UPH Digital Library Miner: A Topic Modelling-based Software Application for Mining Document Collections of a Digital Library

Toluwase A. Olowookere
Department of Computer Science, University of Port Harcourt, Choba, Nigeria.

Ayodeji I. Fasiku
Department of Computer Engineering, Ekiti State University, Ado- Ekiti, Nigeria.

Ifeanyi C. Emeto
Department of Computer Science, University of Port Harcourt, Choba, Nigeria.

ABSTRACT

With changing user expectations, many traditional libraries are moving toward digital content storage. Accessible from anywhere at any time, digital contents as stored in digital libraries provide users with efficient, on-demand information experiences. With this trend, the amount of digital contents especially digital text documents made available to users have tremendously increased over the years, being filled with hidden information in form of the varieties of topics of discourse inherent in them leading to information overload. Accordingly, users, mostly computational researchers are presented with challenges on the discovery and identification of the varieties of topical contents of the collections in the digital library thus making it imperative to develop a means to automatically discover the topics that pervade the collections in a digital library. This paper therefore presents UPH Digital Library Miner, a software application for mining document collections of a digital library for topical structure discovery and topic-based similarities search between collection pairs, using topic modeling algorithm and inverted Kullback-Leibler divergence measure. The application is integrated with document collections built in a widely used digital library software system— Greenstone digital library system, via loose-coupling integration approach. Results obtained from using this software application on the Greenstone's document collections that contain abstracts of about 628 documents from IEEE transactions on Software Engineering show its ability to discover latent topical structures in collections and also report collections that are similar based on their discovered topical structure.

General Terms

Text Mining, Information Extraction, Digital Library.

Keywords

Digital Library, Document Collection, Text mining, Topic Modeling, Topical Structure.

1. INTRODUCTION

With changing user expectations, many traditional libraries are moving toward digital content storage. Accessible from anywhere at any time, digital contents as stored in digital libraries provide users with efficient, on-demand information experiences. With this trend, the amount of digital contents especially digital text documents made available to users have tremendously increased over the years, being filled with hidden information in form of the varieties of topics of discourse inherent in them leading to information overload. Accordingly, users, mostly computational researchers are presented with challenges on the discovery and identification of the varieties of topical contents of the collections in the digital library. One cannot ordinarily and easily identify the various topics of discourse within the large collections of a digital library as simple search does not and cannot present us with the knowledge of the topics of interest that pervade these collections. One way for us to be able to have a grasp of the topical information that run through the documents collections of a digital library is through the approach of mining the text document collections of such digital library.

Text mining or textual data mining can be broadly defined as a knowledge-intensive process in which a user interacts with a document collection over time by using a suite of analysis tools. It refers to an interdisciplinary field that combines the various techniques from areas such as machine learning, natural language processing, computational linguistics, library and information sciences, and statistics for the purpose of retrieving and extracting information from digital text. Moreover, because of the centrality of natural language text to its mission, text mining draws on advances made in computer science disciplines concerned with the handling of natural language. Perhaps most notably, text mining exploits techniques and methodologies from the areas of information retrieval, information extraction, and corpus-based computational linguistics [1].

In a manner analogous to data mining, text mining seeks to discover and extract hidden but potentially useful information from textual data sources through the identification and exploration of interesting patterns [2]. These are kinds of discoveries that a researcher scouring thoroughly through textual documents one by one may never have noticed [3]. In contrast to data mining, the data sources of text mining are document collections, and interesting patterns are found not among structured and formalized database records but in the unstructured textual data in the documents of these collections. However, text mining derives much of its inspiration and direction from seminal research on data mining. [4] Therefore, it is not surprising to find that text mining and data mining systems clearly show many high-level architectural similarities. For instance, both text mining and data mining systems rely on preprocessing routines, pattern discovery algorithms, and presentation-layer components such as visualization tools to enhance the browsing of answer sets.

Topic modeling can be described as a form of text mining, a way of identifying topical patterns in a corpus. It is a method for finding and tracing clusters of words (called "topics" in shorthand) in large bodies of texts. A topic is essentially a recurring pattern of co-occurring words, i.e. cluster of words that frequently occur together in documents in statistically meaningful way. Formally, a topic refers to a probability distribution over words in a vocabulary. Topic models (e.g., [5, 6, 7,]) are based upon the idea that documents are mixtures of topics, where a topic is a probability distribution over words. A topic model being a generative model for documents specifies a simple probabilistic procedure by which documents can be generated. To make a new document, one chooses a distribution over topics. Then, for each word in that document, one chooses a topic at random according to this distribution, and draws a word from that topic. Standard statistical techniques can be used to invert this process, inferring the set of topics that were responsible for generating a collection of documents.

A topic model takes as input a collection of text documents, such as book pages. It outputs a user's preset number of "topics", which are probability distributions over the words in the collection. Topics are essentially determined by which words occur together across the collection. The most likely words for each topic can then be used to provide human-interpretable keywords for the topic [8]. The Topic model represents documents as bags of words without considering word order as being of any importance. The model has the ability to represent large document collections with lower dimensional topics, which represent clusters of similarly behaving (co-occurring) words. The Topic model reflects an intuition that documents contain multiple topics. This work comes up with a means of mining document collections of a digital library for topical structure discovery by developing a topic modelling-based application and via the use of loose-coupling integration method, the application is integrated with document collections built in the widely used digital library software system— Greenstone digital library system[9].

The rest of this paper is structured as follows. Section 2 provides a review of related work while Section 3 discusses the architectural framework, a detailed description of the proposed system's architecture. Section 4 describes the UML use case model of the system while experiments carried out with the aid of the proposed software application is discussed in Section 5 with Section 6 discussing the results obtained and Section 7 concludes the paper by summarizing its contributions and stating future expectations.

2. RELATED WORK

Rauber et al. [10] worked on mining digital library collections in the SOMLIb digital library system, built on neural networks to provide text mining capabilities. At its foundation they used the Self-Organizing Map to provide content-based clustering of documents. The Self-Organizing Map is a popular unsupervised neural network model, and a variation of this model, i.e. the Growing Hierarchical Self-Organizing Map (GHSOM), were used to topically structure a document collection similar to the organization of real-world libraries. By using this extended model, i.e. the GHSOM, they further detected subject hierarchies in a document collection, with the neural network adapting its size and structure automatically during its unsupervised learning process to reflect the topical hierarchy (structure). By mining the weight vector structure of the SOM, using LabelSOM their system was able to select keywords describing the various topical clusters. They demonstrated the capabilities of the SOMLib system using collections of articles from various newspapers and magazines in digital library. However there system does not consider any detection of topical similarity among collections of articles but only captures the topical patterns and the keywords describing them, among other issues. This work however proposes a system that achieves the discovery of topic-based similarity between collection pair through the use of an inverted Kullback-Leibler divergence measure.

