity is below threshold those are treated as irrelevant to metadata.

Term frequency of each word in a document is a weight which depends on the distribution of each word in documents. TF matrix generation is done by using:

$$TermFrequency(TF) = \frac{\text{No. of times term t appears in a document}}{\text{Total number of terms in the document}}$$

helps metadata reduction for usage in classification process.

This method has been already used for boosting clustering of high dimensional data and in this work, SVD for text categorization purpose in terms of semantic metadata.

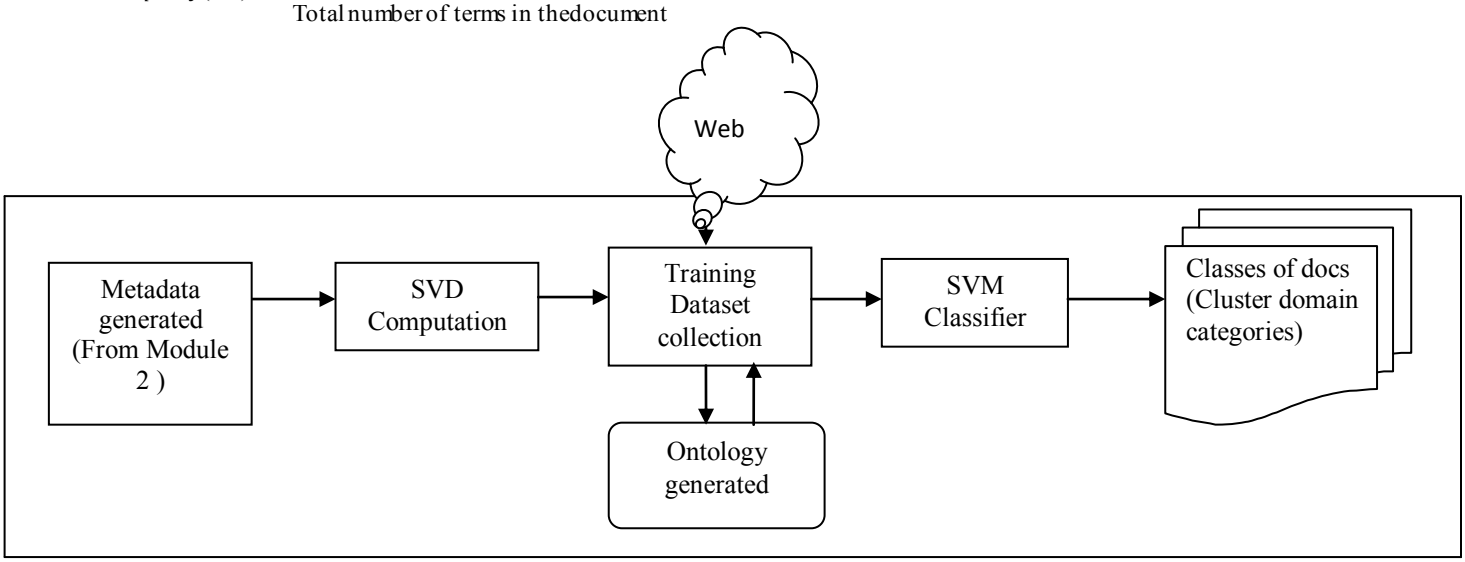


Fig. 3 Working of SVM Classification as per domain categories

3.3 Semantic categorization

This module is used to process service query and decide the overall search category of web services for the website sitemap. Classified metadata is again processed to annotate the parse page semantically, based on classification as predefined in XML files for different categories. This module is divided into two parts. Firstly, vector is decomposed and secondly, classification is done. The outcome of this preprocessing is in a term vector yielding Singular Vector Decomposition (SVD). SVD is a matrix algebra operation that is used to reduce matrix dimensionality yielding a new high dimensional abstract space. The most popular word space model based on SVD to extract semantic information from raw text is Latent Semantic Analysis (LSA), which represents the vector space as a word-by-document co-occurrence matrix [2]. After SVD matrix is computed then SVD for ontology is computed. Then similarity is processed using cosine similarity matrix. Figure 3 shows the working of semantic categorization process.

