Intelligent Information Management, 2012, 4, 364-382
http://dx.doi.org/10.4236/iim.2012.46041 Published Online November 2012 (http://www.SciRP.org/journal/iim)

Low-Power Themes Classifier (LPTC): A Human-Expert-Based Approach for Classification of Scientific Papers/Theses with Low-Power Theme

Mohsen Abasi[1], Mohammad Bagher Ghaznavi-Ghoushchi[2]
[1]Computer Engineering Department, Ferdowsi University of Mashhad, Mashhad, Iran
[2]School of Engineering, Shahed University, Tehran, Iran
Email: abasi.mohsen@stu-mail.um.ac.ir, ghaznavi@shahed.ac.ir

Received September 1, 2012; revised October 26, 2012; accepted October 31, 2012

ABSTRACT

Document classification is widely applied in many scientific areas and academic environments, using NLP techniques and term extraction algorithms like CValue, TfIdf, TermEx, GlossEx, Weirdness and the others like. Nevertheless, they mainly have weaknesses in extracting most important terms when input text has not been rectified grammatically, or even has non-alphabetic methodical and math or chemical notations, and cross-domain inference of terms and phrases. In this paper, we propose a novel Text-Categorization and Term-Extraction method based on human-expert choice of classified categories. Papers are the training phase substances of the proposed algorithm. They have been already labeled with some scientific pre-defined field specific categories, by a human expert, especially one with high experiences and researches and surveys in the field. Our approach thereafter extracts (concept) terms of the labeled papers of each category and assigns all to the category. Categorization of test papers is then applied based on their extracted terms and further comparing with each category's terms. Besides, our approach will produce semantic enabled outputs that are useful for many goals such as knowledge bases and data sets complement of the Linked Data cloud and for semantic querying of them by some languages such as SparQL. Besides, further finding classified papers' gained topic or class will be easy by using URIs contained in the ontological outputs. The experimental results, comparing LPTC with five well-known term extraction algorithms by measuring precision and recall, show that categorization effectiveness can be achieved using our approach. In other words, the method LPTC is significantly superior to CValue, TfIdf, TermEx, GlossEx and Weirdness in the target study. As well, we conclude that higher number of papers for training, even higher precision we have.

Keywords: Natural Language Processing (NLP); Semantic Web; Term Extraction; Text Categorization; Resource Description Framework (RDF); Low-Power Theme

1. Introduction

There may be several survey papers in a research field. Latest and somehow old papers' highlights, in addition to a clear description of the field, are included in the survey papers. They summarize and organize recent research results and experiences in a novel way that integrates and adds understanding to tryings in the field. They use highlights of many papers in a research field to understand what sub-field each paper is in. Imagine an author want to write a survey paper or book in a research field. He has to read many papers have been accepted and indexed by many different conferences and journals. He must read them in details to understand accurately what research sub-filed each is in. Due to the large amount of papers, and also in many subfields, time and energy consuming is a typical process repeated each day until completion of the survey paper. So, automatic tools which can classify papers and help our imagined author in preparation of a good survey paper, are very useful and have an important value.

As another reason of using an automatic paper classifier, categorization of new papers submitted to a conference into the conference's research areas may be supposed. A conference chairman has to select a correct reviewer for each paper submitted to the conference in a little time. Since there are many large amounts of papers submitted to the conference, and they are in a lot of research fields covered in the conference, the process of selecting a correct reviewer has a lot of difficulties. In addition, selecting a correct reviewer has a vital importance in having a conference with a high impact factor; because a good paper may be rejected as a result of selecting an incorrect reviewer hasn't had many experiences in the paper's research area. Even so, as the time pass, the number of papers in previous conference issues is increasing dramatically, so attempting to find a specific category that a paper is related to it becomes harder and harder. In this situation finding an automatic way of learning important terms in each category by last conference years' labeled papers in it and then using these obtained categories' terms in selecting the category of each new submitted paper is important.

Most famous approach used to automatically classify a bunch of papers is by extracting their terms. Machines can do it with NLP techniques. There are some NLP terms extraction algorithms such as CValue, TfIdf, TermEx, Simple Frequency, GlossEx and weirdness that are being used by many classification tools. But they are mostly weak in extracting most important terms when input text has not been structured grammatically correct. Besides, these current algorithms have no using of advantages of semantic techniques in extracting meaningful and concept terms. Because of these and another disadvantages in those, we propose a novel Text-Categorization and Term-Extraction method called Low-Power Themes Classifier (LPTC). The proposed approach gets some scientific pre-defined field specific categories prepared by a human expert, especially one with high experiences and researches and surveys in the field, as its training phase substances and then categorize test papers based on terms extracting from them and comparing with each category's terms. By considering large number of usages of humans for categorization tasks such as ontology generation and many else, in spite of existed high performance and technological machines, we all may believe that humans, especially an expert in a field, have still most accurate skills in classifying texts comparing to all tools running on the machines. Therefore we have used an expert's prepared categories and their containing papers to collect our training terms for declaring correct category of a new unlabeled text paper based on its extracted terms. The proposed approach is mainly based on concept terms in abstract of a research paper and will generate semantic enabled outputs that are useful for complementing knowledge bases and datasets of Linked Data cloud and also for semantic quering of them by some languages such as SparQL.

Our novel tool categorizes input test papers into categories are predefined by a known expert in the related research field. Especially we have chosen Mr. Chandrakasan's famous book [1]. A brief bibliography of him is collected in **Table 1**.

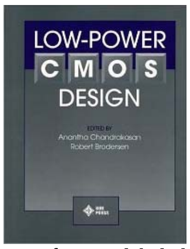
Also a summary of the used books as our scientific research fields resource, is presented in **Table 2**.

Disadvantages of using a reference survey hand book

Table 1. Chandrakasan's brief profile.

Expert	Chandrakasan's Profile	
	Feature	Value
Hardware & Architecture, Electrical & Electronic Engineering, Networks & Communications (Major in digital circuit and systems, low-power)	Publications	433
	Citations	21,986
	G-Index	142
	H-Index	64
	Amazon Books (Author and Co-Author)	14
	MIT Dspace/Theses (Supervisor/Advisor)	282/215
	IEEE Xplore Citations (via Google)	18,900

Table 2. Target survey book as our reference of categories.

Target Resource	Basic Sections in Brief	
	Section	Items
 The set of papers labeled, by our human expert (introduced in **Table 1**), with predefined subcategories on digital circuit and systems, low-power	Computer Aided Design Tools	26
	Circuit and Logic Styles	28
	Driving Interconnect	11
	Efficient DC-DC Conversion and Adaptive Power Supply System	24
	Low Voltage Technologies and Circuits	37
	Memory Circuits	24
	Portable Terminal Electronics	26

as resource of categories is mainly caused by time passing factor. We use categories created by an expert in some years ago and since that time, we have had many changes in research fields/subfields of a domain (e.g. Low Power) including: New fields/subfields addition, dividing existed fields into multiple new fields and etc. These changes can be result in incorrect categorization when we want to classify papers written in recent years. One way to overcome it is using papers submitted in recent years as our learning items. These papers can be found in related annual conferences such as ISSCC[1].

So, using an automatic paper classifier, which can help the reviewer selector in classifying papers into predefined categories (the research areas of the conference) and then sending the paper to the guessed category's reviewer(s), has many benefits.

[1]Conference site: http://www.Isscc.org

So, the above sample reasons encouraged us to develop a tool which classifies unlabeled papers into predefined categories. If the studying of accepted papers in the conference of previous years done accurately, the knowledge of each category's keywords may increase and consequently choosing of the paper reviewer can be done correctly.

One of the methods for automatic document discrimination based on their subjects is Text Categorization (TC). Since a decade, text categorization has become an active field of research in the machine learning community. In the TC problem, we are trying to assign sample documents to the set of predefined classes automatically. TC has different names in the world published papers including "text classification" or "topic spotting", and is useful in Information retrieval (IR) and machine learning (ML). Any task which is involved in segmentation of a document (such as determining the subject of a document, spam filtering, categorizing emails based of their subjects, indexing books in the library and) falls into scope of TC. Most of the approaches are based on the term occurrence frequency. The performance of such surface-based methods can decrease when the texts are too complex, *i.e.*, ambiguous. One alternative is to use the semantic-based approaches to process textual documents according to their meaning [2]. In general, all Automations are appealing because it eliminates the needs of humans to do the task so it reduces time and costs effectively. In addition, this rule is valid in TC. Using text categorization, conferences can categorize their new submitted papers more quickly and efficiently.

After an introduction to the proposed tool and its importance in many situations, we review related works and then the background knowledge we should have to understand the proposed tool's processing steps. Details of the proposed approach and used algorithms are presented in its next section; And in the end, we have results and experiences.

2. Related Work

In recent years, extensive research has been done in the scope of TC including multivariate regression analysis[3, 4], Bayesian probability [5,6], classification based on neural networks [7], inductive learning methods[8,9], learning based on decision trees [5], learning from symbolic rules [8,10,11], nearest neighbor classifier [12,13].

Besides, Text Document Categorization is being done by support vector machine (SVM) theory mostly. SVM has been reported as one of the best performing machine learning approaches for classification. However, many of researches have focused on the critical problem of the SVM in determining the optimal combination of kernel function and parameters, in order to guarantee high efficiency and effectiveness of the classification tasks. By using the computationally intensive grid search algorithms, we can combine the SVM kernel and parameters. [14,15] have some working on this. This method of combination varies different types of kernel functions and parameters through a wide range of values using geometric steps. The set of kernel and parameters combination with the best cross-validation accuracy is selected. Also [16] has presented a new text document classification framework called Euclidean-SVM. It uses SVM approach in the training phase and the Euclidean distance function in the classification phase. By generating a decision surface, namely the optimal separating hyperplane, the SVM constructs a classifier to partition different categories of data points in the vector space.

Like many studying areas, Fuzzy theory has been used in SVM. Some papers illustrate this. Using one-against-one fuzzy support vector machine, [17] is doing multi-class text categorization. Its dataset is Reuter's news. It uses OAO-FSVM to implement a multi-class text classification problem. OAO stands for one-against-one.

Besides these, many attempts have been done measurement criteria used in semantic Text Categorization (TC). Some of them have been presented a different term weighting scheme from usual ones for example TF-IDF. As a sample research, [18] proposes a novel term weighting scheme in comparison to TF-IDF. This novel schema works by exploiting the semantics of categories and indexing terms.

