J. Intell. Syst. **20** (2011), 327–351
DOI 10.1515/JISYS.2011.017

Deep Text Mining for Automatic Keyphrase Extraction from Text Documents

Muhammad Abulaish, Jahiruddin and Lipika Dey

Abstract. Due to existence of a huge amount of textual data either on the World Wide Web or in textual databases like PubMed, the development of novel automatic keyphrase extraction methods has emerged as one of the key research problems in recent past. Consequently, a number of machine learning techniques, mostly supervised, have been proposed to extract keyphrases from text documents. But, one of the main bottlenecks that hinders the success of such systems is the requirement of annotated corpora for training purpose. In this paper, we propose the design of a deep text mining system to identify keyphrases in text documents that are either unstructured or semi-structured in nature. The novelty of our system lies in its applicability on a single document, instead of demanding a collection of annotated texts for training, to identify keyphrases embedded within it. The proposed system applies parsing techniques to identify candidate phrases. After mapping the original set of candidate phrases into a low-dimensional space using Singular Value Decomposition (SVD), the Markov Clustering (MCL) technique is applied to cluster related sentences together. Finally, considering each cluster as a document, Latent Dirichlet Allocation (LDA) is applied to identify feasible keyphrases that are presented to users in non-increasing order of their relevance score values. The efficacy of the proposed system is established through experimentation on datasets from two different domains. On comparative evaluation, we found that the proposed system outperforms `KEA` and `KEA++` that apply the supervised machine learning approach for automatic keyphrase extraction from text documents.

Keywords. Text Mining, Natural Language Processing, Keyphrase Extraction, Latent Dirichlet Allocation, Markov Clustering.

2010 Mathematics Subject Classification. 68T50, 68U15.

1 Introduction

Due to availability of voluminous textual data either on the World Wide Web (WWW) or in textual databases like PubMed, a good number of researches have been directed towards extracting keyphrases from text documents and store them in structured formats for further analysis. Keyphrases (a.k.a. *key-concepts* or *keywords*) generally contain one or more words and provide a conceptualization of a document collection to grasp the main theme and comprehend the content of

the collection without navigating through the pile of documents. Keyphrases are useful for various applications such as document summarization [5], document categorization and clustering [11, 12], digital libraries, and Web search engines. In digital libraries, the keyphrases of a scientific paper can help users to get a rough sense of it, whereas in Web search the keyphrases embedded within web pages can serve as metadata for indexing and retrieving them for user supplied queries. Keyphrases are also helpful to expand user queries, facilitate document skimming by visually emphasizing important phrases; and offer a powerful mean of measuring document similarity that can be exploited to group them into different categories. Recently, the Microsoft academic research portal has also introduced the concept of using keywords for tags cloud generation to describe people, conferences, journals, etc. that are being indexed by this portal. For example, while supplying "IICAI" as a query term to the Microsoft academic research portal, the extracted descriptive keywords are *artificial intelligent*, *artificial neural network*, *data mining*, etc. that are helpful to know the area of researches covered by IICAI conferences without exploring the pile of web pages and previously published research papers.

Keyphrases are particularly useful because they can be interpreted individually and independently of each other. Keyphrases are usually chosen manually, which is a labor-intensive task. In scientific publications, generally authors assign keyphrases to documents they have written, whereas professional indexers often choose phrases from a predefined controlled vocabulary relevant to the domain at hand. Since many documents do not have manually assigned keyphrases, the development of a tool to automatically assign keyphrases to the documents would have a potential use. Moreover, the identification and extraction of keyphrases may be useful for inferring new facts and indexing text corpora for efficient query processing over text documents.

There are generally two different approaches for keyphrase extraction – the supervised learning approach and the unsupervised learning approach. The supervised keyphrase extraction algorithms model the keyphrase extraction process as a classification task. This approach requires a corpus of similar documents in which keyphrases are annotated. But, in many cases training corpuses are not always available and consequently, these algorithms constitute a major drawback. For example, if one encounters a new document, he/she might like to know quickly the main topics addressed. In this case, a keyphrase extraction system is needed that may apply to a single document to identify keyphrases embedded within it.

In this paper, we propose the design of a deep text mining system to identify keyphrases embedded within text documents that are either semi-structured or unstructured in nature. Starting with the identification of candidate phrases using Natural Language Processing (NLP) techniques, the proposed system applies a

series of statistical techniques to identify feasible keyphrases. Unlike supervised machine learning approaches that require a good number of annotated text corpora to tune the model before extracting keyphrases, the proposed method does not require any training data and it can be applied on a single document to identify keyphrases. A preliminary version of this work is [1].

The major advancement of this paper lies in the utilization of dependency relations between words of a sentence to filter-out irrelevant candidate phrases extracted from that sentence. In addition, instead of applying the k-means algorithm to group related sentences of a document together, we have applied Markov clustering which does not require the number of clusters as an input parameter from the user. The novelty of the proposed method over existing ones can be summarized as follows:

- The proposed method uses NLP and statistical techniques rather than supervised machine learning techniques, and consequently it does not require annotated corpora to tune the system for keyphrase extraction. It can be applied on a single document to identify keyphrases embedded within it.
- The proposed method does not use any domain-specific knowledge due to which it can be applied on text documents pertaining to any domain to mine keyphrases from them.
- In order to group related sentences together, the proposed method applies Markov clustering instead of the k-means algorithm in which determination of an ideal value for k is a major bottleneck.
- The proposed method outperforms the widely-used standard keyphrase extraction algorithm, `KEA`, and its successor, `KEA++`.

The remaining part of the paper is structured as follows. Section 2 presents a brief review of the current state-of-the-art in keyphrase extraction from text documents. In Section 3, we present the design and functional details of the proposed keyphrase extraction system. The experimental setup and the evaluation of the proposed system is presented in Section 4. We conclude the paper in Section 5.

2 State-of-the-Art in Keyphrase Extraction from Text Documents

In this section, we present an overview of some of the recent research efforts that have been directed towards the problem of keyphrase extraction and its applications to various domains. Two different types of keyphrase extraction methods (unsupervised and supervised) have been reported in literature. Supervised methods generally use a collection of annotated texts, in which keyphrases are already marked, to learn a classification model, and then use it to identify keyphrases in

new documents. On the other hand, unsupervised methods do not require a training dataset, rather they assign a numeric score to each candidate phrase, based on which top few phrases are declared as keyphrase. Krulwich and Burkey [14] proposed a heuristic approach, such as the use of italics, the presence of phrases in header section and the use of acronyms, to extract keyphrases from text documents. Barker and Cornacchia [3] developed a system, which uses the number of words and the frequency counts of noun phrases and head nouns to identify keyphrases. Tomokiyo and Hurst [22] proposed a keyphrase extraction method based on statistical language models to extract keyphrases. Their method uses point-wise KL-divergence between multiple language models for scoring both *phraseness* and *informativeness* of phrases. Phraseness and informativeness are two features of a keyphrase. Phraseness describes the degree to which a given word sequence is considered to be a phrase, whereas informativeness describes how well a phrase captures the key ideas in a document collection. Mihalcea and Tarau [17] proposed a graph-based ranking model, `TextRank`, to rank keywords. In this method, a keyword is ranked based on the co-occurrence links between words. An important aspect of `TextRank` is that it does not require deep linguistic knowledge nor domain or language specific annotated corpora, which makes it highly portable to other domains, genres, or languages. Nguyen and Phan [18] proposed an ontology based system to extract keyphrases from Vietnamese text documents. They use the Vietnamese Wikipedia as ontology and specific characteristics of the Vietnamese language for keyphrase selection.