3.3.1 This module decomposes aggregated Singular Vector Decomposition (SVD) ontology and cosine similarity matrix generation. It calculates vector matrix and helps application to browse the term concepts and classify webpages in domain categories. For carrying out this work, ontology will be referred from second module, where it was created in terms of metadata.

3.3.1.1 Metadata that's get outputted is large in its size so, needs to be consolidate and reduce in size so as to utilize for crawler application [3]. SVD implemented

In ontology-based web page classification, ontology can be used as background semantic structures for web mining [1]. For e.g., instead of categorizing web pages into categories, ontology-based web page classification may classify web documents as concept instances and web page pairs as relationship instances [4]. This work is carried out with SVM technique.

In SVM classification, cluster of similar documents are formed and it uses SVD to find the semantic similarity between documents. Then the page counts based co-occurrence measures and the snippets-based lexical pattern clusters are combined into one feature vector [10] and are used to train the SVM. Training is the process of taking the content that is known to belong to specified classes and creating a classifier on the basis of that known content. SVM is trained using the existing training dataset of existing processed metadata information and web documents. The groups of training documents are classified into different classes based on the cosine similarity measure. The LibSVM is used as the SVM implementation as it can be used for multiclass classification. LibSVM libraries are open source and have many features. Some of them include:

- Efficient multi-class classification
- Different SVM formulations
- Various kernels (including precomputed kernel matrix)
- Weighted SVM for unbalanced data
- Both C++ and Java sources

In classification, for checking similarity of web pages or data content following algorithmic pseudocode [5] is implemented:

{SWS means Semantic Web Service and SVD states Singular Decomposition Vector} in below pseudo code,

```
Checksimilarity(List_of_webservices)
Begin
Get List_of_webservice {<SWS1, SDV1>, <SWS2,
 SDV2>  ..... < SWSn , SDVn>}
For each tuple do the following process

If ( SWS1.name = SWS2.name) then
   If (SWS1.Description=SWS2.Description) then
       begin
       For each input ( Ii )of SWS1.inputs
       For each input ( Ij )of SWS2.inputs
      Compare_Input ( Ii , Ij )
      For each Output ( Oi )of SWS1.outputs
      For each Output ( oj )of SWS2.outputs
      Compare_output ( oi , oj )
      Add the similar web services into a group
   End
End
```

Form this clusters of similar service categories are mapped and respected category is defined each time. Implemented work gives two fold benefits; firstly, only semantically similar results are retrieved which reduces the number of results extracted and secondly, due to improved focused searching irrelevant results is pruned.

3.4 Performance metrics considered for results

In order to evaluate the performance of semantic focused crawlers, six even indicators from the field of Information Retrieval (IR) are undertaken: harvest rate, precision, recall, F-measure, fallout rate & G-mean.

3.4.1 Harvest Rate: Harvest Rate in the information retrieval is to measure the crawling ability of a crawler, which can be mathematically represented as,

$$\text{Harvest Rate} = \frac{\text{Number of associated metadata}}{\text{Number of generated metadata}}$$

3.4.2 Precision: Precision in the information retrieval is used to measure the preciseness of a retrieval system.

$$\text{Precision} = \frac{|\{\text{Relevent docu ments}\} \cap \{\text{Retrived documents}\}|}{|\{\text{Retrived documents}\}|}$$

3.4.3 Recall: Recall in the information retrieval refers to the measure of effectiveness of a query system

$$\text{Recall} = \frac{|\{\text{Relevent documents}\} \cap \{\text{Retrived documents}\}|}{|\{\text{Relevent documents}\}|}$$

. In our case query is inputted as seed URL. Multiple seed URLs can be executed simultaneously.