Also there are many researches in semantic TC, such as [19] that presents a concise semantic analysis (CSA) technique for text categorization tasks. CSA extracts a few concepts from category labels and then implements concise interpretation on words and documents; [20] that proposes MBPNN (modified back-propagation neural network). We can solve wide kinds of problems by BPNN. It consists of at least three layers (one input, one output, and at least one hidden layer) and uses back-propagation as learning mechanism. [2] In which the semantic meanings of words are extracted using the unified medical language system (UMLS) framework. In TC, each document is represented by a vector of non-negative numbers. Each number shows the occurrence of a specific word in that document. Combining all documents result in a matrix with a dimension of N by d called bag-of-words. N represents the number of documents and each of d columns represents the frequency of the specific words in documents. TC algorithms often analyze bag-of-word to learn from sample documents and predict the category of future documents. Because of high dimensional learn samples, learning will take a lot of time and is not efficient. In most cases, we try to find features, which have most influences in class separation and eliminate other features called feature selection. In this paper, we use a semantic method for feature selection and compare it to one simple non-semantic method. Feature selection helps on reducing the high dimensionality of samples into a much lower dimensionality but tries to keep the distance between classes in the bag-of-word as much as possible. Reducing the dimension helps on lowering the computational cost of categorization, but it is likely to make classes more mixed. Every feature in classes has its own effect in class definition thus removing it has individual influence in class separation, so selecting which feature is better to be removed based on semantic methods from samples is our focus in this paper. For this purpose, results of two feature selection methods were proposed and evaluated, including term strength (TS) [21] method and a semantic method.

Anyone can recognized our proposed approach in categorizing papers as relating to measuring papers' relatedness. Some approaches have been tried to measure similarity of two papers but by their non-textual features [22] in comparing to our proposed approach does its categorization using textual vector of each paper. However for relatedness of scientific papers, the approach proposed in [22] model a digital library on a directed graph in which every node represents one entity: a paper, an author or a venue. On another side, each edge shows relationships between these entities. Therefore there would be six different types of relations between any two papers.

In this paper a novel approach, with a high precision in comparing to similar ones, is proposed for categorizing of unlabeled input papers and then identifying their subject or topic. This novel approach uses NLP techniques for getting a paper's concept terms to be able to measure similarity of abstracts in two different papers.

3. Background

We have used many technologies in our work; mostly semantic web related technologies. Some of them are: RDF (Resource Description Framework), Ontology and famous Ontologies such as SKOS and SIOC, NLP based tools, and etc. This section contains their definition.

One of the important parts of our proposed approach is the process of extracting terms and keywords of a paper's abstract. This process has been called Named Entity Recognition (NER).

3.1. Named Entity Recognition

NER is the process of extracting keywords from a text. It can be done using rule-based or statistical approaches or a combination. NER is as same as Information Extraction task but including two parts: 1) Finding the entity boundaries; 2) Founded text strings categorization into types. Some good NER systems have been developed, such as "Nymble" explained in [23] that use machine learning: Hidden Markov Models and MENE (Maximum Entropy Named Entity) is proposed in [24] that use Maximum Entropy model. NER can be done also with the helping of Semantic and Linked Data technologies.

3.2. Semantic Web and Linked Data

The semantic web is web of data. It is not a different web than our current normal World Wide Web (WWW) in case of containing data, but its data are structured and linked. The main goal of using semantic web is putting data on the Web in a form that both of machines and humans can understand it. For achieving this purpose, Linked Data and its standards help us. So, one important thing anyone should care about in semantic web activeties is Linked Data. Linked Data is a method of broadcasting, sharing, and connecting data on the Web via resolvable URIs that can be followed automatically by a machine. For show all Linked Data around the world, Richard Cyganiak and Anja Jentzsch created the Linked Data Diagram of the Cloud. Each distinct data set published as Linked Data has been shown as a node in this cloud diagram. Also RDF links between items in two data sets are represented as arcs. The Linked Data cloud consists of over 7.4 billion RDF triples, interlinked by 142+ million RDF links (Last updated by 2011-Sep-19). **Figure 1** shows it in a small view.

3.3. Resource Description Framework (RDF)

RDF provides a model for describing resources that may have some properties. In RDF, a "resource" is any object that can be uniquely identified by a Uniform Resource Identifier (URI). RDF, as a product of the Worldwide Web Consortium (W3C), is an application of XML [25]. The proposed tool generates FOAF documents as one of its output types.

3.4. FOAF (Friend of a Friend)

FOAF, the famous semantic web project in our world, help us to achieve one important goal in semantic web: Storing data in distributed locations and using ontologies and reasoning to aggregate it. Users in any social network, can produce their semantic profiles by using this vocabulary [26]. A FOAF document may contains coauthors of an academic user and his research fields of interest. Also it can give us contact information, bibliography of the user.

3.5. Ontology

There are many definitions for word Ontology. However, the following contains a good one: "Ontology is a formal specification of a shared conceptualization" (Tom Gruber,

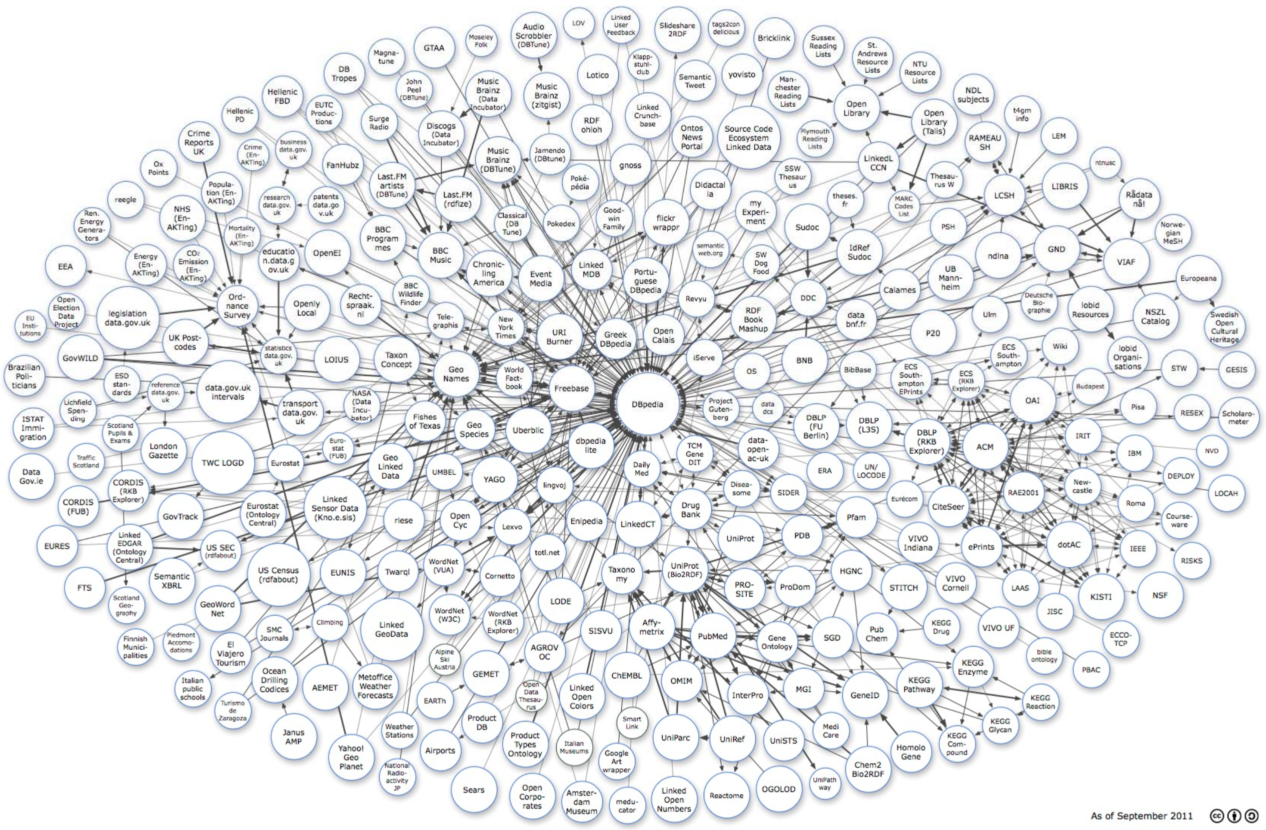


Figure 1. Linked Data cloud (September 2011) [http://richard.cyganiak.de/2007/10/lod/imagemap.html].

see [27]). A large amount of ontologies is currently defined, such as: FOAF, SIOC, and SKOS.

There are some ontology libraries in Web. Ontology libraries are the systems that collect ontologies from different sources and facilitate the tasks of finding, exploring, and using these ontologies [28]. However, we found two sufficient ontologies we have used in generating our outputs: SKOS and SIOC.

Simple Knowledge Organization System (*SKOS*) is used for representation of thesauri and taxonomies concept terms or any other type of structured controlled vocabulary. It is built upon RDF and RDFS.

The *SIOC* (*Semantically-Interlinked Online Communities*) Core Ontology is required to describe information from online communities (e.g., message boards, wikis, we blogs, etc.) on the Semantic Web. SIOC uses RDF too for defining of its data. Exporting information about the content and structure of online community websites in a machine-readable form is a design purpose of SIOC. Thus, various tools, exporters and services have been created to expose SIOC data from existing online communities.

3.6. Vector Space Model

We can represent each text document d_j as a vector in the following manner:

$$d_j = \left(t_{1j}, t_{2j}, \cdots, t_{nj}\right) \quad (1)$$

where t_{mj} is *m*’th term of document d_j. So after this representation, each paper abstract will be a vector. Therefore, vector operations can be used to compare new abstracts with each existing category represented as a vector of its concept terms. We have used these criteria in our test phase, when the database is filled of concept terms and their relevance values in each category.

3.7. Coseine Similarity of an Abstract to a Category

By comparing the deviation of angles between the new abstract’s vector and each category’s vector where the category is represented as the same kind of vector as the abstract, Relevance rankings of abstracts can be calculated, using the assumptions of document similarity’s theory.

In practice, it is easier to calculate the cosine of the angle between the vectors, instead of the angle itself. Formula of the cosine of two angles is as:

$$\cos\Theta = \frac{d_j \cdot q}{\left\|d_j\right\| \times \left\|q\right\|} \quad (2)$$

where d_j is the new abstract's vector and q is the category's vector is tested for relevancy.

Using the cosine the similarity between abstract d_j and category q can be calculated as:

$$\mathrm{sim}\left(d_j,q\right)=\frac{d_j\cdot q}{\left\|d_j\right\|\left\|q\right\|}=\frac{\sum_{i=1}^{N}w_{i,j}w_{i,q}}{\sqrt{\sum_{i=1}^{N}w_{i,j}^2}\sqrt{\sum_{i=1}^{N}w_{i,q}^2}} \quad (3)$$

This measure has the following advantages:

- Allows ranking documents according to their possible relevance;
- Simple model based on linear algebra;
- Allows partial matching.

4. Proposed Approach's Details

To do NER, we had decided to use MIT Text2Rdf java package, at the first. Text2Rdf is a text mining application which converts documents into the unambiguous and machine understandable format RDF. We wanted to use it since it returns important terms of a text in separated RDF nodes like:

<kmm:term>bidirectional power flow</kmm:term>

That we can parse the output RDF result of the package and extract named entities in these nodes as concept terms of an abstract. This way is a good one that has no needs of internet request/response processes and has a high speed and performance. However, we couldn't use it because of some limitations described below:

If one paragraph has many words and complicated sentences, this package can't parse it. For example, the phrase: "The feature of the bidirectional power flow allows the converter to provide energy as a power source or to regenerate power back to the AC power grid with high overall energy efficiency." But with a simpler one it has no problem: "The feature of the bidirectional power flow allows the converter to provide energy as a power source or to regenerate power back to the AC." Also if in one abstract just one *dot* has been omitted accidently, the tool will output no RDF.