Although, most of the works on this problem followed the corpus-based approach, there are single document based approaches as well, e.g., [7]. Wan and Xiao [25] proposed a novel approach to extract keyphrases from a single document by leveraging the neighborhood knowledge of the specific documents. They reported the *f-score* of the system as 31.6% for the expand rank method with 10 neighborhood values. Pasquier [20] proposed a system that uses sentence clustering and Latent Dirichlet Allocation (LDA) to extract keyphrases from a single text document. The best performance of this system is achieved with dimension reduction by using Non-negative Matrix Factorization (NMF), keeping the number of clusters more than 20, and by using the exponential function in ranking of keyphrases after LDA. The *f-score* of this system is reported as 14.7%.

Besides unsupervised keyphrase extraction methods, a number of supervised methods have been also proposed by different researchers. In this approach, generally a training dataset is used to train a classification model, which is then used to classify a candidate phrase either as a keyphrase or as a non-keyphrase. Turney [23] proposed a supervised learning system, `GenEx`, to extract keyphrases from text documents. `GenEx` is designed based on a set of parameterized heuristic rules that are fine-tuned to the training corpus using a genetic algorithm. The exper-

imental results support the claim that a specialized learning algorithm (GenEx) can generate better keyphrases than a general-purpose learning algorithm (C4.5) and the non-learning algorithms that are used in commercial software. Turney has also proposed a machine learning approach to extract keyphrases [24]. In this algorithm, he used nine features to identify a candidate phrase, which includes positional information of the phrase in the document, frequency of the phrase and its component words, phrase length in terms of words and characters, whether the phrase ends in a final adjective, whether it contains a common verb and whether the phrase is a proper noun. Keyphrases are extracted from candidate phrases based on examination of their features. All the above features are calculated after stemming the phrases using the Lovins stemmer. Frank et al. [10] have implemented a system, KEA[1], which is similar to Turney's keyphrase extraction system. After hunting for numerous features, they settled on just two lexical features and later added another one. KEA treats keyphrase extraction as a supervised learning problem, but it uses a Bayesian approach instead of a genetic algorithm approach. KEA includes procedural domain knowledge in the form of a final post-processing operation. Their experiments indicate that KEA and GenEx have statistically equivalent levels of performance. They also claim that training naive Bayes learning technique is quicker than training GenEx which employs special-purpose genetic algorithm for training. The same group, Gutwin et al. [11], developed a new kind of search engine, Keyphind, designed specially to support browsing in which they included KEA. Their experiments suggest that certain kinds of tasks are much easier with Keyphind than with conventional search engines. Despite the small size of the feature set, due to its significance and simplicity, KEA is still comparable to most of the keyphrase extraction systems being developed nowadays.

Later on, an improved version, KEA++, was designed by Medelyan and Witten [15], which enhances KEA by introducing the concept of a domain specific knowledge base using *node degree* for semantic information about the phrases. It exploits the fact that the more a phrase is connected to others through any type of relation via thesaurus terms, the more relevant it becomes for the corresponding context. In [19], Nguyen and Kan tried to work especially for scientific publications. Their system focuses mainly on the structure of a scientific publication (i.e., with abstract followed by an introduction, related work and so on). They modeled the distribution of a keyphrase among different logical sections as a vector of feature's frequency values for 14 generic section headers and created a *maximum entropy* based classifier using four features to infer the generic section header from their list of 14 headers. Apart from structural features their feature set consists of baseline features of KEA, a binary feature value for acronym and linguistic

[1] http://www.nzdl.org/Kea/

features as POS sequence and sequence of morphological suffixes. The feature set of Medelyan et al. [16] contains one additional feature compared to KEA++, namely the *document-specific keyphraseness*, a knowledge-based feature that represents the likelihood of a phrase being a link in the Wikipedia corpus. According to Medeleyan et al. [16], most important phrases are those that appear both in the beginning as well as at the end, and in Maui they introduced an additional lexical feature and two knowledge-based Wikipedia features. *Inverse Wikipedia linkage* is one among them that harnesses information of incoming links to the most likely Wikipedia article for a given candidate phrase to spotlight those phrases that are referred by other commonly used concepts. In [26], Zhang et al. performed this work for multi-word extraction using augmented mutual information. Sarkar [21] has proposed a supervised machine learning method based on KEA to extract keyphrases from biomedical documents. He used the GENIA tagger to parse text documents for candidate phrase extraction. On evaluation, his system outperforms KEA for biomedical documents. SZTERGAK, a recent work by Berend and Farkas [4], discovered some interesting and worthy features to be exploited for the keyphrase extraction task.

3 Proposed Keyphrase Extraction System

In this section, we present the design of the proposed keyphrase extraction system to extract keyphrases from text documents. The proposed system is characterized by the following key functionalities – *document pre-processing and parsing*, *candidate phrase extraction*, *dimensionality reduction*, *phrase clustering using MCL*, and *keyphrase identification using LDA*. Figure 1 presents the architecture of the proposed keyphrase extraction system, which consists of a separate module for each of the above-mentioned functionalities. Functional detail about each module is presented in the following sub-sections.

3.1 Document Pre-Processing and Parsing

The input to this module is a text document from which keyphrases are to be extracted. Initially, the documents are cleaned through filtering meta language tags and unwanted texts like authors' names and affiliations, references, etc. The cleaned documents are tokenized into record-size chunks, boundaries of which are decided heuristically on the basis of the presence of various punctuation marks. Depending on the application, a record-size chunk may contain a sentence, paragraph, or a complete document. Thereafter, the documents are parsed using a parser that assigns Parts-Of-Speech (POS) tags to every word in a sentence, where

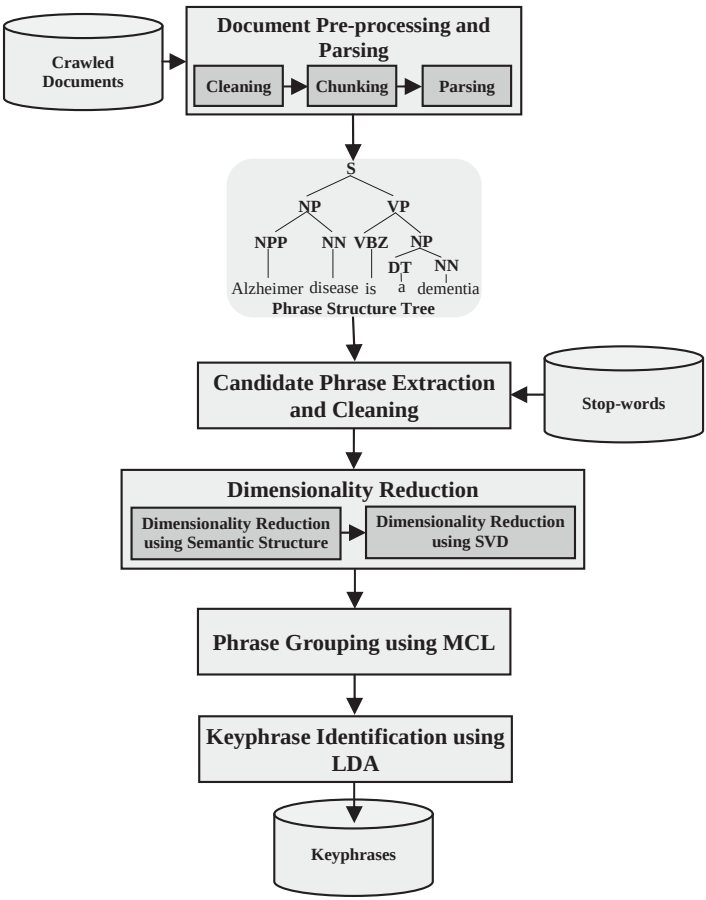


Figure 1. Architecture of the proposed keyphrase extraction system.

a tag reflects the syntactic category of the word [2]. POS analysis plays an important role in text information extraction since the syntactic category of a word determines its role in a sentence to a large extent. The POS tags are useful to identify the grammatical structure of sentences like noun and verb phrases and their interrelationships. For document parsing, we have used the Stanford parser[2], which is a statistical parser. The Stanford parser receives documents as input and works out the grammatical structure of sentences to convert them into equivalent phrase structure and dependency tree. A list of sample sentences along with their phrase structure and dependency trees generated by Stanford parser is shown in Table 1.