3. ARCHITECTURAL FRAMEWORK

The architectural framework design in this work specifies a description of the proposed system in terms of a set of integrated components and concepts about how the components are connected. The proposed system's design is modeled after the concept of the text mining process model in Vidhya and Aghila [11] and the topic modelling-based framework proposed in Olowookere et al. [12]. The architectural design model of the proposed system is illustrated in figure 1. The proposed system consists of three main modules as described in Sub-sections that follows.

3.1 The Retrieval and Preprocessing Module

Upon the retrieval of collection of documents from the repository of a digital library, this module of the system performs preprocessing operations on the documents. The module takes the documents as input and is responsible for identifying and extracting from these raw unstructured text documents, representative features which form the basis upon which the topical structure discovery module acts. The preprocessing operations are carried out by three components; the tokenizer, stop-word removal component and the case folding component.

3.1.1 The Tokenizer

The tokenization component breaks up documents into meaningful constituents (i.e., tokens- words in this case), with space (space bar binary code) used to identify boundary between words (delimiter). The outputs of this component are token sequences. Tokenization actually, is the process of chopping up a given stream of text or character sequence into words, phrases, symbols, or other meaningful elements called tokens which are grouped together as a semantic unit and used as input for further processing such as parsing or text mining.

3.1.2 Stop-word Removal Component

The stop-word removal component when invoked, scans through a default stop-word file or user-defined stop-word file and removes the occurrences the words found in the stop-word file from tokens received from the tokenizer. The component then produces sequence of tokens that are void of those words in the stop-word file. A stop-word file is actually a text file that contains a list of very common English words (at least in the scope of this thesis) which do not convey any important meaning to the topical structure of the document collection— they are high frequent words that carry no information. A user of the proposed system can provide his/her own stop-word file, this is to give room for the inclusion of words that the user may consider to be too frequent or general across the whole collection and therefore may constitute nuisance to the output of the system. The default stop-word list adopted in this system is that which is made available in the Machine Learning for Language toolkit (MALLET) [13].

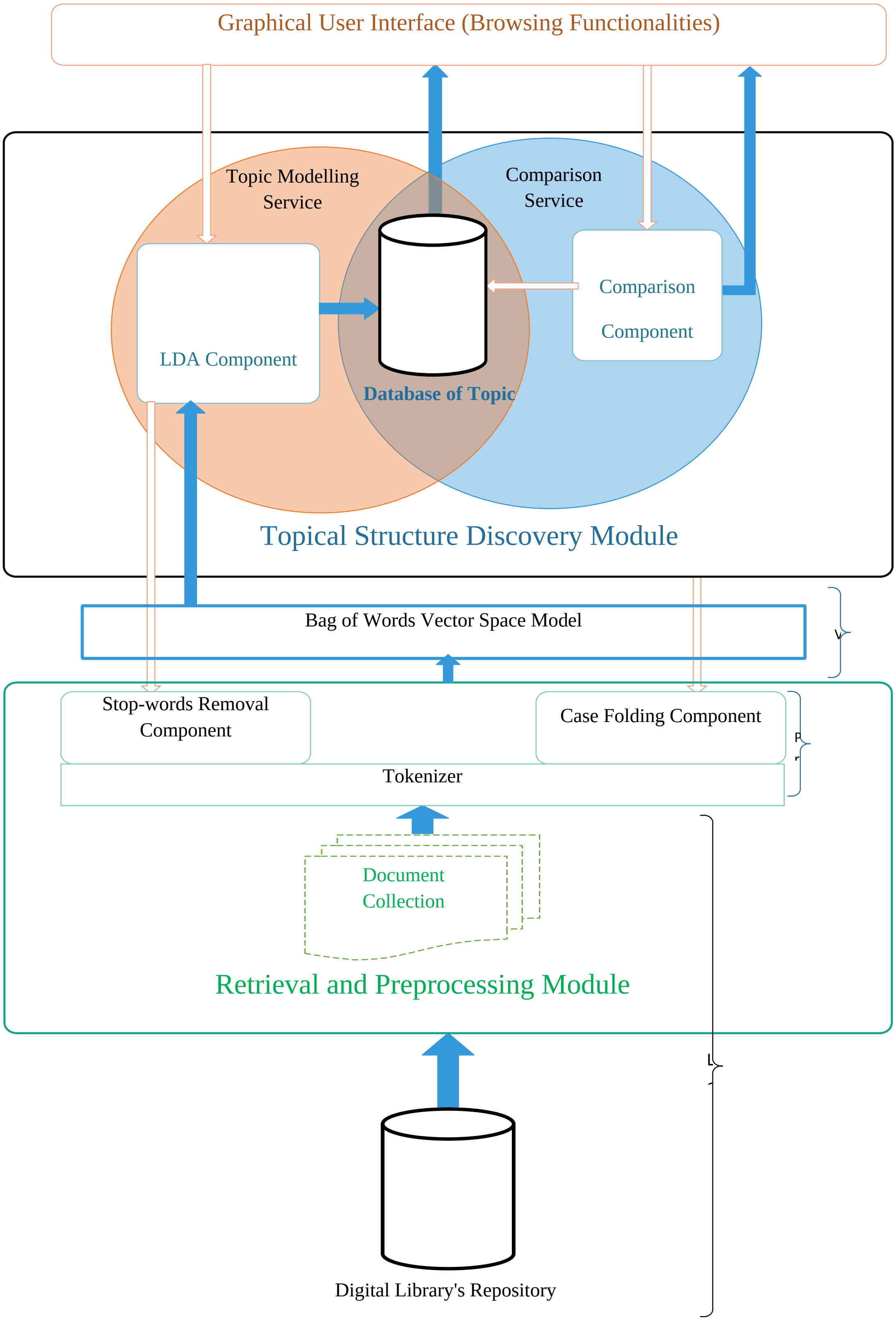

Graphical User Interface (Browsing Functionalities)
Topic Modelling Service
Comparison Service
LDA Component
Database of Topic
Comparison Component
Topical Structure Discovery Module
Bag of Words Vector Space Model
V
Stop-words Removal Component
Case Folding Component
P
r
Tokenizer
Document Collection
Retrieval and Preprocessing Module
L
Digital Library's Repository

Figure 1: Architectural Design Model of the Proposed System

3.1.3 Case Folding Component

This component simply either preserves the case of all the characters of the tokens (leaving them as they appear in the original text documents, i.e., case sensitive— lowercase and uppercase) or forces all characters to lowercase. The user of the proposed system can choose either way.