Because of above and some else limitations, we had to consider a different NER tool. After searching the web in some days, we found a powerful one named: "Al-chemy-API". As its developers said us, it uses well known Natural Language Processing (NLP) algorithms and machine learning. The following contains the features of its web service:

- It supports many languages. With this functionality of our tool, any conference can review a paper that is in a non-English language.
- It links an extracted entity to a variety of other sources: The website for the entity, Freebase, MusicBrainz, and others. By help of this useful advantage we are able to do many cross language works, such as categorizing a paper has been written in a different language from training papers;
- It allows a large number of web service calls per day, so for a high-impact factor conference with a large number of papers to guess their categories in a few days, it is a good one;
- AlchemyAPI provides comprehensive support for RDF and Linked Data, enabling any content to be brought into the "Semantic Web" world with relative ease.

Most of files in our presented tool, both input and generated output ones, are represented semantically. In other words, they obey semantic web standards given by W3C consortium[2].

AlchemyAPI leverages Linked Data to provide additional information describing named entities detected within a piece of content.

So, AlchemyAPI is a good choice for us to be able to reach our goal in linking our generated results to Linked Data.

To use AlchemyAPI, you must have a valid **API access key** via its free-and-easy registration. AlchemyAPI has many tools: Named Entity Extraction, Concept tagging, Keyword/Term Extraction, Sentiment analysis, Relation Extraction, Topic categorization.

If we choose RDF, **Figure 2** can help us in extraction processing.

Whenever an entity is successfully disambiguated, Linked Data is included in API responses. This includes "*sameAs*" RDF links to related objects in the Linked Data cloud's datasets.

Figure 3 is an example RDF-formatted response with links to Linked Data (for the state of "Massachusetts").

Linked Data is supported within all AlchemyAPI response formats, including XML, JSON, and RDF.

4.1. Criteria of Measures

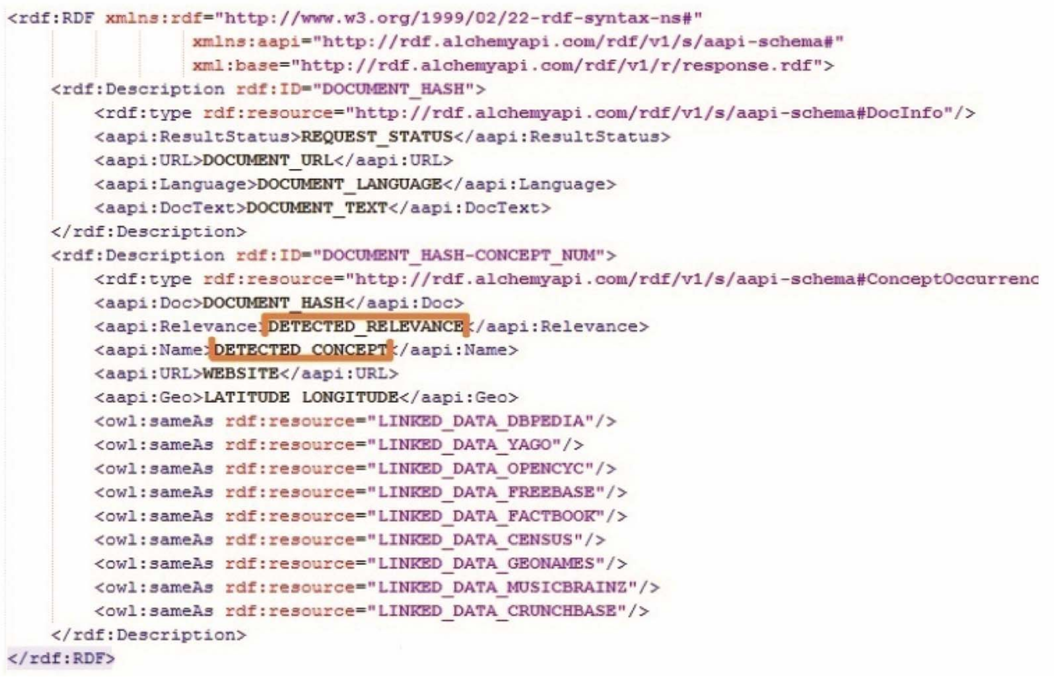
In this section, we introduce our measures used throughout of this paper and discuss how they are calculated and their pros and cons.

4.1.1. Occurrence of Terms in Each Documents

Both evaluated methods in this research are trying to find document categories by analyzing the number of occurrence of concept terms in each document (TF). Also they use another metric representing number of documents has the concept term (DF). These methods use DF to determine which word has low category based information, and so it has a least role in distinguishing of documents.

For calculating DF, number of occurrence of words in each document is counted. Moreover, for each DF a

[2]http://www.w3.org/2001/sw/wiki/Main_Page, http://semanticweb.or g/wiki/Semantic_Web_standards



```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:aapi="http://rdf.alchemyapi.com/rdf/v1/s/aapi-schema#"
         xml:base="http://rdf.alchemyapi.com/rdf/v1/r/response.rdf">
    <rdf:Description rdf:ID="DOCUMENT_HASH">
        <rdf:type rdf:resource="http://rdf.alchemyapi.com/rdf/v1/s/aapi-schema#DocInfo"/>
        <aapi:ResultStatus>REQUEST_STATUS</aapi:ResultStatus>
        <aapi:URL>DOCUMENT_URL</aapi:URL>
        <aapi:Language>DOCUMENT_LANGUAGE</aapi:Language>
        <aapi:DocText>DOCUMENT_TEXT</aapi:DocText>
    </rdf:Description>
    <rdf:Description rdf:ID="DOCUMENT_HASH-CONCEPT_NUM">
        <rdf:type rdf:resource="http://rdf.alchemyapi.com/rdf/v1/s/aapi-schema#ConceptOccurrenc
        <aapi:Doc>DOCUMENT_HASH</aapi:Doc>
        <aapi:Relevance>DETECTED_RELEVANCE</aapi:Relevance>
        <aapi:Name>DETECTED_CONCEPT</aapi:Name>
        <aapi:URL>WEBSITE</aapi:URL>
        <aapi:Geo>LATITUDE LONGITUDE</aapi:Geo>
        <owl:sameAs rdf:resource="LINKED_DATA_DBPEDIA"/>
        <owl:sameAs rdf:resource="LINKED_DATA_YAGO"/>
        <owl:sameAs rdf:resource="LINKED_DATA_OPENCYC"/>
        <owl:sameAs rdf:resource="LINKED_DATA_FREEBASE"/>
        <owl:sameAs rdf:resource="LINKED_DATA_FACTBOOK"/>
        <owl:sameAs rdf:resource="LINKED_DATA_CENSUS"/>
        <owl:sameAs rdf:resource="LINKED_DATA_GEONAMES"/>
        <owl:sameAs rdf:resource="LINKED_DATA_MUSICBRAINZ"/>
        <owl:sameAs rdf:resource="LINKED_DATA_CRUNCHBASE"/>
    </rdf:Description>
</rdf:RDF>
```

Figure 2. RDF response of AlchemyAPI web service. Highlighted positions are locator of a sample concept tarm and its relevance that are extracted and calculated by AlchemyAPI.

```
<owl:sameAs
rdf:resource="http://www.dbpedia.org/resour
ce/Massachusetts"/>
<owl:sameAs
rdf:resource="http://rdf.freebase.com/ns/guid.
9202a 8c04000641f8000000000589f0a"/>
<owl:sameAs
rdf:resource="http://umbel.org/umbel/ne/wiki
pedia/Massachusetts" />
<owl:sameAs
rdf:resource="http://sw.opencyc.org/concept/
Mx4rvVi4Qpw pEbGdrcN5Y29ycA"/>
<owl:sameAs
rdf:resource="http://mpii.de/yago/resource/M
assachusetts"/>
```

Figure 3. Linking to the linked data cloud with RDF sameAs node.

lower boundary is considered. Words with lower DF than this boundary will be removed from document DF vector because these words are assumed to have no meaningful category based information or in the other words, these terms cannot define their category boundary. We can say these words are somehow noise in their documents and removing them is recommended. Threshold the Frequency is a very simple and intuitive way to reduce high dimensional documents to a lower dimension and is very low cost on huge document collections. It should be noted that a term with low frequency is not always a noise so removing it may cause categories to be mixed even more.

4.1.2. Concept Term's Relevance Score

Relevance Score is a self-criteria calculated by the NLP tool: AlchemyAPI we used it. This measure identifies conceptual relevancy of a term to a text, such as an article's abstract. In the other words, testing an abstract of a paper on Philosophy, especially on Metaphysics, term "existence" has a high relevance score. AlchemyAPI web service returns these criteria for each (concept) term in different types of response, such as XML, RDF, and JSON.

Note that this relevance score's possible values are in range of: [0.0 - 1.0] (1.0 = most relevant).

4.1.3. Semantic Text Categorization

After evaluating above approach, we tried a simple intuitive methodology to measure the similarity of each new paper's abstract to each category. The category with highest similarity value is the paper's conceptually closest category. For example, for a sample paper, we obtained the values of **Figure 4**.

So the guessed category of this paper based on our approach is "Access Control" with about 0.563 relevancy.

4.2. Proposed Text Documents Categorization Steps

After making ready the input papers and categorizing them to their research filed based on, for example, a survey, we extract concept terms of their abstracts. We have done this by requesting AlchemyAPI web service with our API key that has permission of 30,000 requests per day. This permission has been gained via mailing to AlchemyAPI's supporting man. The default allowed number of requests is less than this, but after defining yourself as an approved academic user to AlchemyAPI, 30,000 would be the maximum number. More daily API calls (up to 200,000,000 daily) are available through an AlchemyAPI Subscription Plan. The AlchemyAPI response type is selectable, that we selected RDF one for extracting terms and their relevance. **Figure 5** is a sample RDF response. In this RDF response, ontologies such as: AAPI, OWL, GEO (representing the geographic coordinates associated with the concept tag) have been used.

Also there exists a *sameAs* link to *DBpedia* for the concept term. This link is provided only for entities that exist in this linked data-set. Besides *DBpedia*, there are many other linked databases: *Yago*, *Opencyc*, *Freebase*, *Geonames*, *MusicBrainz*, etc.

The running phases are described in detail below:

4.3. Running Phases

In our proposed tool's processing, there are two phases **Training phase** and **Test phase**, as shown in **Figure 6**.