[2] `http://nlp.stanford.edu/downloads/lex-parser.shtml`

Sent. no.	Sentence	Phrase structure tree	Dependency tree
S1	IL-2 gene expression and NF-kappa B activation through CD28 requires reactive oxygen production by 5-lipoxygenase.	(ROOT (S (NP (NP (NN IL-2) (NN gene) (NN expression) (CC and) (JJ NF-kappa) (NNP B) (NN activation)) (PP (IN through) (NP (NNP CD28)))) (VP (VBZ requires) (NP (JJ reactive) (NN oxygen) (NN production)) (PP (IN by) (ADJP (JJ 5-lipoxygenase)))) (. .)))	nn(activation-7, IL-2-1) nn(activation-7, gene-2) nn(activation-7, expression-3)amod(activation-7, NF-kappa-5) nn(activation-7, B-6) nsubj(requires-10, activation-7) through(activation-7, CD28-9) amod(production-13, reactive-11) nn(production-13, oxygen-12) dobj(requires-10, production-13) by(requires-10, 5-lipoxygenase-15)
S2	Activation of the CD28 surface receptor provides a major costimulatory signal for T cell activation resulting in enhanced production of interleukin-2 (IL-2) and cell proliferation.	(ROOT (S (NP (NP (NNP Activation)) (PP (IN of) (NP (DT the) (NNP CD28) (NN surface) (NN receptor)))) (VP (VBZ provides) (NP (DT a) (JJ major) (NN costimulatory) (NN signal)) (PP (IN for) (NP (NP (NNP T) (NN cell) (NN activation)) (VP (VBG resulting) (PP (IN in) (NP (NP (JJ enhanced) (NN production)) (PP (IN of) (NP (NP (NP (NN interleukin-2)) (PRN (-LRB--LRB-) (NP (NNP IL-2)) (-RRB--RRB-))) (CC and) (NP (NN cell) (NN proliferation))))))))) (. .)))	nsubj(provides-7, Activation-1) det(receptor-6, the-3) nn(receptor-6, CD28-4) nn(receptor-6, surface-5) of(Activation-1, receptor-6) det(signal-11, a-8) amod(signal-11, major-9) nn(signal-11, costimulatory-10) dobj(provides-7, signal-11) nn(activation-15, T-13) nn(activation-15, cell-14) for(provides-7, activation-15) partmod(activation-15, resulting-16) amod(production-19, enhanced-18) in(resulting-16, production-19) of(production-19, interleukin-2-21) and(interleukin-2-21, IL-2-23) nn(proliferation-27, cell-26) and(interleukin-2-21, proliferation-27)
S3	Delineation of the CD28 signaling cascade was found to involve protein tyrosine kinase activity, followed by the activation of phospholipase A2 and 5-lipoxygenase.	(ROOT (S (NP (NP (NNP Delineation)) (PP (IN of) (NP (DT the) (NNP CD28) (NNP signaling) (NN cascade)))) (VP (VBD was) (VP (VBN found) (S (VP (TO to) (VP (VB involve) (NP (NP (NN protein) (NN tyrosine) (NN kinase) (NN activity)) (, ,) (UCP (VP (VBN followed) (PP (IN by) (NP (NP (DT the) (NN activation)) (PP (IN of) (NP (JJ phospholipase) (NN A2)))))) (CC and) (ADJP (JJ 5-lipoxygenase)))))))) (. .)))	nsubjpass(found-8, Delineation-1) det(cascade-6, the-3) nn(cascade-6, CD28-4) nn(cascade-6, signaling-5) of(Delineation-1, cascade-6) aux(found-8, was-7) aux(involve-10, to-9) ccomp(found-8, involve-10) nn(activity-14, protein-11) nn(activity-14, tyrosine-12) nn(activity-14, kinase-13) dobj(involve-10, activity-14) dep(activity-14, followed-16) and(followed-16, by-17) det(activation-19, the-18) dep(by-17, activation-19) amod(A2-22, phospholipase-21) of(activation-19, A2-22) and(followed-16, 5-lipoxygenase-24)
S4	Our data suggest that lipoxygenase metabolites activate ROI formation which then induce IL-2 expression via NF-kappa B activation.	(ROOT (S (NP (PRP$ Our) (NNS data)) (VP (VBP suggest) (SBAR (IN that) (S (NP (JJ lipoxygenase) (NNS metabolites)) (VP (VBP activate) (NP (NP (NNP ROI) (NN formation)) (SBAR (WHNP (WDT which)) (S (ADVP (RB then)) (VP (VB induce) (NP (NP (NN IL-2) (NN expression)) (PP (IN via) (NP (NNP NF-kappa) (NNP B) (NN activation)))))))))))) (. .)))	dep(data-2, Our-1) nsubj(suggest-3, data-2) amod(metabolites-6, lipoxygenase-5) nsubj(activate-7, metabolites-6) that(suggest-3, activate-7) nn(formation-9, ROI-8) dobj(activate-7, formation-9) dep(formation-9, which-10) advmod(induce-12, then-11) dep(which-10, induce-12) nn(expression-14, IL-2-13) dobj(induce-12, expression-14) nn(activation-18, NF-kappa-16) nn(activation-18, B-17) via(expression-14, activation-18)
S5	These findings should be useful for therapeutic strategies and the development of immunosuppressants targeting the CD28 costimulatory pathway.	(ROOT (S (NP (DT These) (NNS findings)) (VP (MD should) (VP (VB be) (ADJP (JJ useful) (PP (IN for) (NP (NP (JJ therapeutic) (NNS strategies)) (CC and) (NP (NP (DT the) (NN development)) (PP (IN of) (NP (NP (NNS immunosuppressants)) (VP (VBG targeting) (NP (DT the) (NNP CD28) (NN costimulatory)) (ADVP (RB pathway)))))))))))) (. .)))	det(findings-2, These-1) nsubj(useful-5, findings-2) aux(useful-5, should-3) aux(useful-5, be-4) amod(strategies-8, therapeutic-7) for(useful-5, strategies-8) det(development-11, the-10) and(strategies-8, development-11) of(development-11, immunosuppressants-13) partmod(immunosuppressants-13, targeting-14) det(costimulatory-17, the-15) nn(costimulatory-17, CD28-16) dobj(targeting-14, costimulatory-17) advmod(targeting-14, pathway-18)

Table 1. Sample sentences along with their phrase structure and dependency trees generated by the Stanford parser.