3.2 The Vectorization Module

To apply mathematical-based algorithms to natural language, there is need to convert the language into a mathematical format. The system uses a simple model known as the bag-of-words vector space model (BoW-VSM). VSM is an algebraic model representing textual information as a vector. VSM is a space where text is represented as a vector of numbers instead of its original string textual representation; the VSM represents the features extracted from the document. Modeling of documents into a vector space is to first create a dictionary of words present in the documents. To do that, all words are simply selected from the document and assigned with an index. Then using the indexes of the dictionary, each document is represented by a vector by counting the words in each document to turn each document into a word frequency vector and convert it to a dimension in the vector space. The vectorization module converts the sequence of token received from the retrieval and preprocessing module into the feature vectors which represents the documents, and the resulting vector representations are fed into the LDA component of the system.

3.3 The Topical Structure Discovery Module

The topical structure discovery module is the core of the proposed system, UPH Digital Library Miner. The module feeds on the feature vector representation of the documents and employs its components in the tasks of topical structure discovery. The components embedded in this module are the LDA component, the database for topic distribution and the comparison component.

3.3.1 The LDA Component

The LDA topic model is the core part of the topical structure discovery module. The LDA technique is used by the proposed system to analyze the words of the preprocessed text documents in order to discover the topics that run through them, and to discover from this analysis the proportion of the topics that are mentioned in each document. The LDA algorithm having been preset to model a fixed number of topics, draws a distribution over the words of the collection. Acting on the documents one after the other, the distribution over topics would place probability on each topic, and each word is drawn from one of those topics. A motivating intuition of this algorithm is that all the documents in the collection share the same set of topics, but each document exhibits those topics in different proportion. The algorithm of the LDA is described below;

The LDA Probabilistic Generative Process:

Step 1: For each topic number_1 to topic number_k, [$t_{1:K}$];

a. Draw a distribution over words

$p(w|t) \equiv t_k$ (i.e., Per-Topic word distribution).

Step 2: For each document *d* in the collection [1…*D*];

a. Randomly draw a distribution over topics

$p(t|d) \equiv \theta_d$ (i.e., Per-Document topic distribution)

b. For each word *w* in the document;

i. Randomly draw a topic from the distribution over topics in Step 2a (i.e., Per-document per-word topic assignment)

ii. Randomly draw a word from the corresponding distribution over the vocabulary (word).

The generative process of the LDA defines a joint probability distribution over both the hidden (latent) random variables and the observed variables. Clearly stating, the observed variables are the words of the documents while the hidden variables are the topical structure— per-topic word distribution (the topics), per-document topic distribution, and the per-document per-word topic assignment.

However, the goal of the LDA is not the generation of random documents through these distributions, but rather inferring the distributions from observed document. This inference process uses the observed words of the documents to infer the hidden topical structure, a process which can be seen as reversing the generative process— discovering the hidden structure that likely generated the observed document collection. Therefore, the goal of using topic modelling technique in the proposed system is to automatically discover the inherent topics

from the collection of documents. It is emphasize in this paper that the algorithm does not have any information about the themes of the documents and that the documents are not labeled with topics or keywords, hence its unsupervised nature.

The inference process uses the joint probability distribution from the generative process to compute the conditional probability distribution of the hidden (latent) variables given the observed variables. This conditional distribution is what is referred to as Posterior Distribution.

3.3.2 The Database of Topic Distribution

This paper refers to this database as database of topic distribution since the result of the LDA which is purely based on the topical distributions in the document collections are transmitted or stored in this database to provide access to the comparison component of the system. It forms a shared storage for the topic modeling service and the comparison service. XAMPP Server for Windows was used for the database design.

3.3.3 The Comparison Component

In this component, the inverted Kullback-Leibler divergence measure as earlier stated in this work is used to check for the semantic similarity between any two document collections. When two document collections are selected for comparison, the component assesses their topic distributions from the database of topic distribution to determine the percentage to which these two collections are similar.

The Kullback-Leibler (KL) divergence being a method employed in the measurement of distance between two probability distributions is inverted in the proposed system as a means to find the semantic similarity between two document collections di and dj based on their respective topic distributions. The Kullback-Leibler divergence is actually a distance function rather than similarity function, this is because it usually achieves it minimum when the two probability distributions being compared are maximally similar to each other. According to Nelken and Shieber [14], Kullback-Leibler divergence is an asymmetric dissimilarity measure between two distributions say x and y, it measures the added number of bits that are needed to encode events that are sampled from x using a code based on y.

4. USE CASE DIAGRAM

The UML (Unified Modelling Language) Use Case diagram is used to model and identify the static functional requirements of the software system. It is used to describe the interaction between the user and the system. It describes what the system does from the standpoints of an external observer. The use case diagram of the proposed system is shown in figure 2. The user selects the document collection he aims to mine from the repository, selects the output folder for results, and enters the number of topics to be learned. The user can choose to change the default settings in the advanced settings, and then click on "learn topics" to activate the system to discover the topics in the collection. The result can be explored via the "view distributions" tab. The user can then choose two already mined document collections for semantic similarity comparison

5. EXPERIMENTAL SETUP

In this section, the specific software packages and tools that were used in the development, implementation and testing of the application are briefly stated. Also, the empirical procedures used in the process of this implementation and testing of the application are discussed. These are the four (4) essential software packages and tools used for the experimental setup:

a) NetBeans Integrated Development Environment, version 8.0 (32 bits).

b) MALLET software package, version 2.0.7 [12].

c) XAMPP Server for Windows, version 3.2.1

d) Greenstone Digital Library Software, version 2.86 [9].