4.3.1. Training (Learning) Phase

In this phase, we teach the machine which words/phrases are repeated more than others in each category. Then we fill the database with these kinds of data. We have several folders each has one available category's documents. In each document, we have either the abstract text of an article or PDF version of the whole paper which must be processed for converting to clear text and then abstract extraction. First of all, we extract the terms of each document with the AlchemyAPI web service and store them with their repletion frequency in the database. After that for each category, we make the union of the terms of its documents' abstracts and store them in the database with the sum of each term's frequency in all the documents of that category. Now we have the categories list and their terms with repletion of each term in that category. For easier comparing of together we need to have a database with the same terms. So at the last step of learning phase, we make union of the terms of all groups together and put zero for terms that are not appeared in a category. Then we have a table in the database that has the terms of all categories and the frequency of each term in each category. Now we can use these data for comparing new documents and finding the nearest group that the new article belongs to. You can see this phase in flowchart of **Figure 7**.

4.3.2. Test Phase

At this step, as flowchart in **Figure 8** shows, the input is a document that we want to know its category. First of all, we extract the abstract of that document and then the frequency of each term of that abstract. For being able to compare the results of this step to the results of training phase (the data in database), we should change the format of storing the new data by changing the term position to database term's position, omitting extra terms and put zero for non-repeated terms. Now we can compare the repeat frequency of terms of the input document with all the other categories' terms and find the nearest category to this document. For understanding how close is the article to each category, we use $\cos\theta$ of two term-frequency vectors. The bigger $\cos\Theta$ of the angle between two vectors, the closer the vectors, the more similar input document to the category.

In this phase, we also use some sub functions as follows:

4.3.2.1. Calculating Most Relevant Category to a Test Paper

Which category will is the real test paper's category? The answer is the goal of Test phase. As above described, in this phase we calculate how much relevant the test paper is to each category. In reality, Relevancies values are cosine of the angle between two vectors: **document's vector** and **category's vector.** These values are represented as floating point numbers. After calculation of these cosine values, we see which category has the bigger value to the document and pick it up as the most relevant category to the test paper. **Figure 9** shows the steps of similarity calculation algorithm (for a category) in detail.

4.3.2.2. Calculating Similarity of the Test Paper to a Category

Based on cosine approach and by using database filled in training phase, we calculate the similarity of the test paper to a both represented as the vector of their concept tags and keywords. The whole process has been shown in **Figure 9**.

4.4. The Outputs of the Proposed Tool

The produced outputs of our tool are very useful and can be used in many situations.

In this section we describe use cases and formats of the different important tool outputs.

```
Computer Aided Design Tools#0.015482821057133608
Cricuit and Logic Styles#0.01697272632720655
Driving Interconnect#0.0026487816687796695
Efficient DC-DC Conver…#0.03763287708305729
Low Voltage Technologies…#0.02668243571956768
Memory Circuits#0.1699517413227048
Portable Terminal El…#0.0034081276892642147
```

Figure 4. Pre defined categories and their similarities (relevancies) to a sample test paper.

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:aapi="http://rdf.alchemyapi.com/rdf/v1/s/aapi-
schema#" xmlns:geo="http://www.w3.org/2003/01/geo/wgs84_pos#" xmlns:owl=http://www.w3.org/2002/07/owl# …>
  <rdf:Description  rdf:ID="dafb6c3104655b2a84d4808c9e77cd6d8e4f090ac-gc_0">
    <rdf:type rdf:resource="http://rdf.alchemyapi.com/rdf/v1/s/aapi-schema#ConceptOccurrence"/>
    <aapi:Doc>dafb6c3104655b2a84d4808c9e77cd6d8e4f090ac</aapi:Doc>
    <aapi:Relevance>0.929312</aapi:Relevance>
    <aapi:Name>Design</aapi:Name>
      <owl:sameAs rdf:resource="http://dbpedia.org/resource/Design"/>
      <owl:sameAs rdf:resource="http://rdf.freebase.com/ns/guid.9202a8c04000641f8000000000012f73"/>
      <owl:sameAs rdf:resource="http://sw.opencyc.org/concept/Mx4rvVivMpwpEbGdrcN5Y29ycA"/>
  </rdf:Description>
```

Figure 5. Sample RDF response of AlchemyAPI for a paper with a sample named entity and its relevance to the paper.

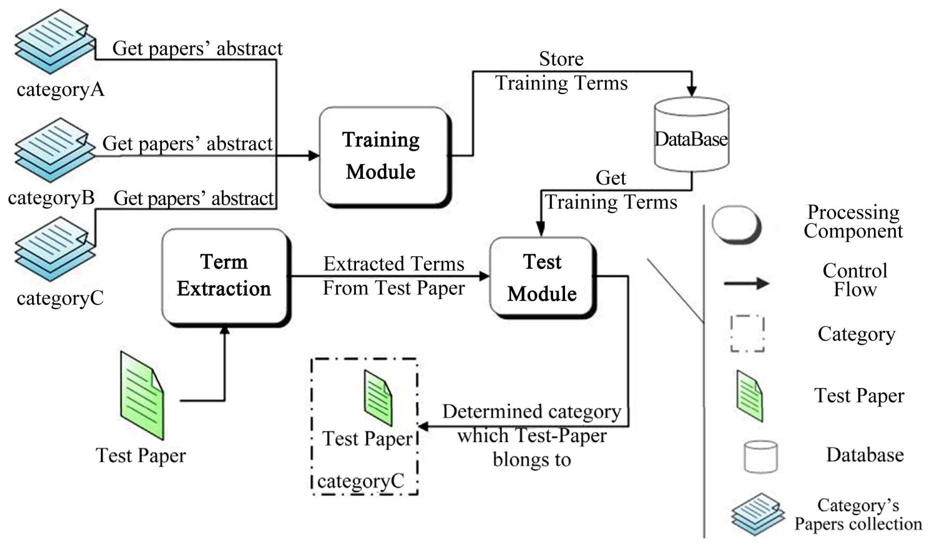


Figure 6. Total system in a view. From start to end: (a) Training phase—Categorized (Labeled) papers as inputs and a filled database of terms & Relevancies of them as output; (b) Test phase—Comparing a test paper to each category and guessing of the category paper belongs to.

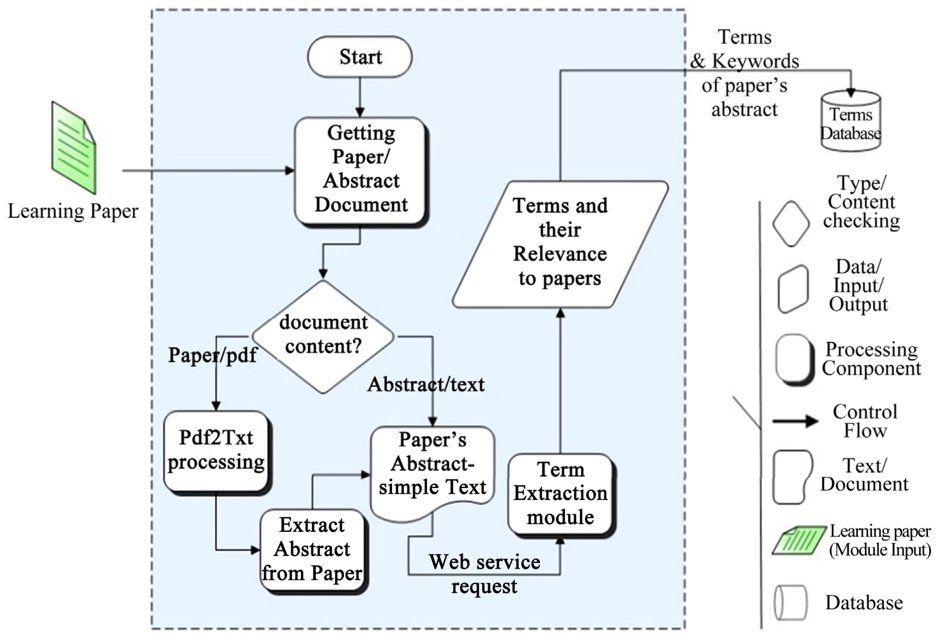


Figure 7. Training phase: Filling database with each category's concept terms and keywords extracted from the category's papers abstracts.

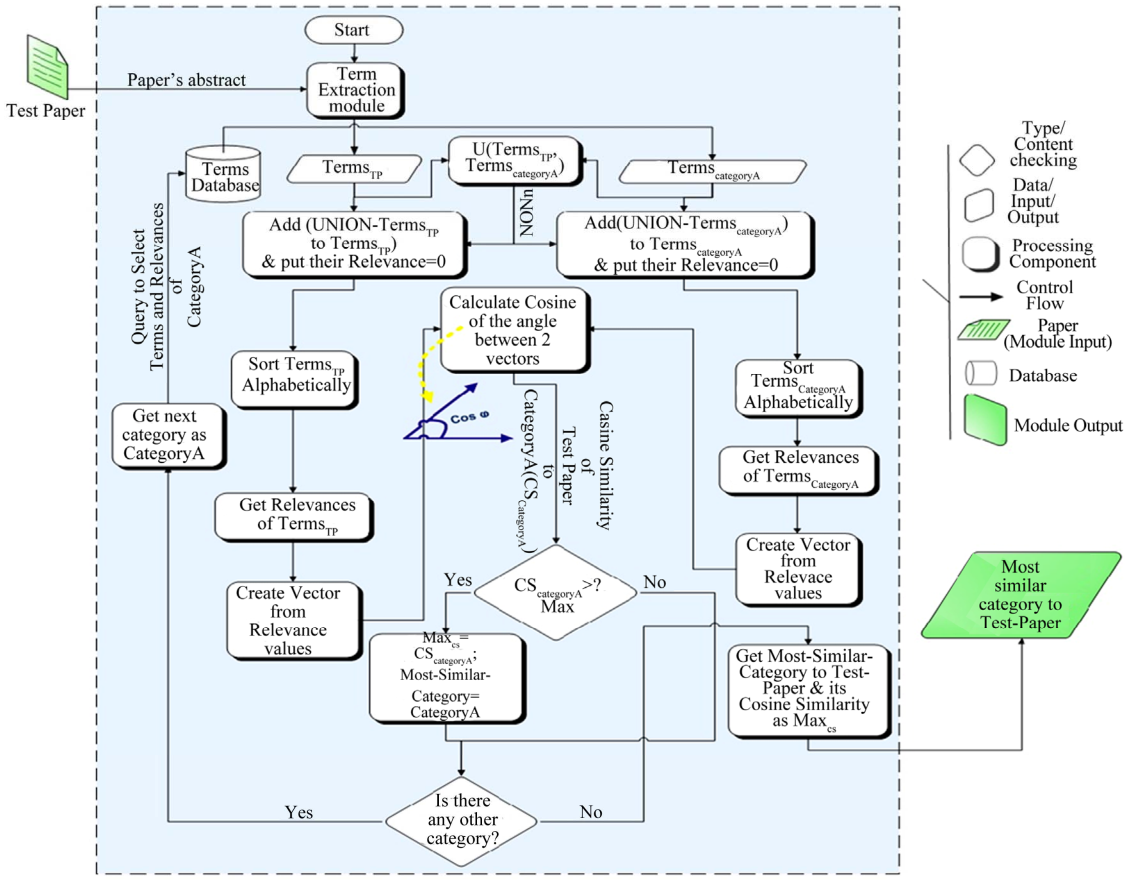

Start
Test Paper
Paper's abstract
Term Extraction module
Terms Database
Terms$_{TP}$
U(Terms$_{TP}$, Terms$_{categoryA}$)
Terms$_{categoryA}$
UNION
Add (UNION-Terms$_{TP}$ to Terms$_{TP}$) & put their Relevance=0
Add(UNION-Terms$_{categoryA}$) to Terms$_{categoryA}$ & put their Relevance=0
Query to Select Terms and Relevances of CategoryA
Calculate Cosine of the angle between 2 vectors
Cos φ
Sort Terms$_{TP}$ Alphabetically
Sort Terms$_{CategoryA}$ Alphabetically
Cosine Similarity of Test Paper to CategoryA(CS$_{CategoryA}$)
Get next category as CategoryA
Get Relevances of Terms$_{CategoryA}$
Get Relevances of Terms$_{TP}$
Create Vector from Relevace values
Create Vector from Relevance values
Yes
CS$_{categoryA}$>? Max
No
Max$_{cs}$ = CS$_{categoryA}$; Most-Similar-Category= CategoryA
Get Most-Similar-Category to Test-Paper & its Cosine Similarity as Max$_{cs}$
Most similar category to Test-Paper
Yes
Is there any other category?
No
Type/ Content checking
Data/ Input/ Output
Processing Component
Control Flow
Paper (Module Input)
Database
Module Output

Figure 8. Test phase: Process of guessing test papers (papers we want to guess their categories).

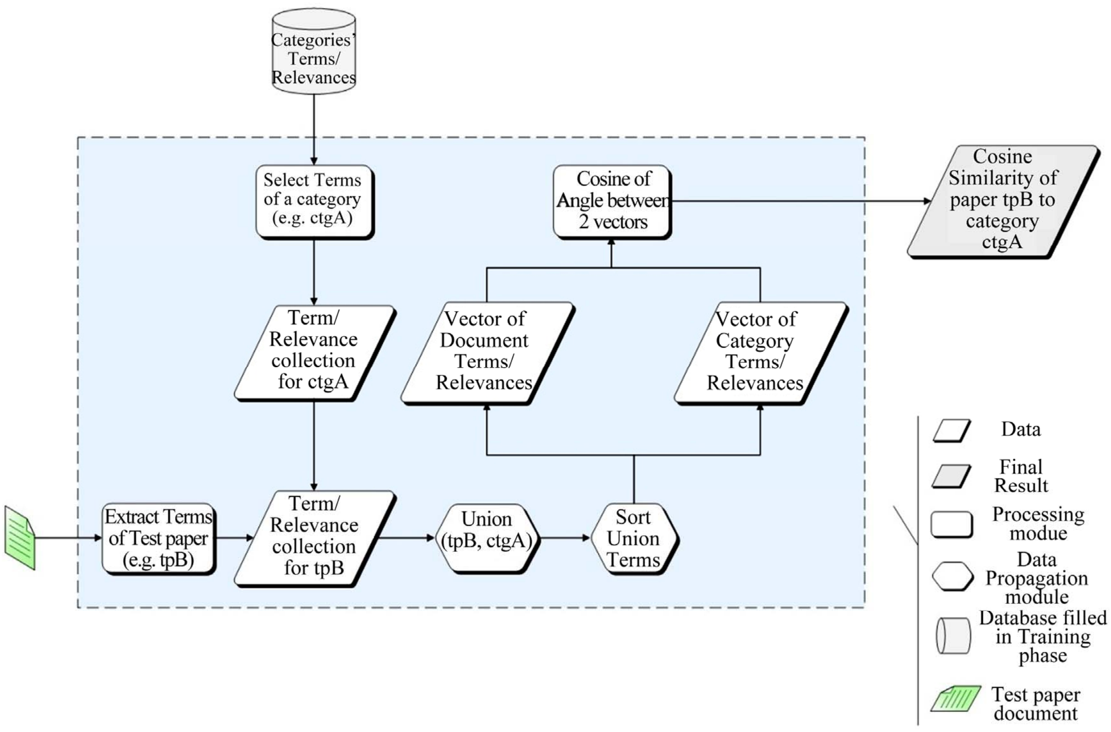

Categories' Terms/ Relevances
Select Terms of a category (e.g. ctgA)
Cosine of Angle between 2 vectors
Cosine Similarity of paper tpB to category ctgA
Term/ Relevance collection for ctgA
Vector of Document Terms/ Relevances
Vector of Category Terms/ Relevances
Extract Terms of Test paper (e.g. tpB)
Term/ Relevance collection for tpB
Union (tpB, ctgA)
Sort Union Terms
Data
Final Result
Processing modue
Data Propagation module
Database filled in Training phase
Test paper document

Figure 9. Algorithm of calculating *cosine similarity* of a test paper to a category by using Database filled in the training phase.

FOAF of the Authors

In the learning phase, besides extracting terms of the input papers from their abstracts, we save name of the papers'. We store these names in RDF files using FOAF vocabulary described above. For a sample FOAF output see **Figure 10**.

Generated FOAF documents can be viewed using some visualization tools such as: GraphViz and Prefuse. For a Prefuse based sample visualization of a social network file indicating friends of persons in **Figure 10**, see **Figure 11**.

From this figure, we can see that this FOAF is about "**Cabal-Yepez**" and also we can infer that "**Granados-Lieberman**" is one of the persons have submitted a paper with the help of "**Cabal-Yepez**". In the simplest words, "**Granados-Lieberman**" is a co-author of "**Cabal-Yepez**".

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-
rdf-syntax-ns#"
xmlns:admin="http://webns.net/mvcb/"
xmlns:foaf="http://xmlns.com/foaf/0.1/"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
<foaf:Person>
<foaf:name> Cabal-Yepez, E.</foaf:name>
<foaf:knows>
 <foaf:Person>
  <foaf:name>Granados-Lieberman, D.</foaf:name>
 </foaf:Person>
 <foaf:Person>
  <foaf:name> Romero-Troncoso, R.J.</foaf:name>
 </foaf:Person>
 <foaf:Person>
  <foaf:name> Osornio-Rios, R.A.</foaf:name>
 </foaf:Person>
 <foaf:Person>
  <foaf:name> Garcia-Perez, A.</foaf:name>
 </foaf:Person>
</foaf:knows>
</foaf:Person>
</rdf:RDF>
```