3.4.4 Fallout Rate: Fallout Rate is considered for calculation of non-relevant concept associated by whole collection of irrelevant metadata.

$$\text{Fallout rate} = \frac{\text{No. of relevant \& } no.of\ non - \text{relevant metadata}}{\text{Number of non} - \text{relevant metadata}}$$

3.4.5 F-measure: F-measure is another measure that combines precision and recall in which users can specify the preference on Recall or Precision by configuring different weights.

$$\text{F} - \text{measure} = \frac{2 * \text{recall} * \text{precision}}{\text{recall} + \text{precision}}$$

3.4.6 G-Mean: This measure is used to simultaneously maximize the accuracy in positive and negative examples with a favorable trade-off.

$$\text{G-Mean} = \sqrt{\text{Sensitivity} * \text{Specificity}}$$

3.4.7 Receiver Operating Characteristics (ROC): ROC curve gives a visual indication if a classifier is superior to another classifier towards various characteristics. The area under ROC curve (AUC) is used to summarize the performance of classifier into a single metric. The larger the AUC, the better is the classifier performance.

As per the definition of Harvest rate, performance should be as more as possible but No crawler till now achieves 100%. Precision refers to the proportion of actual positive samples among all samples that are predicted as being positive while Recall is the proportion of positive samples that are identified by the classifier, which is the same as sensitivity. Fallout should be near to <1% and near to 0%, but if any cases it is more than 1% then the performance deteriorates. F-Measure is a popular measure for unbalanced data classification problems & depicts the trade-off between precision and recall. It is the harmonic mean for precision and recall to evenly evaluate. High F-measure value signifies a high value for both precision & recall. One of another metric that allows to simultaneously maximize the accuracy in positive and negative examples with a favorable trade-off is metric called the Geometric Mean (GM) [16]. So GM is use to maximize accuracy of majority samples and recall with a favorable trade-off. G-mean is tradeoff between sensitivity and specificity. If G-mean is nearly equal to one, it means that TP and TN rate are well balance. If G-mean if zero it means that all the positive samples are misclassified. From this we can conclude that if any classifier having high G-mean, it means this classifier is good than another one. For evaluation purpose our technique is been tested on two classification techniques SVD and SVDONT (SVD ontology). Both algorithms are implanted in Java with WEKA repositories and tested on different range of documents.

4. EXPERIMENTAL SET-UP & RESULTS

An application tool is developed to carry all execution tasks of specified modules. The application server tool will allow only authenticated user to login and use it. The collected web contents are displayed and separated in tabs so that all its crawled information can be viewed. For execution, real time traffic from WWW is crawled for every seed URL all available recent contents from Web are stored in repository

with the help of crawling functionality. Repository is group of XML files where contents are written with its reference URI. Whenever same name website is repeated for crawling then repository is useful for comparing & checking the content change in live web page. Probabilistic offline category dataset is taken for proving our implemented classifier for different domains with set of features. Features in our case are number of web documents.

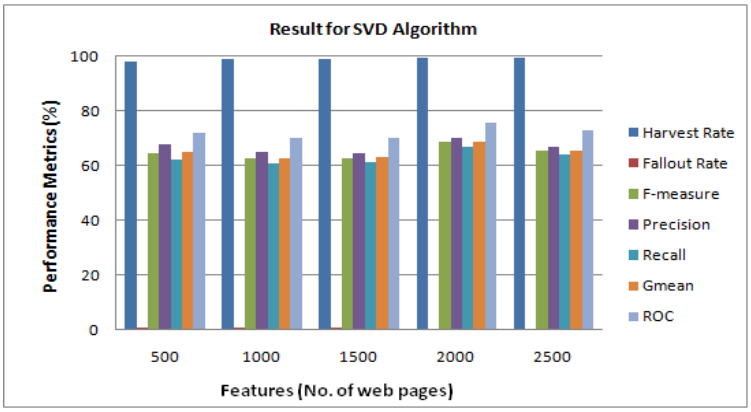
For ontology generation WEKA libraries are used and required packages are embedded with swing components [19]. Various metrics are depicted in table 1 and table 2 for SVD and SVD ontology (SVDONT). Table 1 shows the different computations as calculated referring above stated mathematical formulas for SVD. Last row in table is calculated as average of all features.