In this work, in order to achieve the topic modeling function of the system, some class libraries of the Gibbs sampling-based implementation of the Latent Dirichlet Allocation have been reused, bringing it to the specifications of this work. In essence the application is implemented partly with a few reusable classes from MALLET topic modeling tool [12, 15]. In the study, the Greenstone Digital Library software (GSDL) [9] is used to build the collections whose repositories is intended to be mined.

Based on the integration approach adopted in this work (the loose coupling integration), the application is designed and equipped to access the repository (storage) of the collections as provisioned by the GSDL, unlike attempting to design the application as an integral part of the GSDL. The documents in the collections are enriched with metadata such as year of publication, title and volume, in order to enhance easy browsing and arrangement in the digital library.

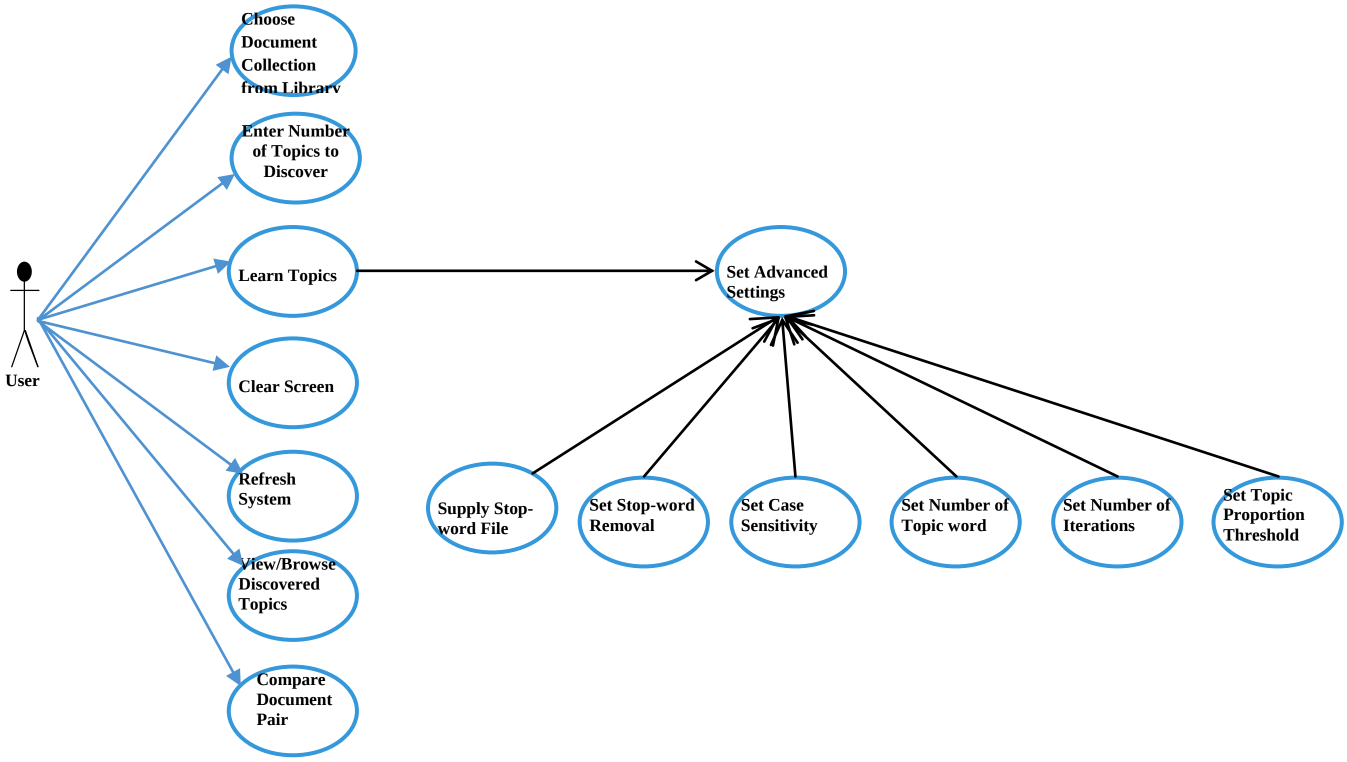
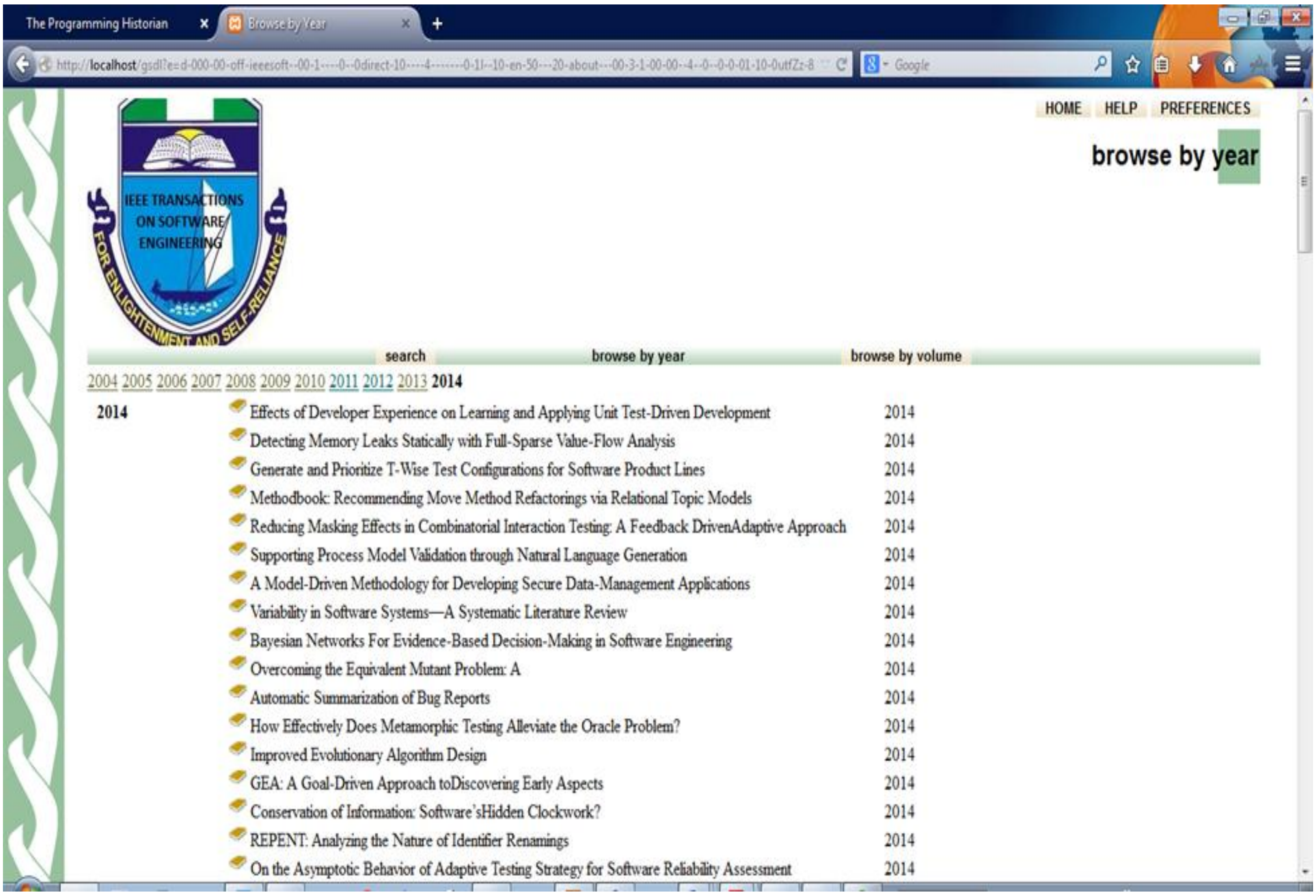