Figure 10. Sample of FOAF output document.

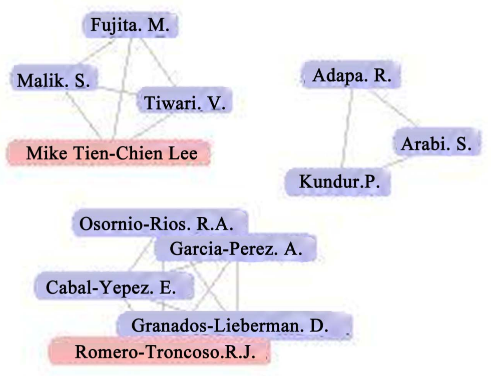


Figure 11. Sample visualization of a FOAF document with Prefuse toolkit [http://prefuse.org/gallery/graphview].

With these FOAF files in hand, we can understand what an academic author has research on and who his co-authors in writing papers are. This can be done by using SPARQL, a rich query language used for Semantic Web purposes such as consulting databases of FOAF data.

With the help of these FOAF documents as our novel proposed tool, we can make Linked Data much larger and more powerful. For instance, we can make links between two familiar datasets, DBLP (a server providing bibliographic information on major computer science journals and proceedings [29]) and FOAF profiles, in Linked Data Cloud depicted in **Figure 1**.

4.5. Our Standard (Semantic) Output Formats

Since our output's formats are useful in many further research attempts, we consider them here in this paper:

4.5.1. Output Format One

This output format is a simple text file that we make it for each of our documents and says the relevance of the document to each of the categories. Relevance is a number between 0 and 1 that shows how close the document is to the category. The style of storing these Txt files depends on all of categories and their documents. Each Txt file is stored in a folder positioned like the initial folders' hierarchy that was the input to the test phase. In each Txt file number of lines is equal to number of training phase's categories, and at the first of each line name of each category is written and in front of that, there is the relevance number of the document to that category separated with a # character and the name of the Txt file is the same name of the input tested document. **Figure 12** is an example of this output type, assuming that there are only three different categories in our field: Access Control, Secure Semantic Web Services and Trust Management.

4.5.2. Output Format Two

This output file is a unique Txt file that stores the final result of the process of categorizing the documents in test phase. Number of lines in this file is equal to number of the documents and each line starts with the name of document and its path on the hard disk and front of it there is the name of guessed category for that document separating with ---- sign. After that there is a # sign and then the relevance value of the document to the guessed category. **Figure 13** shows you an example of this output with three lines:

4.5.3. Output Format Three

The purpose of this type of output is the same as the second type. Only the format is different and more machine understandable. The format of the third output type is XML and you can see a short example of it in **Figure 14** below. This sample output includes only one category with two documents assigned to it.

4.5.4. Output Format Four

This type is the most useful type for further semantic purposes and researches. The structure of this output is also the same as the second and third outputs and just the format is RDF in this step. The process of making this RDF file is done by two ontologies *SKOS*, *SIOC* that we introduced them above.

The *post* RDF schema element of *SIOC* is used to represent each document, and *topic* element is used to identify guessed category of that document. Within that *SKOS* ontology's classes are used since this ontology has definitions for the categories as they are a research filed each. One sample of this useful and good output is **Figure 15**.