3.2 Candidate Phrase Extraction

Candidate phrase extraction is an important task for identification of keyphrases in text documents. In this phase, the phrase structure trees generated by the Stanford parser are analyzed to extract candidate phrases from them. For phrase extraction, we consider only those internal NP (noun phrase) nodes whose child nodes appear at leaf-level in the phrase structure tree. If NP has two or more child nodes then a *string concatenation* function is applied to club them together to generate a multi-words noun phrase, otherwise the single word is considered as a noun phrase. All phrases are cleaned by removing the stop-words[3] and the words having special symbols, from beginning as well as from end of the phrases. On multi-words phrases, we apply the n-gram technique, where the value of n is restricted to maximum 3, to generate smaller phrases, i.e., we generate all 1-, 2-, and 3-grams. These phrases are further cleaned and those representing noise are filtered out. A phrase is filtered out if it does not have English alphabets or it has some special symbols or its length is less than 3 characters. Algorithm 1, presents the candidate phrase extraction process in a formal way. A list of extracted candidate phrases from the phrase structure trees shown in Table 1 is given in Table 2.

3.3 Dimensionality Reduction

Since the document vectors constructed by using the weights of the extracted candidate phrases are sparse and of high dimensions as number of candidate phrases are too large, we need to map them into a low-dimensional space. The dimensional reduction is implemented as a two-phase process. First, the semantic structure of the text is used to filter-out phrases that are less probable to qualify for a keyphrase. Thereafter, Singular Value Decomposition (SVD) is applied to reduce the dimension further by mapping the set of retained candidate phrases into a lower-dimensional space. Further details about these steps are presented in the following sub-sections.

Dimensionality Reduction using the Semantic Structure of the Text

We can reduce the dimension of document vectors by reducing the size of the candidate list. In this phase, we exploit the semantic structure of texts to remove a candidate phrase if it is not a part of either *subject* or *object* constituent of its source sentence. The subject and object of a sentence is determined from its dependency tree generated by the Stanford parser. To identify the subject of a sentence, we traverse the dependency tree of the sentence to identify a word w to which a verb

[3] A list of 500 stop-words appears at `http://www.abulaish.com/stopwords.txt`.

Algorithm 1 candidatePhraseExtraction(F)

```
1:  L_Phrase ← φ
2:  for each T ∈ F do
3:     for each internal NP node λ ∈ T do
4:        if all child nodes of λ are leaf node then
5:           p ← Null
6:           for each node ξ ∈ child[λ] do
7:              p ← p + word(ξ)
8:           end for
9:           p ← clean(p)
10:          if length(p) ≥ 2 then
11:             for each 1-, 2-, and 3-gram α of p do
12:                α ← clean(α)
13:                if α ≠ Null AND valid(α) = True then
14:                   L_Phrase ← L_Phrase ∪ α
15:                end if
16:             end for
17:          else if p ≠ Null AND valid(p) = True then
18:             L_Phrase ← L_Phrase ∪ {p}
19:          end if
20:       end if
21:    end for
22: end for
23: Return L_Phrase
```

v is related through the dependency relation **subj** . Then, we obtain the boundary of the subject by searching a word at lowest position (say, i) and another word at highest position (say, j) related to w through some dependency relations. If there is no word at a position lower than the position of w which is related with w, then i represents the position of w. Similarly, if there is no word at a position greater than the position of w which is related with w, then j represents the position of w. Thereafter, the subject is obtained by clubbing the words from position i to j together. Similarly, the object of a sentence is determined by using the dependency relation **obj** .

The subject and object of the sample sentences given in Table 1 is shown in Table 3. Once, the subject and object of a sentence is identified, retention of a candidate phrase extracted from that sentence is determined on the basis of its containment to either subject or object constituents of the sentence. That is, if a phrase is a part of either the subject or object of the sentence, it is retained, otherwise it is

Sent. no.	Candidate phrase
S1	activation, CD28, expression, expression and NF-kappa, gene, gene expression, IL-2, IL-2 gene, IL-2 gene expression, NF-kappa, NF-kappa B activation, oxygen, oxygen production, production, reactive, reactive oxygen, reactive oxygen production
S2	Activation, CD28, CD28 surface, CD28 surface receptor, cell, cell activation, cell proliferation, costimulatory, costimulatory signal, enhanced, enhanced production, IL-2, interleukin-2, major, major costimulatory, major costimulatory signal, production, proliferation, receptor, signal, surface, surface receptor
S3	activation, activity, cascade, CD28, CD28 signaling, CD28 signaling cascade, Delineation, kinase, kinase activity, phospholipase, phospholipase A2, protein, protein tyrosine, protein tyrosine kinase, signaling, signaling cascade, tyrosine, tyrosine kinase, tyrosine kinase activity
S4	activation, data, expression, formation, IL-2, IL-2 expression, lipoxygenase, lipoxygenase metabolites, metabolites, NF-kappa, NF-kappa B activation, ROI, ROI formation
S5	CD28, CD28 costimulatory, costimulatory, development, findings, immunosuppressants, strategies, therapeutic, therapeutic strategies

Table 2. Candidate phrases extracted from the sample sentences of Table 1.

Sent. no.	Subject(s)	Object(s)
S1	“IL-2 gene expression and NF-kappa B activation through CD28”	“reactive oxygen production”
S2	“Activation of the CD28 surface receptor”	“a major costimulatory signal”
S3	“Delineation of the CD28 signaling cascade”	“protein tyrosine kinase activity, followed”
S4	“Our data”, “lipoxygenase metabolites”	“ROI formation which”, “IL-2 expression via NF-kappa B activation”
S5	“These findings”	“the CD28 costimulatory”

Table 3. *Subject* and *object* of the sample sentences of Table 1.

Sent. no.	Candidate phrase
S1	activation, CD28, expression, expression and NF-kappa, gene, gene expression, IL-2, IL-2 gene, IL-2 gene expression, NF-kappa, NF-kappa B activation, oxygen, oxygen production, production, reactive, reactive oxygen, reactive oxygen production
S2	Activation, CD28, CD28 surface, CD28 surface receptor, costimulatory, costimulatory signal, major, major costimulatory, major costimulatory signal, receptor, signal, surface, surface receptor
S3	activity, cascade, CD28, CD28 signaling, CD28 signaling cascade, Delineation, kinase, kinase activity, protein, protein tyrosine, protein tyrosine kinase, signaling, signaling cascade, tyrosine, tyrosine kinase, tyrosine kinase activity
S4	activation, data, expression, formation, IL-2, IL-2 expression, lipoxygenase, lipoxygenase metabolites, metabolites, NF-kappa, NF-kappa B activation, ROI, ROI formation
S5	CD28, CD28 costimulatory, costimulatory, findings

Table 4. Retained candidate phrases after applying semantic structure based filtering.