Figure 2: UML Use Case Diagram of the Proposed Systems

Figure 3 is a snapshot of the digital library collections built in Greenstone system for the purpose of this study. The Dublin Core metadata format (dc) was used in tagging each document with metadata. There are about 628 documents in the digital library collection— they are abstract of articles downloaded from the IEEE Xplore digital library, specifically, chosen from the IEEE transactions on software engineering (August 2004 – August 2014) [16]. However, the concern of the application, the UPH Digital Library Miner is to mine the raw documents (unprocessed) in the built digital library, without any recourse to the metadata of the documents. That is to say no information (subject or topical information) is needed by the application to discover the topical structure, and so text mining is done directly from the greenstone collections repository.


The Programming Historian
Browse by Year
http://localhost/gsdl?e=d-000-00-off-ieeesoft--00-1----0--0direct-10----4-------0-1l--10-en-50---20-about---00-3-1-00-00--4--0--0-0-01-10-0utfZz-8
Google
HOME
HELP
PREFERENCES
browse by year
IEEE TRANSACTIONS ON SOFTWARE ENGINEERING
FOR ENLIGHTENMENT AND SELF-RELIANCE
search
browse by year
browse by volume
2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014

2014	Effects of Developer Experience on Learning and Applying Unit Test-Driven Development	2014
	Detecting Memory Leaks Statically with Full-Sparse Value-Flow Analysis	2014
	Generate and Prioritize T-Wise Test Configurations for Software Product Lines	2014
	Methodbook: Recommending Move Method Refactorings via Relational Topic Models	2014
	Reducing Masking Effects in Combinatorial Interaction Testing: A Feedback DrivenAdaptive Approach	2014
	Supporting Process Model Validation through Natural Language Generation	2014
	A Model-Driven Methodology for Developing Secure Data-Management Applications	2014
	Variability in Software Systems—A Systematic Literature Review	2014
	Bayesian Networks For Evidence-Based Decision-Making in Software Engineering	2014
	Overcoming the Equivalent Mutant Problem: A	2014
	Automatic Summarization of Bug Reports	2014
	How Effectively Does Metamorphic Testing Alleviate the Oracle Problem?	2014
	Improved Evolutionary Algorithm Design	2014
	GEA: A Goal-Driven Approach toDiscovering Early Aspects	2014
	Conservation of Information: Software'sHidden Clockwork?	2014
	REPENT: Analyzing the Nature of Identifier Renamings	2014
	On the Asymptotic Behavior of Adaptive Testing Strategy for Software Reliability Assessment	2014

Figure 3: A screenshot of Digital Library Collections built in Greenstone.

5.1 The Experiment

Experiment was carried out on the collections in the digital library using the developed application to mine a selected collection (in this case, the volume of the IEEE transactions on Software Engineering selected for mining) from the Greenstone's repository. Choosing to remove stop-words and retaining characters' case, the parameters are set for the LDA-based learning algorithm as shown in table 1.

Table 1: Parameters Values for the experiment

Parameters	Value
Number of iterations	200
Number of topic words to print	4
Topic proportion threshold	0.0
Number of Topics	5