5. Experiments and Results

5.1. Evaluation of the Proposed Algorithm Itself

Based on the survey book on "Low power digital CMOS design" [1], we generated a dataset from papers labeled by category names defined in an XML file presented in **Figure 16** in the following.

```
Computer Aided Design Tools #0.0
Cricuit and Logic Styles #0.3079
Portable Terminal Electronics #0.04752
```

Figure 12. First format between the outputs of the proposed approach.

```
D:\files\test\input-files\c1\122.htm--
Computer Aided Design Tools #0.7632
D:\files\test\input-files\c1\123.htm--
Computer Aided Design Tools #0.9386
D:\files\test\input-files\c1\137.htm--
Computer Aided Design Tools #0.7855
```

Figure 13. The sample outputs of second format.

```
<category>
  <ctgID>2</ctgID>
  <ctgName>Cricuit and Logic Styles</ctgName>
  <documents>
    <document>
      <filePath>D:\files\test\input-files\Cricuit and Logic Styles\Adiabatic
Logic Circuits\13</filePath>
      <relevantToCategory>0.08555848705352052</relevantToCategory>
    </document>
    <document>
      <filePath>D:\files\test\input-files\Cricuit and Logic Styles\Adiabatic
Logic Circuits\14</filePath>
      <relevantToCategory>0.07929412363572458</relevantToCategory>
    </document>
  </documents>
</category>
```

Figure 14. Third format of the outputs of the proposed approach.

```
<?xml ...?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-
ns#"    xmlns:sioc="http://rdfs.org/sioc/ns#"
xmlns:skos="http://www.w3.org/2004/02/skos/core#">
<sioc:post rdf:about="D:\files\test\input-files\Computer Aided
Design Tools\Power Analysis Techniques\12">
<sioc:topic>
<skos:Concept rdf:about="http://example.org/skos/concepts#Low
Voltage Technologies and Criuits">
<skos:prefLabel xml:lang="en">Low Voltage Technologies and
Criuits</skos:prefLabel>
</skos:Concept>
</sioc:topic>
</sioc:post>
```

Figure 15. Fourth format of the outputs of the proposed approach.

```
<categories>
 <category>
  <ctgID>1</ctgID>
  <name> Computer Aided Design Tools</name>
 </category>
 .....
 </categories>
```

Figure 16. Prepared dataset of categories for training phase of the approach.

```
<files>
 <file>
  <catID>1</catID>
<relativePath>cadt\241422.htm</relativePath>
 </file>
 ....
 </files>
```

Figure 17. Inside of each of the categories collected in the prepared dataset.

Also each paper's category is defined in the standard XML as in **Figure 17**.

Where "*relativePath*" defines the folder in which the paper's file exists.

The initial simple Dataset had about 176 papers files for the training phase in seven categories and also 38 files as test phase's inputs. The training phase time for this dataset was about 50.9913 minutes and for test phase, 24.586 minutes in a system with a core(TM)2 duo CPU 2.5 GHz, 4 GB Ram. The time is high in comparing to the other computer processing approaches, since this approach uses web services and so this time is highly dependent to the internet connection speed, mainly requesting/responding speed.

The precision of the approach after test phase is about totally: 72%. It means that if we had tested 100 papers for categorizing them, our semantic approach would guess the category of 72 of them accurately. This precision is highly dependent of the number of papers used for learning in the training phase. The more papers in training phase, the more extracted terms in the database, and consequently more papers categorized in correct categories.

More terms in a category is very useful in predicting a test paper's correct research filed, since with more terms, we know more about a field and important keywords used in papers on it.

In the evaluation running of the tool, we had 176 paper files. Number of papers, by each category, is presented in **Figure 18**.

The precision in each category is also calculated during processing. Partial precisions and recalls have been shown in **Figure 19**. For an exact and detailed definition of Precision and Recall metrics, especially in our proposed approch's evaluation steps, you may see formulas (4) and (6) under Section 5.2.1.

These results can be improved if we use WordNet based stemming. We have implemented this stemmer also, but due to high time of it in processing, we had not used it in this step.

The Terms/Keywords have been extracted from all training papers have either one word, two words, three words or more that three words. Each of these term types' count is calculated after Training phase completion. **Figure 20** shows results of this partial calculation. We can see that 3-word and more than three-word terms have a little difference in their growth patterns in comparison to the total count and the other term types' count. For a sample, category: "Circuit and Logic Styles" has 187 3-word terms in comparing to 152 term count of category: "Computer Aided Design Tools" (187 > 152), but for 1-word terms, it is 109 in comparing to 129 (but 109 < 129).

Also more exact numbers of **Figure 20** are presented in **Table 3**.

5.2. Comparison with Conventional Algorithms

Our approach is mainly based on the extraction algorithm steps. In the above statistics, you can see results of our used semantic approach with the following steps:

- Extracting concept terms of papers' abstract by using NLP approaches, such as: AlchemyAPI (defined in above);

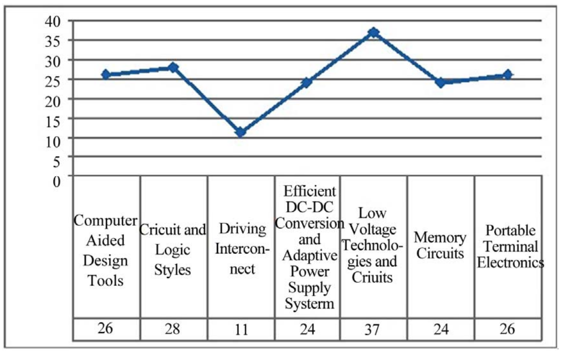


Figure 18. Number of papers used in the training phase of our experiment.

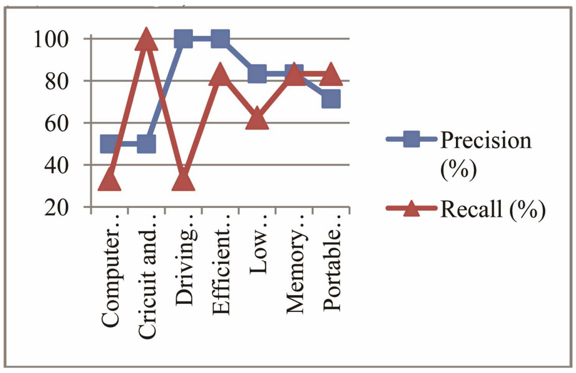


Figure 19. Precision & recall in each category.

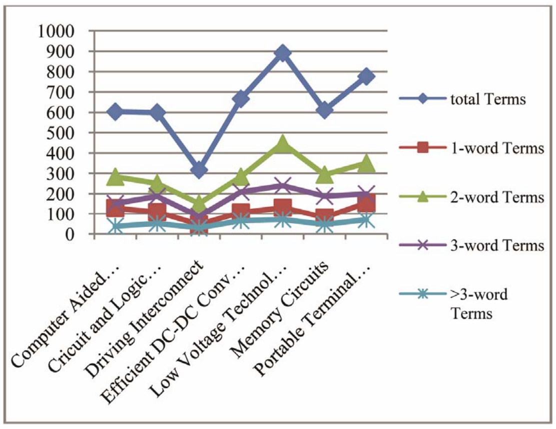


Figure 20. Count of each term type in each category in algorithm LPTC evaluation.

Table 3. LPTC's extracted terms count.

Category	1-Word Terms	2-Word Terms	3-Word Terms	>3-Word Terms	All Extracted Terms (ET)
Computer Aided Design Tools	129	284	152	38	603
Cricuit and Logic Styles	109	251	187	52	599
Driving Interconnect	47	152	88	30	317
Efficient DC-DC Conversion ···	107	284	209	66	666
Low Voltage Technologies ···	131	448	240	72	891
Memory Circuits	82	294	188	47	611
Portable Terminal Electronics	155	351	199	71	776
Total	760	2064	1263	376	4463

- AlchemyAPI uses knowledge bases, e.g. DBPedia, for better linking of extracted terms with their relevant semantic information and data;
- Stemming extracted terms for more correct comparing in 2 phases, Simple OR by helping from external knowledge such as WordNet.

So our used approach is a semantic enabled one. But there are many other terms extraction algorithms. We have evaluated 6 of most famous between them: C-Value [30], TfIdf used in [31,32], Basic (Simple) Frequency, GlossEx [33,34], TermsEx [35], Weirdness [36].

Thank to changing of source codes of these 6 algorithms implemented in JATE[3] (Java Automatic Term Extraction) toolkit and making them sufficient to our need in evaluation and comparison of them with our presented approach, we had a simple task in compare to implementing them from start by ourselves. However, decompiling codes of this toolkit, finding needed files location definition in the codes (such as text file of Stop Words path and BNC corpus file path), and other needed java classes for running each algorithm separately, finding main class/function of each algorithm separately and changing input parameter types of it to both our text papers' file paths lists and single file path, took 1 month in evaluation part of our presented approach times.

Our developed tool logs almost all events during its execution. It also saves the results of its two phases (Training and Test phase) into their sufficient Txt or CSV files. Therefore the running results can be viewed by other applications such as Excel. The results save into separate files for different algorithms (6 JATE algorithms and our presented LPTC algorithm). These aftermath statistics are shown in the following figures. At the end of training phase execution, which its main part is Terms extraction of input papers' abstract, we have had log files containing number of Terms (Terms count) extracted from abstracts by each algorithm. **Table 4** shows these values for C-Value algorithm. All 5 other algorithms, as implemented in JATE toolkit, extract equal number of Terms (see **Table 5** for number of Terms extracted by them). In these two tables, Terms count is separated by Terms type. This type is detected by number of words in the Term and there are 4 types: 1-word Terms (e.g. "*optimization*" or "*resynchronization*"), 2-word Terms (e.g. "*power system*" or "*lower leakage*" or "*multigrid-based technique*"), 3-word Terms (e.g. "*benchmark microprocessor chip*" or "*software power model*" or "*optimization and simulation*" or "*large-scale powersupply network*"), ">3-word" Terms (e.g. "*real and reactive power optimization*" or "*hierarchical power supply distribution model*" or "*signal-to-interference ratio power control technique*").

Execution times have been logged also. **Table 6** shows them in minutes. Most of values in this table are times (minutes) taken in Terms Extraction module. For example, CValue running time in extracting terms of category1 (named "Computer Aided Design Tools") is 9.078 minutes (9 minutes and about 4 seconds) and of category2 ("Circuit and Logic Styles") is 4.531 minutes and so on. CValue term extraction module has taken 37.782 minutes totally (sum of times spent in extracting terms of all categories). Also total running time of training phase for CValue algorithm (including processing abstracts time, extracting terms and inserting into database time, union of all categories' terms time) is 694 minutes.

By focusing on total TE (Term Extraction) times of the algorithms in **Table 6**, we can conclude that 3 last algorithms (GlossEx, TermEx, and Weirdness) totally need more minutes in extracting terms of the categories. It is because of this fact that these 3 algorithms use a reference corpus beside of the target corpus (papers collection of a category) for comparing the term frequency in both. Furthermore TermEx algorithm takes much time (54.593 minutes) than GlossEx and Weirdness since it additionally takes into account a atypical dimension called "domain consensus" which captures domain concepts with high frequencies within small subset of the corpus (e.g., single document) but are completely absent in the remainder of the corpus [37].

Evaluation Results of Test Phase

Precision is the most important factor in evaluating of a classifier algorithm. Precision is the proportion of returned documents that are targets. For a sample category i in a classification problem, it is calculated as following formula (4).

$$\text{Precision} = \frac{TP_i}{TP_i + FP_i} \tag{4}$$

While TP_i is number of papers labeled with category_i by both the expert (*i.e*. Prof. Chandrakasan in our case as explained above) and classifier (*i.e*. our proposed LPTC classifier tool). In other words, percent of papers labeled correctly in a category is precision of that category. This measure is calculated for each category tested in each algorithm separately. Precision of all algorithms, one by one and in each category, is shown in **Figure 21** (in scale of 1). Total precision of each algorithm is presented in **Figure 21** (in scale of percentage or 100).

If you want to know more about the other factors in formula (4), see **Table 7**.