Ph. ID	Candidate phrase	Ph. ID	Candidate phrase	Ph. ID	Candidate phrase
P1	NF-kappa	P19	CD28 surface	P37	protein tyrosine
P2	activation	P20	CD28 surface receptor	P38	protein tyrosine kinase
P3	expression	P21	costimulatory signal	P39	signaling
P4	gene expression	P22	major	P40	signaling cascade
P5	CD28	P23	major costimulatory	P41	tyrosine
P6	IL-2	P24	major costimulatory signal	P42	tyrosine kinase
P7	NF-kappa B activation	P25	receptor	P43	tyrosine kinase activity
P8	costimulatory	P26	signal	P44	data
P9	expression and NF-kappa	P27	surface	P45	formation
P10	gene	P28	surface receptor	P46	IL-2 expression
P11	IL-2 gene	P29	activity	P47	lipoxygenase
P12	IL-2 gene expression	P30	cascade	P48	lipoxygenase metabolites
P13	oxygen	P31	CD28 signaling	P49	metabolites
P14	oxygen production	P32	CD28 signaling cascade	P50	ROI
P15	production	P33	Delineation	P51	ROI formation
P16	reactive	P34	kinase	P52	CD28 costimulatory
P17	reactive oxygen	P35	kinase activity	P53	findings
P18	reactive oxygen production	P36	protein		

Table 5. Unique candidate phrases along with their phrase IDs.

deleted from the list of phrases. For example, the candidate phrases *development*, *immunosuppressants*, *strategies*, *therapeutic*, and *therapeutic strategies* extracted from the sample sentence S5 are deleted from the list of candidate phrases as they are not present in either subject or object of the sentence S5. In this way, a number of unwanted phrases is filtered out, which results in a reduced dimension of the phrase-sentence matrix needed for applying SVD to map the phrase set into a low-dimensional space. Further details about matrix construction and SVD appear in the following subsection. A list of retained candidate phrases after applying semantic structure based filtering on the example sentences of Table 1 is given in Table 4. A list of unique candidate phrases along with their unique id numbers is also shown in Table 5, which is needed to exemplify the SVD process.

Dimensionality Reduction using SVD

Once the list of candidate phrases from each sentence of a document is compiled, the document is converted into a *phrase-sentence* matrix A. In this matrix, a row represents a candidate phrase and the number of rows is the same as the length of the candidate phrase list. Similarly, a column of matrix A represents a sentence and the number of columns is the same as the number of record-size chunks present in the text document. The (i, j)th element of matrix A, a_{ij}, is determined as the

weight of phrase p_i in the jth sentence. The weight of a phrase p_i in the jth sentence, $\omega(p_{i,j})$, is calculated using equations (1) and (2), where $\text{tf}(p_{i,j})$ is the number of times p_i occurs in the jth chunk. $|D|$ is the total number of record-size chunks in the document, and $|\{d_j : p_i \in d_j\}|$ is the number of chunks containing p_i. The matrix A is normalized in such a way that the length of the sentence vector becomes 1. Figure 2 (a) presents the phrase-sentence matrix representation of the example sentences given in Table 1. The order of this matrix is 53×5 as there are total 5 record-size chunks (see Table 1) and total 53 candidate phrases are extracted from them (see Table 5).

$$\omega(p_{i,j}) = \text{tf}(p_{i,j}) \times \text{idf}(p_i), \tag{1}$$

$$\text{idf}(p_i) = \log \frac{|D|}{|\{d_j : p_i \in d_j\}|} + 1. \tag{2}$$

Since the phrase-sentence matrix is generally a sparse matrix and the dimension of the phrase as well as sentence vectors are large (a larger document may results in a large set of candidate phrases), we apply Singular Value Decomposition (SVD) [9] to map the feature set into a low-dimensional space. This increases the efficiency of the proposed system both in terms of memory and computing time requirements. For a given $m \times n$ matrix with $m \geq n$, the SVD decomposes it into an $m \times n$ orthogonal matrix U, an $n \times n$ diagonal matrix S, and an $n \times n$ orthogonal matrix V such that $A = USV^T$. In this decomposition, U represents the phrase matrix and V represents the sentence matrix. Each row of matrix V represents a sentence vector whose dimension is reduced from m to n in the new feature space. We can also reduce the dimension of sentence vectors by taking only first r columns of matrix V, where r is the rank of A. Figures 2 (b)–(d) show these matrices generated by applying SVD on the phrase-sentence matrix A of Figure 2 (a).

3.4 Phrase Grouping using MCL

Once we get the sentence representation in reduced feature space, the Markov clustering (MCL) [8] is applied on it to group the related sentences together. For clustering, we have used matrix V as input in which each row corresponds to a vector representation of the corresponding sentence. The MCL is an iterative method that interleaves matrix expansion and inflation steps. Matrix expansion corresponds to taking successive powers of the transition matrix, while matrix inflation makes the higher probability transition and reduces the lower probability transition. MCL does not require to set the parameter k, the number of output clusters, rather it tunes only the inflation parameter $r > 1$. A small value of r

A=

0.55	0.00	0.00	0.40	0.00
0.21	0.16	0.00	0.30	0.00
0.27	0.00	0.00	0.20	0.00
0.39	0.00	0.00	0.00	0.00
0.08	0.12	0.10	0.00	0.24
0.14	0.00	0.00	0.20	0.00
0.14	0.00	0.00	0.20	0.00
0.00	0.21	0.00	0.00	0.43
0.20	0.00	0.00	0.00	0.00
0.20	0.00	0.00	0.00	0.00
0.20	0.00	0.00	0.00	0.00
0.20	0.00	0.00	0.00	0.00
0.20	0.00	0.00	0.00	0.00
0.20	0.00	0.00	0.00	0.00
0.20	0.00	0.00	0.00	0.00
0.20	0.00	0.00	0.00	0.00
0.20	0.00	0.00	0.00	0.00
0.20	0.00	0.00	0.00	0.00
0.00	0.30	0.00	0.00	0.00
0.00	0.30	0.00	0.00	0.00
0.00	0.30	0.00	0.00	0.00
0.00	0.30	0.00	0.00	0.00
0.00	0.30	0.00	0.00	0.00
0.00	0.30	0.00	0.00	0.00
0.00	0.30	0.00	0.00	0.00
0.00	0.30	0.00	0.00	0.00
0.00	0.30	0.00	0.00	0.00
0.00	0.30	0.00	0.00	0.00
0.00	0.00	0.26	0.00	0.00
0.00	0.00	0.26	0.00	0.00
0.00	0.00	0.26	0.00	0.00
0.00	0.00	0.26	0.00	0.00
0.00	0.00	0.26	0.00	0.00
0.00	0.00	0.26	0.00	0.00
0.00	0.00	0.26	0.00	0.00
0.00	0.00	0.26	0.00	0.00
0.00	0.00	0.26	0.00	0.00
0.00	0.00	0.26	0.00	0.00
0.00	0.00	0.26	0.00	0.00
0.00	0.00	0.26	0.00	0.00
0.00	0.00	0.26	0.00	0.00
0.00	0.00	0.26	0.00	0.00
0.00	0.00	0.26	0.00	0.00
0.00	0.00	0.00	0.28	0.00
0.00	0.00	0.00	0.28	0.00
0.00	0.00	0.00	0.28	0.00
0.00	0.00	0.00	0.28	0.00
0.00	0.00	0.00	0.28	0.00
0.00	0.00	0.00	0.28	0.00
0.00	0.00	0.00	0.28	0.00
0.00	0.00	0.00	0.28	0.00
0.00	0.00	0.00	0.00	0.61
0.00	0.00	0.00	0.00	0.61