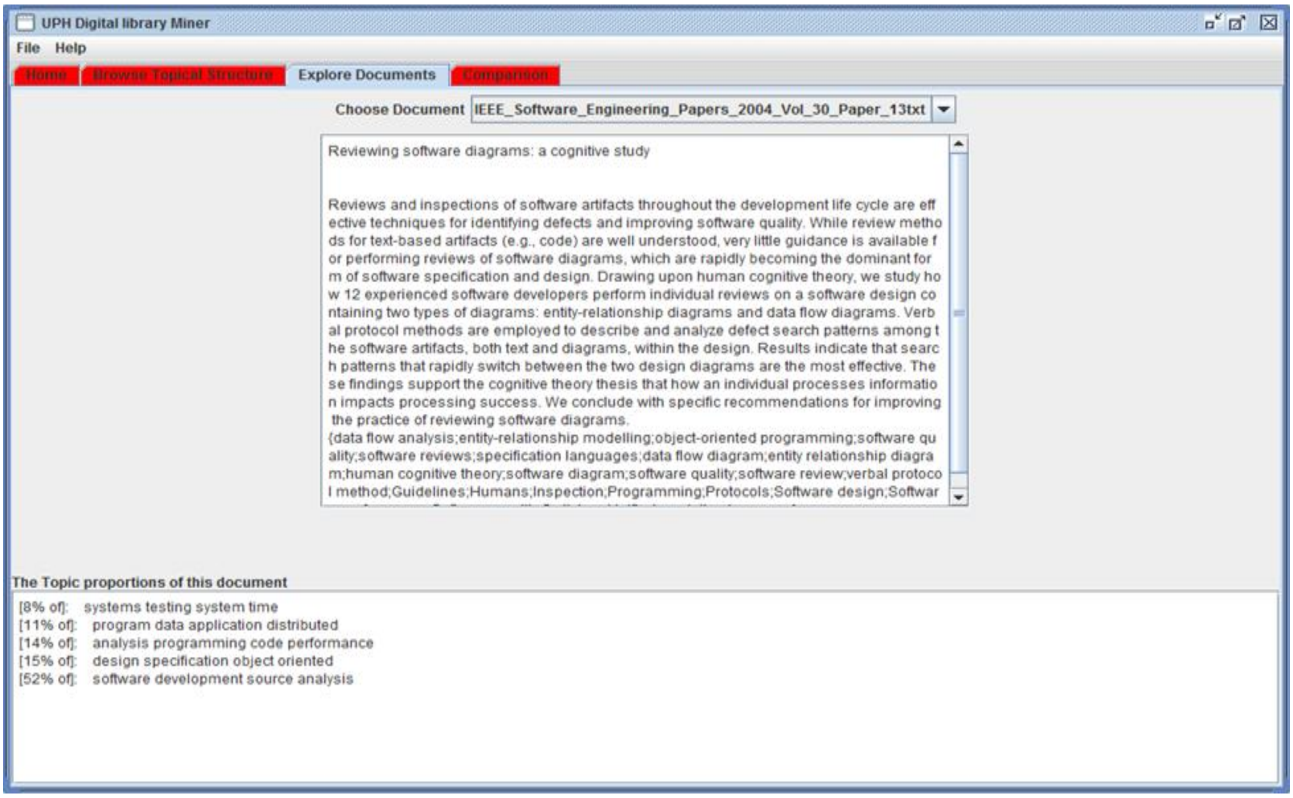
With all the parameters set and ensuring that the database is up and running, the application is then launched to discover the topics in the selected collection of the digital library. As soon as its discovery process completes, the documents and their discovered topical content proportions (in one-hundredth) are written to the database automatically. This process is repeated for all the collections available to be text-mined.

Figure 4 shows sample results obtained consisting the original contents of two sample documents (Paper 13 and Paper 5) in the IEEE collection that was mined and the proportions of the different topics discovered in them, which form the topical structure. The Comparison component of the application finds the similarity (based on the discovered topical structure) between two document collections from which topics have already been discovered. Figure 5 shows the result of finding similarities between the same collection, while figure 6 shows another result of finding similarities between different collections.

6. RESULT DISCUSSION

In the experiment which was carried, the developed application (UPH Digital Library Miner) has been used for the discovery of topical structures from document collections of the built digital library and also compared collections based on their discovered topical structures. Some of the outputs generated from this mining process by the system were shown. In figures 4a and 4b, taking a look at the cross-collection topics, based on the number of topics that have been chosen to be extracted from the collection (5 topics in this case), some of the topics can be interpreted literarily using the top words that make them up. Table 2 shows a probable interpretation of these topics in the software engineering domain.

In figures 4a and 4b, it is observed that the two documents considered touch the same multiple probable topics but in different proportions, with the most prominent topic assigned the highest percentage (proportion in one-hundredth). Examining paper 5 from figure 4b, it can be observed that the topic proportion is not based on that single document but



Choose Document IEEE_Software_Engineering_Papers_2004_Vol_30_Paper_13txt

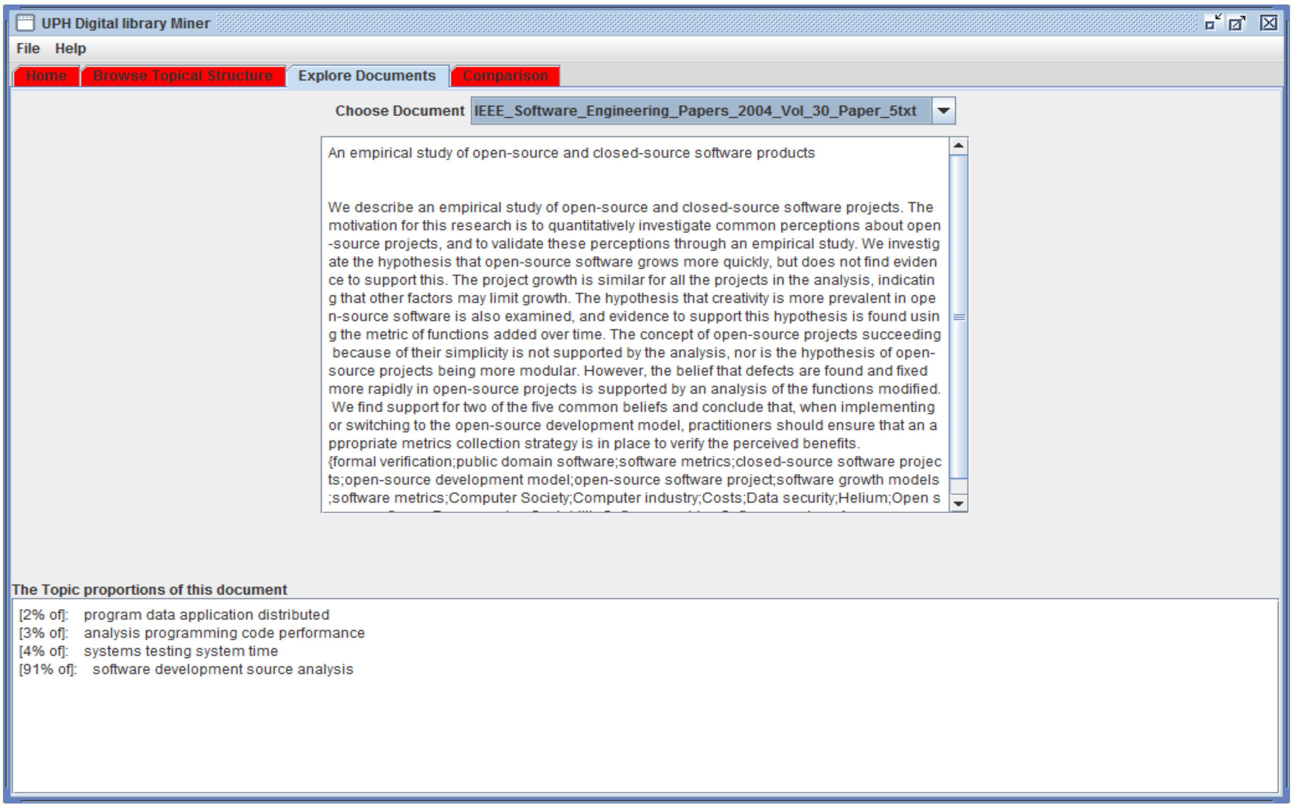
```
Reviewing software diagrams: a cognitive study


Reviews and inspections of software artifacts throughout the development life cycle are eff
ective techniques for identifying defects and improving software quality. While review metho
ds for text-based artifacts (e.g., code) are well understood, very little guidance is available f
or performing reviews of software diagrams, which are rapidly becoming the dominant for
m of software specification and design. Drawing upon human cognitive theory, we study ho
w 12 experienced software developers perform individual reviews on a software design co
ntaining two types of diagrams: entity-relationship diagrams and data flow diagrams. Verb
al protocol methods are employed to describe and analyze defect search patterns among t
he software artifacts, both text and diagrams, within the design. Results indicate that searc
h patterns that rapidly switch between the two design diagrams are the most effective. The
se findings support the cognitive theory thesis that how an individual processes informatio
n impacts processing success. We conclude with specific recommendations for improving
 the practice of reviewing software diagrams.
{data flow analysis;entity-relationship modelling;object-oriented programming;software qu
ality;software reviews;specification languages;data flow diagram;entity relationship diagra
m;human cognitive theory;software diagram;software quality;software review;verbal protoco
l method;Guidelines;Humans;Inspection;Programming;Protocols;Software design;Softwar
```