FN_i is False Negative and identifies papers labeled with category_i by the expert but identified belonging to another category_j for $\forall j \in C, j \neq i$ in which C is set of all categories. In reality, we have: $FN_i = PRP_i - TP_i$ that PRP_i is number of all papers expert based labeled with category_j. Like it, TN_i is True Negative and

[3]code.google.com/p/jatetoolkit

Table 4. CValue extracted terms count.

Category Name	1-Word Terms	2-Word Terms	3-Word Terms	>3-Word Terms	All Extracted Terms (ET)	Terms in papers' abstract (all)	ET/All
Computer Aided Design Tools	165	264	169	102	700	1900	0.36842105
Circuit and Logic Styles	179	273	198	95	745	2316	0.3216753
Driving Interconnect	103	190	95	45	433	1084	0.39944649
Efficient DC-DC Conversion…	184	330	205	119	838	2588	0.32380216
Low Voltage Technologies…	253	477	248	161	1139	3288	0.34641119
Memory Circuits	165	292	175	91	723	2417	0.29913115
Portable Terminal Electronics	260	407	231	155	1053	2793	0.37701396
Total	1309	2233	1321	768	5631	16386	0.34364702

Table 5. All other 5 algorithms (TF-IDF, simple Frequency, GlossEx, TermEx, Weirdness) extracted terms count.

Category Name	1-Word Terms	2-Word Terms	3-Word Terms	>3-Word Terms	All extracted terms (ET)	Terms in papers' abstract (all)	ET/All
Computer Aided Design Tools	213	288	178	104	783	1900	0.412105
Circuit and Logic Styles	223	304	202	97	826	2316	0.356649
Driving Interconnect	131	204	98	45	478	1084	0.440959
Efficient DC-DC Conversion and Adaptive Power Supply System	223	370	207	119	919	2588	0.3551
Low Voltage Technologies and Circuits	308	526	261	163	1258	3288	0.382603
Memory Circuits	207	320	179	91	797	2417	0.329748
Portable	327	462	237	155	1181	2793	0.422
Terminal Electronics							843
Total	1632	2474	1362	774	6242	16386	0.380935

Table 6. Algorithms' running times (minutes) of both total training and in only Term Extraction (TE) module for each category.

Algorithm	Category 1	Category 2	Category 3	Category 4	Category 5	Category 6	Category 7	Total TE time	Total running time
CValue	9.078	4.531	2.313	4.907	6.75	4.703	5.5	**37.782**	**694**
TF-IDF	3.625	4.516	2.25	4.672	6.094	4.859	5.172	**31.188**	**685.688**
Simple Frequency	3.656	4.296	2.516	4.703	6.125	4.563	5.406	**31.265**	**682.765**
GlossEx	7.203	7.812	5.156	8.359	9.5	7.938	8.469	**54.437**	**711.188**
TermEx	6.953	7.422	5.5	8.234	9.781	7.875	8.828	**54.593**	**710.375**
Weirdness	6.703	7.766	5.188	8.313	9.875	7.75	8.656	**54.251**	**704.984**
Sum of All Algorithms	**37.218**	**36.343**	**22.923**	**39.188**	**48.125**	**37.688**	**42.031**	**263.516**	**4189**
Files #	**26**	**28**	**11**	**12**	**37**	**24**	**26**	**164**	**164**

shows papers labeled with category_j for $\forall j \in C, j \neq i$. If we suppose that $AllPR$ is all test papers in all categories, we can write:

$$TN_i = AllPR - PRP_i - FP_i \quad (5)$$

Always beside of precision in classification problems, we see another measure called recall. In a good summary definition, Recall is the proportion of target documents returned. It is calculated by formula in (6). Comparative recalls for each algorithm in each category are shown in **Figure 22** (in scale of 1). Also totally obtained recall for each is shown in **Figure 23** (in scale of percentage or 100).

$$\text{Recall} = \frac{TP_i}{TP_i + FN_i} \quad (6)$$

You see that number of Terms extracted by our approach is lowest comparing to the other algorithms (**Figure 24**), but precision and also recall of it in guessing correct category of an input test paper, is in highest position (see **Figures 23** and **25**).

So a situation against to "more extracted terms, higher precision"rule presented before, is occurred. This is because of just one important factor is existed in our approach but not in the other Term Extraction algorithms: semantic extracting of Concept Terms. In the other ap-

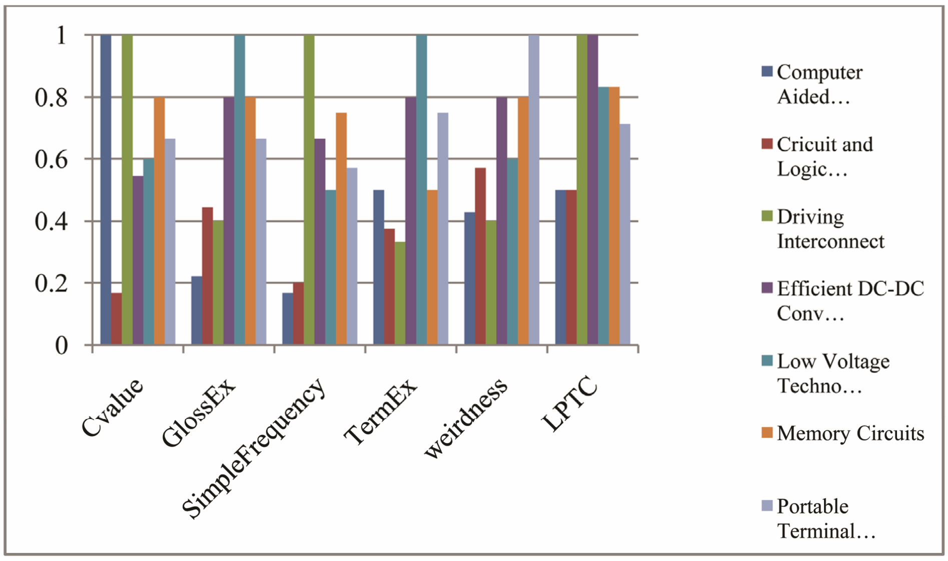


Figure 21. Precision of each algorithm per category.

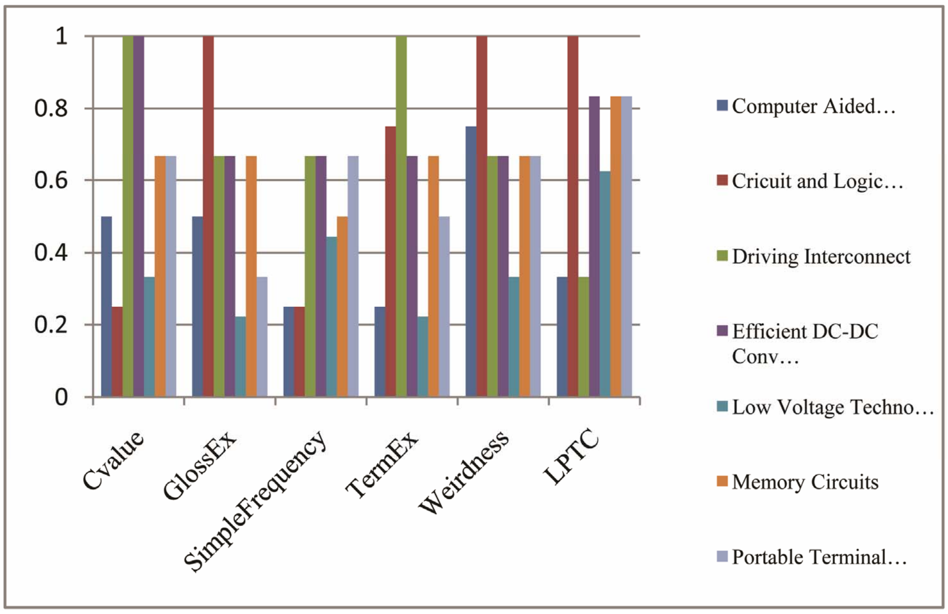


Figure 22. Recall of each algorithm per category.

Table 7. Different resulted sets of test papers.

Category		Expert judgement	
		TRUE	FALSE
Classifier judgement	TRUE	TP_i	FP_i
	FALSE	FN_i	TN_i

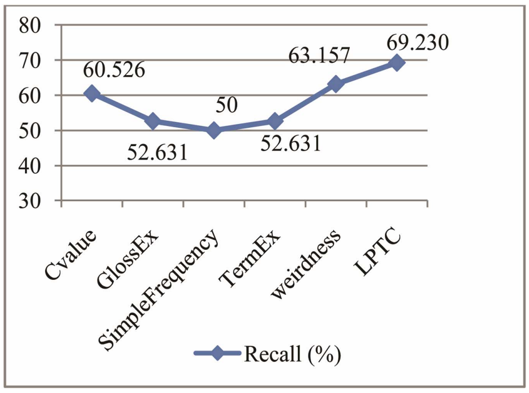


Figure 23. Total recall in each algorithm.

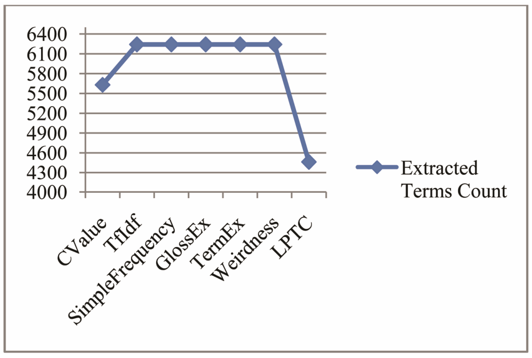


Figure 24. Number of terms extracted by each algorithm.

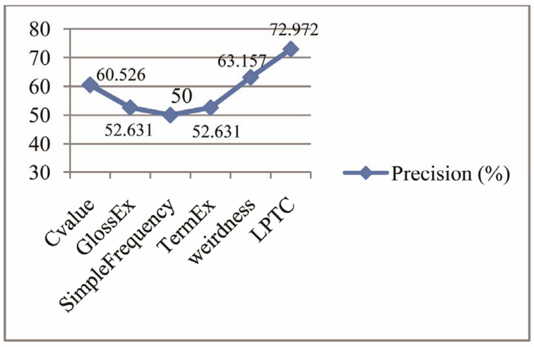


Figure 25. Precision of correct category guessing in each algorithm.

proaches, as explained above, terms are extracted only by their frequency and document frequency, in different ways. So there is no help of external resources in them. In other words, in our presented approach, external semantic resources, such as knowledge bases in Linked Data, helps in determination of important and key concept Terms and also field terminology of the paper. For a comparison sample, suppose we have no occurrence of term “Transistor”[4] totally or just 1 occurrence in only one of our input abstracts. This term is famous in low power field. Although our semantic NLP presented method can extract this as a concept term, It cannot be consequently detected by frequency based term extraction algorithms, like JATE algorithms. As a result of this advantage in our approach, a higher percentage is obtained, although the number of terms is low. The above disadvantage is existed for terms with high document frequency (occured in almost all abstracts) since JATE algorithms (e.g. TfIdf) are not able to detect these types of terms as important key-words.