Features (No. of pages)	Harvest Rate (%)	Fallout Rate (%)	F-measure (%)	Precision (%)	Recall (%)	G-mean (%)	ROC (%)
500	98.2	0.81	64.64	67.54	61.97	64.7	71.9
1000	98.86	0.52	62.69	64.96	60.58	62.73	69.94
1500	99.01	0.47	62.79	64.53	61.15	62.83	70.03
2000	99.23	0.36	68.55	70.15	67.02	68.57	75.78
2500	99.21	0.37	65.41	66.92	63.97	65.43	72.64
Average	**98.90**	**0.50**	**64.81**	**66.82**	**62.93**	**64.85**	**72.06**

Table 1: Results analysis for SVD technique

As seen in table 1 fallout rate is less than 1% and g-mean is also 0.6485 which is acceptable (60%) as more than 0.5 and near to 1. Harvest rate is increased for more no. of feature set of pages, it means it is also more than 98% at its average. Precision is more than recall and F-measure which is taken as harmonic mean for two also is in middle of precision and recall. ROC is achieved with 72% accuracy.

fig 4:Graphical orientation of results for SVD



Now let us take a look at table no. 2 values and examine for SVDONT technique.

Features (No. of pages)	Harvest Rate (%)	Fallout Rate (%)	F-measure (%)	Precision (%)	Recall (%)	G-mean (%)	ROC (%)
500	98.99	0.41	80.17	83.77	76.86	80.24	87.46
1000	98.49	0.2	83.35	86.37	80.54	83.4	90.62
1500	99.69	0.14	87.15	89.57	84.87	87.19	94.4
2000	99.73	0.11	89.08	91.16	87.1	89.11	96.32
2500	99.75	0.1	88.96	91.01	86.99	88.98	96.19
Average	**99.33**	**0.19**	**85.74**	**88.37**	**83.27**	**85.78**	**92.99**

Table 2: Results analysis for SVDONT technique

As seen in table 2 fallout rate is less than 1% and closer to 0. G-mean is also 0.8575 which is acceptable (85%) as more than 0.5 and near to 1. Harvest rate is increased for more no. of feature set of pages, achieved with ontology more than 99% at its average. Precision is more than recall and F-measure also is in middle of precision and recall. ROC here is 92.99%. graph for same is showed in fig. 5.

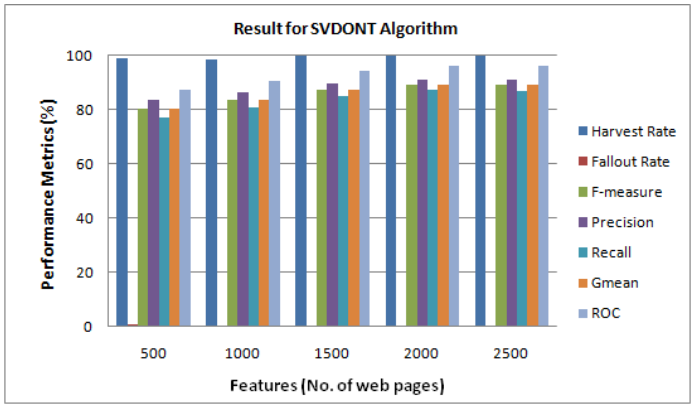


fig 5:Graphical orientation of results for SVDONT

So in comparison for fig. 4 with fig. 5, SVDONT performs better than regular SVD. Classification technique for LIBSVM proves to be effective with ontology been used in crawling. From both tables, it is seen work done is efficient and fits well in quality standards of data mining. Crawling system is effective with ontology use in discovery, classification and annotating service metadata.