(a) Matrix A

U=

0.55	−0.12	−0.02	−0.05	0.14
0.32	0.03	0.03	0.10	−0.08
0.28	−0.06	−0.01	−0.03	0.07
0.23	−0.05	−0.01	−0.03	0.35
0.08	0.25	−0.05	−0.09	0.06
0.20	−0.04	0.00	−0.02	−0.05
0.20	−0.04	0.00	−0.02	−0.05
0.07	0.42	0.09	−0.16	−0.02
0.11	−0.02	0.00	−0.02	0.18
0.11	−0.02	0.00	−0.02	0.18
0.11	−0.02	0.00	−0.02	0.18
0.11	−0.02	0.00	−0.02	0.18
0.11	−0.02	0.00	−0.02	0.18
0.11	−0.02	0.00	−0.02	0.18
0.11	−0.02	0.00	−0.02	0.18
0.11	−0.02	0.00	−0.02	0.18
0.11	−0.02	0.00	−0.02	0.18
0.11	−0.02	0.00	−0.02	0.18
0.05	0.19	0.06	0.23	0.01
0.05	0.19	0.06	0.23	0.01
0.05	0.19	0.06	0.23	0.01
0.05	0.19	0.06	0.23	0.01
0.05	0.19	0.06	0.23	0.01
0.05	0.19	0.06	0.23	0.01
0.05	0.19	0.06	0.23	0.01
0.05	0.19	0.06	0.23	0.01
0.05	0.19	0.06	0.23	0.01
0.05	0.19	0.06	0.23	0.01
0.01	0.05	−0.25	0.02	0.00
0.01	0.05	−0.25	0.02	0.00
0.01	0.05	−0.25	0.02	0.00
0.01	0.05	−0.25	0.02	0.00
0.01	0.05	−0.25	0.02	0.00
0.01	0.05	−0.25	0.02	0.00
0.01	0.05	−0.25	0.02	0.00
0.01	0.05	−0.25	0.02	0.00
0.01	0.05	−0.25	0.02	0.00
0.01	0.05	−0.25	0.02	0.00
0.01	0.05	−0.25	0.02	0.00
0.01	0.05	−0.25	0.02	0.00
0.01	0.05	−0.25	0.02	0.00
0.01	0.05	−0.25	0.02	0.00
0.01	0.05	−0.25	0.02	0.00
0.17	−0.04	0.00	−0.01	−0.26
0.17	−0.04	0.00	−0.01	−0.26
0.17	−0.04	0.00	−0.01	−0.26
0.17	−0.04	0.00	−0.01	−0.26
0.17	−0.04	0.00	−0.01	−0.26
0.17	−0.04	0.00	−0.01	−0.26
0.17	−0.04	0.00	−0.01	−0.26
0.17	−0.04	0.00	−0.01	−0.26
0.05	0.41	0.06	−0.46	−0.03
0.05	0.41	0.06	−0.46	−0.03

(b) Matrix U

S=

1.18	0.00	0.00	0.00	0.00
0.00	1.06	0.00	0.00	0.00
0.00	0.00	1.00	0.00	0.00
0.00	0.00	0.00	0.94	0.00
0.00	0.00	0.00	0.00	0.78

(c) Matrix S

V=

0.69	−0.12	−0.02	−0.08	0.71
0.18	0.65	0.20	0.71	0.02
0.03	0.21	−0.97	0.08	−0.01
0.69	−0.15	−0.01	−0.02	−0.71
0.09	0.70	0.10	−0.70	−0.04

(d) Matrix V

Figure 2. SVD example (a) phrase-sentence matrix A, (b) decomposed matrix U, (c) decomposed matrix S, and (d) decomposed matrix V.

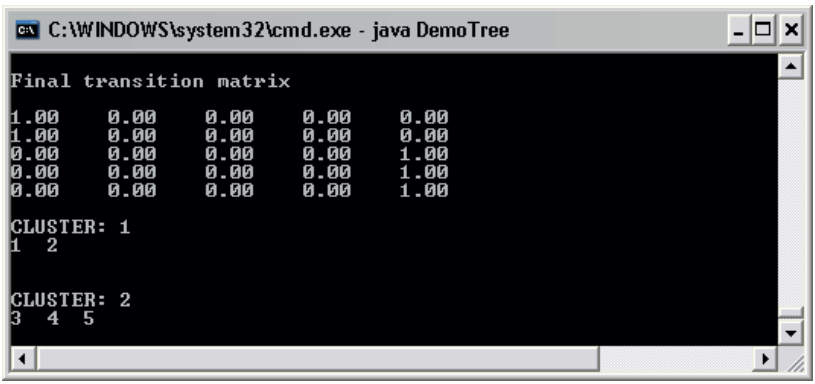


Figure 3. Clustering results obtained after executing the MCL algorithm on matrix V of Figure 2(d), which represents the sample sentences of Table 1 in reduced feature space.

results in larger clusters, whereas a high value of r is required to generate smaller clusters. Figure 3 shows a snapshot of the clustering result obtained after executing the MCL algorithm, with inflation parameter $r = 20$, on matrix V of Figure 2 (d) which represents the sentences of Table 1 in reduced feature space.

3.5 Keyphrase Identification using LDA

In this section, we discuss the Latent Dirichlet Allocation (LDA) execution process, which is used to identify feasible keyphrases. LDA is a generative probabilistic model in which documents are represented as random mixtures over latent topics characterized by a distribution over words [6]. For LDA execution, we create a data file using the clusters obtained in the previous section. In this file, the first line contains an integer value k representing the number of clusters (number of documents for LDA). Following this, there are k paragraphs, one for each cluster, containing the list of phrases obtained from the sentences belonging to the corresponding cluster. Figure 4 shows the LDA input data file generated for the sample sentences given in Table 1 that are grouped into two clusters as shown in Figure 3.

We have used `JGibbLDA`[4] to execute LDA to get the matrices Θ and Φ. We have set the Dirichlet hyper parameters α and β as 0.1 and 0.5, respectively, at the time of LDA execution. The Φ matrix contains the phrase-topic distributions, i.e., $p(\text{phrase}_p \mid \text{topic}_t)$. Each row in this matrix is a topic and each column is a candidate phrase in the document. The Θ matrix contains the topic-cluster distributions,

[4] `http://jgibblda.sourceforge.net/`

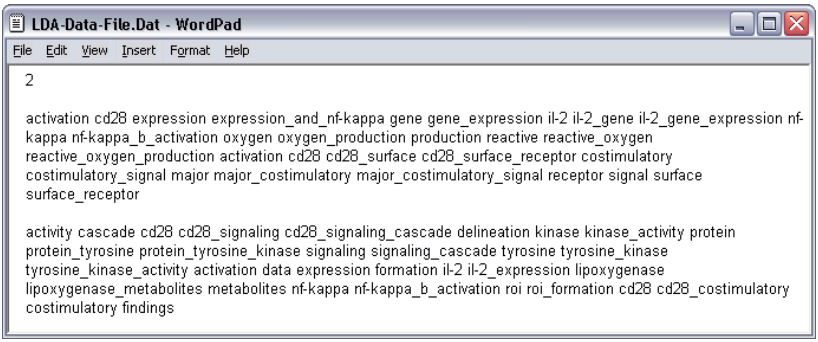


Figure 4. Input data file for LDA created using the bag-of-words representations of the sample sentences of Table 1.

i.e., $p(\text{topic}_t \mid \text{cluster}_c)$. Each row in this matrix is a cluster (document) and each column is a topic.