6. Conclusions and Future Work

We saw that an automatic approach, like proposed Low-Power Themes Classifier (LPTC), can end the high time and resources consuming process of manual labeling a bunch of papers. Because of some unavoidabe disadvantages in the current NER methods, such as CValue, TfIdf, GlossEx, TermEx, Weirdness that are mainly based on Natural Language Processing (NLP) techniques and classifiers, a semantic approach based on RDF, FOAF and ontologies was proposed. Since papers’ terms are extracted semantically in our approach, highest precision (among the other Term-Extraction algorithms) in guessing correct category of a test paper is obtained, although it has smallest number of extracted terms. Higher precision compring to another classifiying approaches is also because of usage of training categories prepared by an expert in our sample specific field (e.g. Electronic).

The proposed approach is mainly based on concept terms, represented in RDF documents, in abstract of a research paper and will generate semantic enabled outputs are useful for complementing knowledge bases and datasets of Linked Data cloud. We also showed that our proposed approach’s precision is 72% that is at least 9% better than other algorithms. In addition, we can say that higher number of papers for training, even higher precision we have.

The time is almost high for our approach, so an improved performance can be a start point for a further work. Also by the output type 4 described above which is based on RDF standard, we can have many new ideas, such as querying these RDF documents with SPARQL for finding guessed category of a document tested with this approach already, and adding interests and co-authors of scientific researchers to Linked Data DBLP dataset. This

[4]A transistor is a semiconductor device used to amplify and switch electronic signals and electrical power [http://en.wikipedia.org/wiki/Transistor].

tool can be used in guessing not only category of a document or paper but also conceptual topic of a text or an HTML web page (a good way for search engines to have better results based on semantic analysis).

Another using of this approach is in the multi-Fields criteria, and emerging of the research areas' subtopics and concepts.

REFERENCES

[1] A. P. Chandrakasan and R. W. Brodersen, "Low Power Digital CMOS Design," Norwell, Kluwer, 1995. doi:10.1007/978-1-4615-2325-3

[2] S. Aseervatham and Y. Bennani, "Semi-Structured Document Categorization with a Semantic Kernel," *Pattern Recognition*, Vol. 42, No. 9, 2009, pp. 2067-2076. doi:10.1016/j.patcog.2008.10.024

[3] N. Fuhr, S. Hartmann, G. Knorz, G. Lustig, M. Schwantner and K. Tzeras, "AIR/X-a Rule Based Multistage Indexing System for Large Subject Fields," *Proceedings of Riao*'91, 1991, pp. 606-623.

[4] Y. Yang and C. G. Chute, "An Example-Based Mapping Method for Text Categorization and Retrieval," *ACM Transactions on Information Systems* (*TOIS*), Vol. 12, No. 3, 1994, pp. 252-277. doi:10.1145/183422.183424

[5] D. D. Lewis and M. Ringuette, "A Comparison of Two Learning Algorithms for Text Categorization," *Third Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, 11-13 April 1994, pp. 81-93.

[6] K. Tzeras and S. Hartmann, "Automatic Indexing Based on Bayesian Inference Networks," ACM Press, New York City, 1993, pp. 22-35.

[7] E. Wiener, J. Pedersen and A. Weigend, "A Neural Network Approach to Topic Spotting," *Proceedings of SDAIR*-95, 4*th Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, 1995, pp. 317-332.

[8] W. W. Cohen and Y. Singer, "Context-Sensitive Learning Methods for Text Categorization," *ACM Transactions on Information Systems* (*TOIS*), Vol. 17, No. 2, 1999, pp. 141-173. doi:10.1145/306686.306688

[9] D. D. Lewis, R. E. Schapire, J. P. Callan and R. Papka, "Training Algorithms for Linear Text Classifiers," ACM Press, New York City, 1996, pp. 298-306.

[10] C. Apté, F. Damerau and S. M. Weiss, "Towards Language Independent Automated Learning of Text Categorization Models," *Proceedings of the* 17*th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Dublin, Springer-Verlag, New York, 1994, pp. 23-30.

[11] I. Moulinier, G. Raskinis and J. G. Ganascia, "Text Categorization: A Symbolic Approach," *Proceedings of the Fifth Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, April 1996, pp. 87-99.

[12] R. H. Creecy, B. M. Masand, S. J. Smith and D. L. Waltz, "Trading MIPS and Memory for Knowledge Engineering," *Communications of the ACM*, Vol. 35, No. 8, 1992, pp. 48-64. doi:10.1145/135226.135228

[13] Y. Yang, "Expert Network: Effective and Efficient Learning from Human Decisions in Text Categorization and Retrieval," *Proceedings of the* 17*th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Dublin, Springer-Verlag, New York, 1994, pp. 13-22.

[14] C. W. Hsu and C. J. Lin, "A Comparison of Methods for Multiclass Support Vector Machines," *IEEE Transactions on Neural Networks*, Vol. 13, No. 2, 2002, pp. 415-425. doi:10.1109/72.991427

[15] C. Staelin, "Parameter Selection for Support Vector Machines," Hewlett-Packard Company, Palo Alto, 2003.

[16] L. H. Lee, C. H. Wan, R. Rajkumar and D. Isa, "An Enhanced Support Vector Machine Classification Framework by Using Euclidean Distance Function for Text Document Categorization," *Applied Intelligence*, Vol. 37, 2012, pp. 80-99.

[17] T. Y. Wang and H. M. Chiang, "One-Against-One Fuzzy Support Vector Machine Classifier: An Approach to Text Categorization," *Expert Systems with Applications*, Vol. 36, No. 6, 2009 pp. 10030-10034. doi:10.1016/j.eswa.2009.01.025

[18] Q. Luo, E. Chen and H. Xiong, "A Semantic Term Weighting Scheme for Text Categorization," *Expert Systems with Applications*, Vol. 38, No. 10, 2011, pp. 12708-12716. doi:10.1016/j.eswa.2011.04.058

[19] Z. Li, Z. Xiong, Y. Zhang, C. Liu and K. Li, "Fast Text Categorization Using Concise Semantic Analysis," *Pattern Recognition Letters*, Vol. 32, No. 3, 2011, pp. 441-448. doi:10.1016/j.patrec.2010.11.001

[20] B. Yu, Z.-B. Xu and C.-H. Li, "Latent Semantic Analysis for Text Categorization Using Neural Network," *Knowledge-Based Systems*, Vol. 21, No. 8, 2008, pp. 900-904. doi:10.1016/j.knosys.2008.03.045

[21] W. J. Wilbur and K. Sirotkin, "The Automatic Identification of Stop Words," *Journal of Information Science*, Vol. 18, No. 1, 1992, pp. 45-55. doi:10.1177/016555159201800106

[22] F. Zarrinkalam and M. Kahani, "A New Metric for Measuring Relatedness of Scientific Papers Based on Non-Textual Features," *Intelligent Information Management*, Vol. 4, No. 4, 2012, pp. 99-107. doi:10.4236/iim.2012.44016

[23] D. M. Bikel, S. Miller, R. Schwartz and R. Weischedel, "Nymble: A High-Performance Learning Name-Finder," *Proceedings of the* 5*th Conference on Applied Natural Language Processing*, Washington DC, April 1997, pp. 194-201.

[24] A. Borthwick, J. Sterling, E. Agichtein and R. Grishman, "Exploiting Diverse Knowledge Sources via Maximum Entropy in Named Entity Recognition," *Proceedings of the* 6*th Workshop on Very Large Corpora*, 1998, pp. 152-160.

[25] E. Miller, "An Introduction to the Resource Description Framework," *Bulletin of the American Society for Information Science and Technology*, Vol. 25, No. 1, 1998, pp. 15-19. doi:10.1002/bult.105

[26] J. Golbeck and M. Rothstein, "Linking Social Networks

on the Web with Foaf: A Semantic Web Case Study," *Proceedings of the* 23*rd national Conference on Artificial Intelligence*—*Vol.* 2, AAAI Press, Chicago, 2008, pp. 1138-1143.

[27] T. R. Gruber, "A Translation Approach to Portable Ontology Specifications," *Knowledge Acquisition*, Vol. 5, No. 2, 1993, pp. 199-220. doi:10.1006/knac.1993.1008

[28] M. D'Aquin and N. F. Noy, "Where to Publish and Find Ontologies? A Survey of Ontology Libraries," *Web Semantics*: *Science*, *Services and Agents on the World Wide Web*, Vol. 11, 2011, pp. 96-111. doi:10.1016/j.websem.2011.08.005

[29] M. Ley, "The DBLP Computer Science Bibliography: Evolution, Research Issues, Perspectives String Processing and Information Retrieval," Springer, Berlin and Heidelberg, 2002.

[30] K. Frantzi, S. Ananiadou and H. Mima, "Automatic Recognition of Multi-Word Terms: The C-value/NC-Value Method," *International Journal on Digital Libraries*, Vol. 3, No. 2, 2000, pp. 115-130. doi:10.1007/s007999900023

[31] D. A. Evans and R. G. Lefferts, "Clarit-Trec Experiments," *Information Processing & Management*, Vol. 31, No. 3, 1995, pp. 385-395. doi:10.1016/0306-4573(94)00054-7

[32] O. Medelyan and I. H. Witten, "Thesaurus Based Automatic Keyphrase Indexing," *Proceedings of the* 6*th ACM/IEEE-CS Joint Conference on Digital Libraries*, ACM, Chapel Hill, 11-15 June 2006, pp. 296-297.

[33] L. Kozakov, Y. Park, T. Fin, Y. Drissi, Y. Doganata and T. Cofino, "Glossary Extraction and Utilization in the Information Search and Delivery System for IBM Technical Support," *IBM Systems Journal*, Vol. 43, No. 3, 2004, pp. 546-563. doi:10.1147/sj.433.0546

[34] Y. Park, R. J. Byrd and B. K. Boguraev, "Automatic Glossary Extraction: Beyond Terminology Identification," *Proceedings of the* 19*th International Conference on Computational Linguistics*, Association for Computational Linguistics, Taipei, Vol. 1, 2002, pp. 1-7.

[35] F. Sclano and P. Velardi, "Termextractor: A Web Application to Learn the Common Terminology of Interest Groups and Research Communities," 9*th Conference on Terminology and Artificial Intelligence* (*TIA* 2007), Sophia Antipolis, 8-9 October 2007.

[36] K. Ahmad, L. Gillam and L. Tostevin, "University of Surrey Participation in TREC8: Weirdness Indexing for Logical Document Extrapolation and Retrieval (WILDER)," *Proceedings of the* 8*th Text Retrieval Conference* (*TREC*-8), 1999.

[37] Z. Zhang, J. E. Iria, C. Brewster and F. Ciravegna, "A Comparative Evaluation of Term Recognition Algorithms," *Proceedings of the* 6*th International Conference on Language Resources and Evaluation* (*LREC*08), Marrakech, 2008.