The Topic proportions of this document

```
[8% of]:   systems testing system time
[11% of]:   program data application distributed
[14% of]:   analysis programming code performance
[15% of]:   design specification object oriented
[52% of]:   software development source analysis
```

Figure 4a: A Sample of Topics discovered in the 2004 collection (Examining Papers 13)



Choose Document IEEE_Software_Engineering_Papers_2004_Vol_30_Paper_5txt

```
An empirical study of open-source and closed-source software products


We describe an empirical study of open-source and closed-source software projects. The
motivation for this research is to quantitatively investigate common perceptions about open
-source projects, and to validate these perceptions through an empirical study. We investig
ate the hypothesis that open-source software grows more quickly, but does not find eviden
ce to support this. The project growth is similar for all the projects in the analysis, indicatin
g that other factors may limit growth. The hypothesis that creativity is more prevalent in ope
n-source software is also examined, and evidence to support this hypothesis is found usin
g the metric of functions added over time. The concept of open-source projects succeeding
 because of their simplicity is not supported by the analysis, nor is the hypothesis of open-
source projects being more modular. However, the belief that defects are found and fixed
more rapidly in open-source projects is supported by an analysis of the functions modified.
 We find support for two of the five common beliefs and conclude that, when implementing
or switching to the open-source development model, practitioners should ensure that an a
ppropriate metrics collection strategy is in place to verify the perceived benefits.
{formal verification;public domain software;software metrics;closed-source software projec
ts;open-source development model;open-source software project;software growth models
;software metrics;Computer Society;Computer industry;Costs;Data security;Helium;Open s
```

The Topic proportions of this document

```
[2% of]:   program data application distributed
[3% of]:   analysis programming code performance
[4% of]:   systems testing system time
[91% of]:   software development source analysis
```

Figure 4b: A Sample of Topics discovered in the 2004 collection (Examining Papers 5)

discovered based on co-occurrence of words from the whole collection. Noticeably, the system reports that the document touched four (4) probable topics even though the system has been set to discover five (5) topics from the whole collection. This shows that a topic is not forced on any given document in the collection, in as much as it does not probably touch such topic.

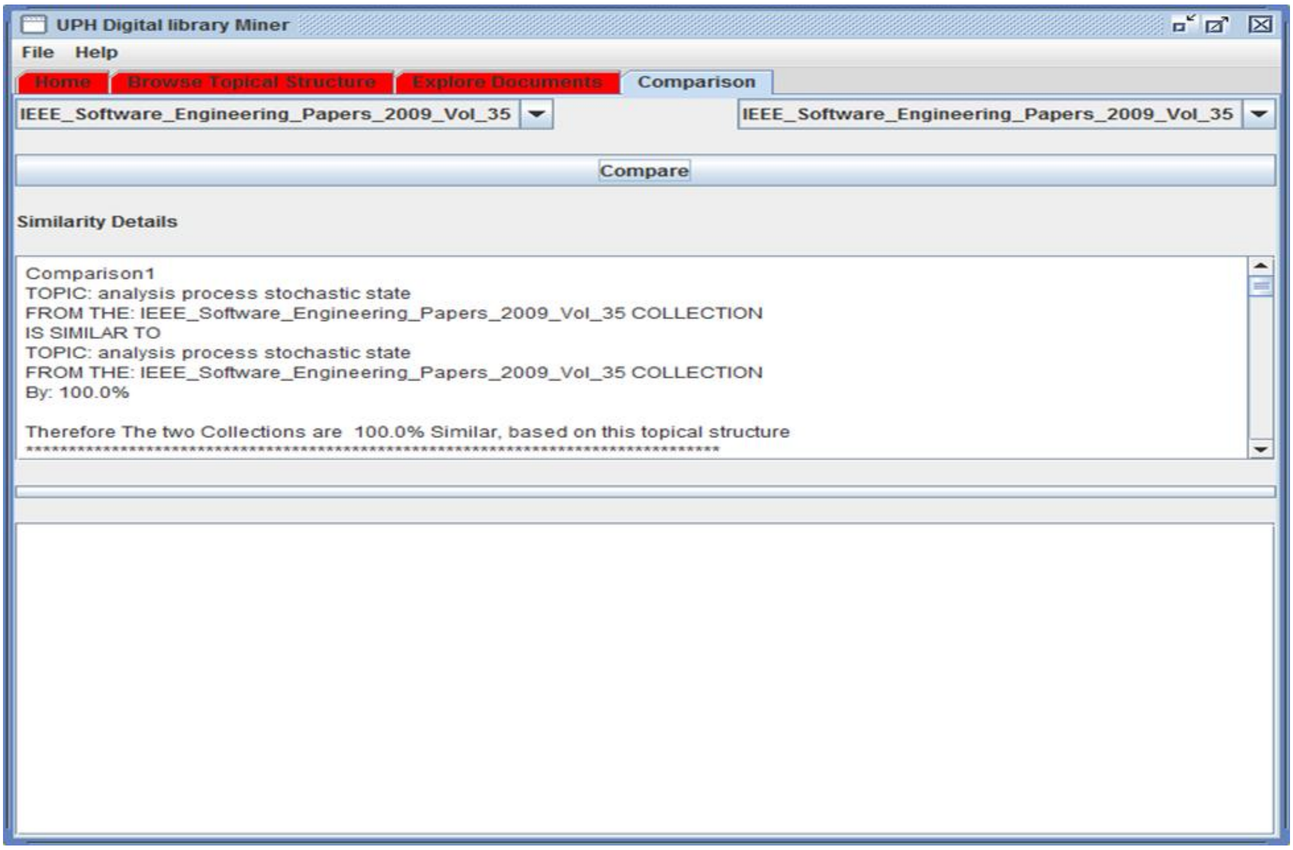
Taking a look at the comparison in figure 5, the same documents collection (2009 collection) is being compared for topic-based similarity check. It is obvious as expected that the selected collections are completely similar (100%) based on the topical structure discovered in them. This is naturally expected to be so since they contain exactly the same content.