In line with [20], we use the matrices Φ and Θ to assign a ranking score to each phrase using equations (3) and (4) in which $|s[l]|$ is the size (number of phrases) of the lth cluster, n is the number of topics (we have taken $n = 100$), and k is the number of clusters that is treated as number of documents in our case. After calculating the score of each phrase, we arrange them in non-increasing order of their scores and select top-p phrases as keyphrases. Table 6 shows top-16 keyphrases along with their ranking scores identified from the example sentences of Table 1.

$$\text{score}(p_i) = \max_{j=1,\ldots,n} \{\Phi_{j,i} \times \omega_j\}, \tag{3}$$

$$\omega_j = \sum_{l=1}^{k} \Theta_{l,j} \times |s[l]|. \tag{4}$$

4 Experimental Setup and Performance Evaluation

To evaluate the accuracy of the proposed system, we carried out experiments with text documents from two different domains and compared our system with the state-of-the-art systems KEA and KEA++ in which KEA++ is a successor of KEA and provides better performance than KEA through using domain knowledge. In both experiments, we found that the proposed system has better performance than KEA and KEA++. We have used standard information retrieval performance evaluation metrics – *precision*, *recall*, and *f-score* (a.k.a. *f-measure* or f_1-*measure* in which

Keyphrase	Score	Keyphrase	Score
CD28	0.4559	protein tyrosine kinase	0.2735
expression	0.2735	IL-2 gene expression	0.2735
NF-kappa	0.2735	ROI formation	0.2735
NF-kappa B activation	0.2735	tyrosine	0.1559
cascade	0.2735	oxygen production	0.1559
CD28 signaling cascade	0.2735	CD28 surface receptor	0.1550
kinase activity	0.2735	activation	0.1248
protein	0.2735	CD28 signaling	0.1248

Table 6. Top-16 keyphrases along with their scores extracted from the sample sentences of Table 1.

precision and *recall* are given equal weightage) to compare the results. The precision value of the system reflects its capability to identify correct keyphrases, whereas the recall value reflects the capability of the system to locate all instances of keyphrases in a document, and f-score is defined as the harmonic mean of the duo. The formulations of these metrics are presented in equations (5), (6) and (7), respectively. Since ranking of keyphrases is an important factor in keyphrase extraction, we have also measured the accuracy of our ranking strategy for the keyphrases in terms of MRR (Mean Reciprocal Rank). The MRR value is calculated using equation (8), in which K is the set of actual keyphrases and rank_i is the rank of the ith keyphrase in the ranked list of extracted keyphrases.

$$\text{precision}(\pi) = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{5}$$

$$\text{recall}(\rho) = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{6}$$

$$f\text{-score}(F_1) = 2 \times \frac{\pi \times \rho}{\pi + \rho}, \tag{7}$$

$$\text{MRR} = \frac{1}{|K|} \times \sum_{i=1}^{|K|} \frac{1}{\text{rank}_i}. \tag{8}$$

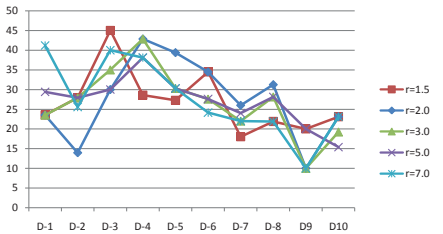
4.1 Experimental Results on GENIA Documents

The performance of the proposed method is evaluated on a set of text documents collected from the GENIA corpus [13], in which biological concepts are manually tagged using GENIA ontology concepts by the domain experts. For evaluation purpose, we have assumed that annotated texts appearing in GENIA documents represent important concepts (biological entities) and thus considered as keyphrases. Out of 2000 text documents appearing as a part of the GENIA corpus, we have randomly selected 10 documents, and from each document we have extracted and stored the annotated texts into a separate file before removing the annotations from it to facilitate the automatic calculation of performance evaluation metrics. After removing annotation from sample documents, we have executed our proposed system to identify feasible keyphrases from them. Since the average number of tagged texts (keyphrases) in sample documents is 13% of the document size, where document size is defined as the number of words in it, we have considered 13% top-ranked phrases as keyphrases and calculated the values of the performance evaluation metrics using them. From the set of extracted keyphrases, corresponding to each document, we calculate the true positives TP (number of correct keyphrases the system identifies as correct), the false positives FP (number of incorrect keyphrases the system falsely identifies as correct), and the false negatives FN (number of correct keyphrases the system fails to identify as correct). These parameters are used to calculate the value of *precision*, *recall*, and f*-score* using equations (5), (6) and (7), respectively. The *macro-average* values for all sample documents are computed by adding up all the individual TP, FP and FN values and then calculating the *precision*, *recall* and f*-score* values from them.

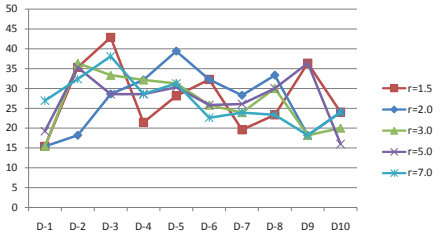
For calculating TP, FP and FN values, an evaluation program was written in Java that looks for the extracted keyphrases as well as the gold standard keyphrases from the test documents. This module stems the keyphrases before comparing them using the Porter stemmer to unify the differently inflected terms with a common stem. Table 7 summarizes the performance evaluation results of our proposed system for different values of r, where r is the inflation parameter used in the Markov clustering (MCL) process. Table 7 also presents the MRR values that present the ranking accuracy of the system. Figure 5 presents a visualization of the precision, recall and f-score for each test document. We have also compared the performance of our proposed system with `KEA` for cut-off values 3 and 5, i.e., considering top-3 and top-5 phrases from the ranked list as keyphrases. Figure 6 provides a comparative view of the performance of the proposed system with `KEA` in terms of precision, recall and f-score values. Although, the performance of the proposed system is better than `KEA` in all aspects, it needs further improvement, specifically in terms of recall values. On analysis it was found that most

	r = **2.0**				r = **3.0**				r = **5.0**				r = **7.0**			
Doc No.	π	ρ	F_1	MRR	π	ρ	F_1	MRR	π	ρ	F_1	MRR	π	ρ	F_1	MRR
1	23.53	15.38	18.6	0.10	23.53	15.38	18.6	0.08	29.41	19.23	23.26	0.09	41.18	26.92	32.56	0.09
2	13.95	18.18	15.79	0.06	27.91	36.36	31.58	0.08	27.91	35.29	31.17	0.08	25.58	32.35	28.57	0.07
3	30.00	28.57	29.27	0.10	35.00	33.33	34.15	0.11	30.00	28.57	29.27	0.21	40.00	38.10	39.02	0.10
4	42.86	32.14	36.73	0.16	42.86	32.14	36.73	0.21	38.10	28.57	32.65	0.17	38.10	28.57	32.65	0.13
5	39.39	39.39	39.39	0.12	30.30	31.25	30.77	0.09	30.30	30.30	30.30	0.10	30.30	31.25	30.77	0.10
6	34.48	32.26	33.33	0.09	27.59	25.81	26.67	0.10	27.59	25.81	26.67	0.17	24.14	22.58	23.33	0.13
7	26.00	28.26	27.08	0.13	22.00	23.91	22.92	0.05	24.00	26.09	25.00	0.05	22.00	23.91	22.92	0.07
8	31.25	33.33	32.26	0.10	28.13	30.00	29.03	0.10	28.13	30.00	29.03	0.11	21.88	23.33	22.58	0.08
9	10.00	18.18	12.90	0.05	10.00	18.18	12.90	0.09	20.00	36.36	25.81	0.16	10.00	18.18	12.90	0.14
10	23.08	24.00	23.53	0.15	19.23	20.00	19.61	0.06	15.38	16.00	15.69	0.09	23.08	24.00	23.53	0.08
Average	27.45	26.97	26.89	0.11	26.65	26.64	26.30	0.10	27.08	27.62	26.88	0.12	27.62	26.92	26.88	0.10

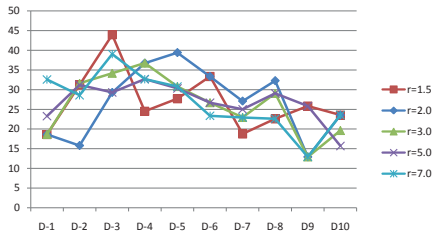
Table 7. Performance evaluation results of the proposed system over documents from GENIA corpus.