5. CONCLUSION

This paper presents a implemented framework for automatic service discovery in a huge digital web ecosystem. It proves an efficient approach to build a service ontology structure and classifies the web service into cluster for user service improvisation at middleware of search engines for specifications of web 2.0. In order to achieve the goal of semantic service crawling, discovery and classification this paper evaluates work done and after comparison states to be more effective. This approach also defines a format for service metadata and service concept, which enables the

function of similarity computation and association between metadata and concepts. The main function of this work is to semantically discover the service information from the web pages by parsing, annotating, and storing their service information which is used for classification & categorization of the web service based on ontology domain knowledge.

Future work carries crawling the Deep Web as well as reckoning with Web 3.0, also future Web 4.0 concepts as a confronting task; an advancement that goes beyond the content, SW and ontology by mesh bits of content

REFERENCES

[1] Piotr Borkowski, Krzysztof Ciesielski, "Semantic classifier approach to document classification", CSIR Poland, Jan 2017.

[2] Gui Xian Xu, Chang-xhi wang, "Semantic classification method for web network", Springer - Cluster computation conference, Beijing, China, DOI 10.1007/s10586-017-0742-6, 19 Jan 2017.

[3] Weng Zang, Fan Xiao, "Using SVD on Clusters to Improve Precision of Inter-document Similarity Measure", Computational Intelligence and Neuroscience, Beijing University Volume 10, June 2016.

[4] Abhale Babasaheb Annasaheb, "Data Mining classification Techniques: A Recent Survey", IJTER Volume 4, Issue 8, August (2016)

[5] V. V. Sawant, Dr. Vijay R.Ghorpade, "Semantic based Crawling for Automated Service Discovery and Categorization of Web Services in Digital Environment", IJETER, Vol 4. Issue 7, July 2016.

[6] Udapure, T., Kale, R., & Dharmik, R., "Study of Web Crawler and its Different Types", IOSR Journal of Computer Engineering, 16(1), 2014.

[7] V. V. Sawant, Dr. V. R. Ghorpade, "Automatic Semantic Classification and Categorization of web services in Digital Environment", in ICCCT 2014, IEEE Hyderabad Section, 11-13 Dec 2014.

[8] R. Lakshmi Tulasi, Meda Sreenivasa Rao, K. Ankita and R. Hgoudar, "Ontology-Based Automatic Annotation: An Approach for Efficient Retrieval of Semantic Results of Web Documents", Springer, Singapore 2017, DOI 10.1007/978-981-10-2471-9_32.

[9] Ahmed I. Saleh, Abulwafa, "A Web Page Distillation Strategy for Efficient Focused Crawling", Applied Soft Computing Journal, doi:10.1016/2016.12.028, Dec 2016.

[10] Kavitha Chinniyan, Sudha Gangadharan, "Semantic Similarity based Web Document Classification Using Support Vector Machine", IJAJIT, March 2015.

[11] Aabhas V. Paliwal, Basit Shaafiq, Jaideep Vaidya, "Semantics-based Automated Service Discovery", IEEE Transactions on Services Computing, Vol. 5, No. 2, June 2012, pp. 260-275.

[12] Chintan Patel, Supekar K., "OntoKhoj: A semantic web Portal for Ontology Searching, Ranking and Classification", 19th ACM Conf., Nov. 2009, pp.652.

[13] J. L. M. Lastra, M. Delamer, "Semantic web services in factory automation: Fundamental insights and research roadmap", IEEE Trans. Ind. Informat., Vol. 2., Feb. 2006

[14] Li Ding, Tim Finin, Anupam Joshi, "Swoogle: A Semantic web search & Metadata Engine", DARPA, 2009.

[15] H. Boley, E. Chang, "Digital Ecosystems: Principles and semantics", in Proc. IEEE DEST, Cairns, Australia, 2007, pp. 398-403.

[16] Rushi Longadge, Snehlata S. Dongre, Latesh Malik, "Multi-Cluster Based Approach for skewed Data in Data Mining" IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 12, Issue 6 (Jul. - Aug. 2013), PP 66-73

[17] http://www.tfidf.com/

[18] www.snowball.tartarus.org/algo.../lovins/stemmer.html

[19] https://weka.wikispaces.com/Use+WEKA+in+your+Java+code