In figure 6 however, the collection by the left (volume 35, 2009) is comparatively similar to the collection at the left (volume 37, 2011) by 25%, based on topical structure that was discovered in them.

From the results of mining through the collections of the IEEE software engineering from 2004 to 2014, it is observed that the topic that relates to "Software/Program testing" is prominent (pervades) throughout the collections over the years. The consistent discussions on this topic points to its importance and the attention that the software engineering community gives to this topic.

Table 2: An interpretation of topics from the IEEE Collection

	Topic 1 **Distributed Programming**	Topic 2 **Performance Analysis**	Topic 3 **System Testing**	Topic 4 **Program Specification**	Topic 5 **Software Development**
The top Topic Words	*Program* *Data* *Application* *Distributed*	*Analysis* *Programming* *Code* *Performance*	*Systems* *Testing* *System* *Time*	*Design* *Specification* *Object* *Oriented*	*Software* *Development* *Source* *Analysis*


UPH Digital library Miner
File
Help
Home
Browse Topical Structure
Explore Documents
Comparison
IEEE_Software_Engineering_Papers_2009_Vol_35
IEEE_Software_Engineering_Papers_2009_Vol_35
Compare
Similarity Details

```
Comparison1
TOPIC: analysis process stochastic state
FROM THE: IEEE_Software_Engineering_Papers_2009_Vol_35 COLLECTION
IS SIMILAR TO
TOPIC: analysis process stochastic state
FROM THE: IEEE_Software_Engineering_Papers_2009_Vol_35 COLLECTION
By: 100.0%

Therefore The two Collections are  100.0% Similar, based on this topical structure
************************************************************************************
```

Figure 5: Comparing the same collection.

7. CONCLUSION

This paper has presented a topic modelling-based software application for discovering the topical structure of document collections of a digital library. The application also has the capability of finding topic-based similarities between document collection pairs. Adopting a loose-coupling technique, the application was integrated with Greenstone digital library system and then deployed to mine collections in the digital library's repository, distinctly showing the topical structure of collections and similarities between collections. The approach proposed in this work will enhance text mining capabilities in digital library systems, including similarity search for the purpose of easier classification of documents. The outcome of this study will be of benefit to information users and researchers alike as it aids them to analyze digital library collections based on their topical contents. Future work is intended to improve on the integration approach of the system considering a tight-coupling integration with a digital library environment and also investigate an automatic topic-words interpretations of discovered topics in the system.

8. REFERENCES

[1] Hearst, M. 1999. Untangling Text Mining. In Proceedings of the 37th Annual Meeting of the Association of Computational Linguistics. College Park MD, Association of Computational Linguistics, Morristown, NJ. pp.3-10.

[2] Feldman, R. 1998. Practical Text Mining. In Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery. London: 478.

[3] Lammey, R. 2014. CrossRef's Text and Data Mining Services. Learned Publishing, Vol. 27, No. 4, pp. 245-250.

[4] Feldman, R and Sanger, J. 2006. The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data, New York: Cambridge University Press.

[5] Hofmann, T. 1999. Probabilistic Latent Semantic Indexing. In In Proceedings of the 22nd International Conference on Research and Development in Information Retrieval, pp. 50-57.

[6] Steyvers, M. and Griffiths, T. 2005. Probabilistic Topic Models. In Landauer, T., McNamara, D., Dennis, S. and Kintsch, W. (ed), Latent Semantic Analysis: A Road to Meaning, Laurence Erlbaum.

[7] Blei, D., Ng, A. and Jordan, M. 2003. Latent Dirichlet allocation. Journal of Machine Learning Research, vol. 3, pp. 993–1022.

[8] Mimno D. and McCallum, A. 2007. Mining a Digital Library for Influential Authors. In JCDL'07 ACM, Vancouver, British Columbia, Canada, June 18–23.

[9] Rajasekharan, K. and Nafala, K. M. 2007. Building up a Digital Library with Greenstone, A Self-Instructional Guide for Beginners. Thrissur, India.

[10] Rauber, A. and Merkl, D. 2003. Text Mining in the SOMLib Digital Library System: The Representation of Topics and Genres. Applied Intelligence, vol. 18, 271–293.

[11] Vidhya, K. A. and Aghila, G. 2010. Text Mining Process, Techniques and Tools: an Overview. International Journal of Information Technology and Knowledge Management, vol. 2, no. 2, pp. 613-622.

[12] Olowookere, T. A., Eke B. O. and Oghenekaro, L. U. 2015. A Topic Modelling-Based Framework for Mining Digital Library's Text Documents. IEEE African Journal of Computing and ICTs, Nigeria, vol. 8, no 4.

[13] McCallum, A. K. 2002. MALLET: A Machine Learning for Language Toolkit. http://mallet.cs.umass.edu.

[14] Nelken, R. and Shieber, S. M. 2006. Computing The Kullback-Leibler Divergence Between Probabilistic Automata Using Rational Kernels. Harvard University, Division of Engineering and Applied Sciences, Cambridge.

[15] Ramage, D. and Rosen, E. 2009. Topic Modeling Toolbox Stanford NLP group, http://nlp.stanford.edu/software/tmt/tmt-0.4.ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=28304&punumber=32.