(a) *precision* plot

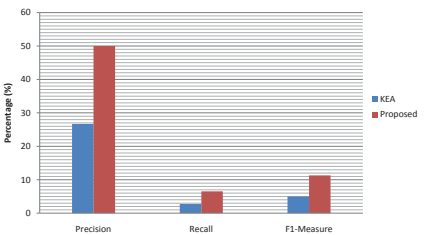


(b) *recall* plot

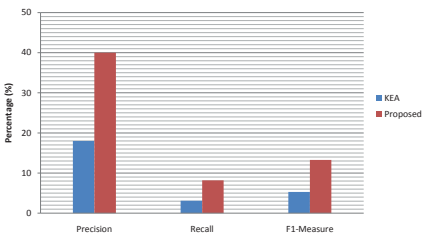


(c) *f-score* plot

Figure 5. Plotting of evaluation results of the proposed system on GENIA documents for different values of r.



(a) Comparison results for top-3 keyphrases.



(b) Comparison results for top-5 keyphrases.

Figure 6. Comparative performance of the proposed system and KEA over GENIA documents.

of the misses occur when a word is misclassified by the parser during document pre-processing and parsing phase.

4.2 Experimental Results on Agricultural Documents

In order to judge the scalability of the proposed system, in this section, we present experimental results on documents from agriculture domain. We also present a comparative evaluation of the proposed system with widely-used state-of-the-art techniques KEA and KEA++ (advanced version of the former one). KEA simply employs supervised learning algorithm to impart domain knowledge in it, whereas to capture domain knowledge more rigorously, KEA++ uses external thesaurus links which makes it constrained to the domain of the thesaurus used. We have compared the proposed approach with the existing systems of KEA[5] and KEA++[6], both of which are trained on a set of 25 agricultural documents, using *agrovoc* thesaurus for thesaurus links in KEA++, and tested on 5 other documents from the same domain. The individual results of each of the 5 test documents are presented in Table 8 for top-3, -5, -7 and -9 most promising keyphrases separately, and finally their macro-average results are also presented to compare the overall summarized result. Figure 7 presents a visualization of the comparative performance of the proposed system, KEA and KEA++, over these documents.

It can be observed in Figure 7 that in all the four cases of top three to nine keyphrases, our system produces the best values followed by KEA++ and the worst results are shown by KEA. It establishes the soundness of our system, which even without employing any domain knowledge as opposed to KEA++, outperforms both of them.

5 Conclusion and Future Work

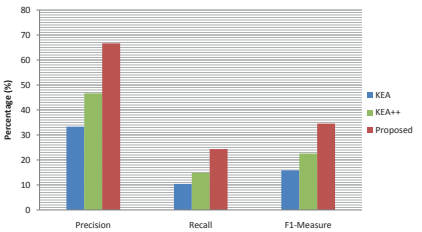
In this paper, we have proposed the design of a deep text mining system to identify keyphrases embedded within text documents. The proposed system applies a rule-based heuristic approach to identify potential candidate phrases after analyzing phrase structure trees generated by the Stanford parser. The extracted candidate phrases are passed through various phases including stop-words removal and phrase cleaning to eliminate as many noisy phrases as possible at the early stage of the processing. Moreover, instead of applying the k-means algorithm to group related sentences of a document together, we have applied Markov clustering which does not require the number of clusters as an input parameter from the user. In contrast to supervised learning approaches, the novelty of the proposed system lies in

[5] *Version 3.0* is a Java implementation of the original KEA by Frank et al. in 1999.

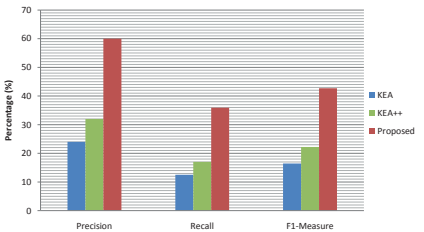
[6] *Version 5.0* is the Java implementation of KEA++ by Medeleyan and Witten in 2006.

Cut-off Point	KEA ($\pi/\rho/F_1$)	KEA++ ($\pi/\rho/F_1$)	Proposed ($\pi/\rho/F_1$)
Document 1			
Top-3	33.33/11.11/16.66	33.33/11.11/16.66	66.67/22.22/33.33
Top-5	20.00/11.11/14.28	20.00/11.11/14.28	60.00/33.33/42.86
Top-7	14.28/11.11/12.5	14.28/11.11/12.5	57.14/40.00/47.06
Top-9	11.11/11.11/11.11	11.11/11.11/11.11	44.44/40.00/42.11
Document 2			
Top-3	33.33/10.00/15.38	33.00/10.00/15.38	66.67/20.00/30.77
Top-5	20.00/10.00/13.33	40.00/20.00/26.67	60.00/30.00/40.00
Top-7	14.28/10.00/11.76	28.57/20.00/23.53	57.14/40.00/47.06
Top-9	11.11/10.00/10.53	22.22/20.00/21.06	44.44/40.00/42.11
Document 3			
Top-3	66.67/40.00/50.00	100.00/75.00/85.71	33.33/25.00/28.57
Top-5	40.00/40.00/40.00	60.00/75.00/66.67	40.00/50.00/44.44
Top-7	28.57/40.00/33.33	42.86/75.00/54.55	28.57/50.00/36.36
Top-9	22.22/40.00/28.57	33.33/75.00/46.15	22.22/50.00/30.77
Document 4			
Top-3	33.33/14.29/20.00	33.33/14.29/20.00	100.00/42.86/60.00
Top-5	40.00/28.57/33.33	20.00/14.29/16.67	60.00/42.86/50.00
Top-7	28.57/28.57/28.57	14.29/14.29/14.29	42.86/42.86/42.86
Top-9	22.22/28.57/25.00	11.11/14.29/12.5	33.33/42.86/37.50
Document 5			
Top-3	00.00/00.00/00.00	33.33/05.88/10.00	66.67/11.76/20.00
Top-5	00.00/00.00/00.00	20.00/05.88/09.09	80.00/23.53/36.36
Top-7	00.00/00.00/00.00	42.86/17.65/25.00	71.43/29.41/41.67
Top-9	0.00/0.00/0.00	55.56/29.41/38.46	66.67/35.29/46.15
Macro-Average			
Top-3	33.33/10.42/15.87	46.67/14.89/22.58	66.67/24.37/**34.53**
Top-5	24.00/12.50/16.43	32.00/17.02/22.22	60.00/35.94/**42.73**
Top-7	17.14/12.50/14.45	28.57/21.28/24.39	51.43/40.45/**43.00**
Top-9	13.33/12.50/12.90	26.67/25.53/26.09	42.22/41.63/**39.73**

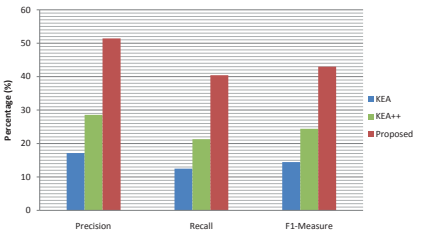
Table 8. Comparison results of the proposed system with KEA and KEA++ on agricultural documents.



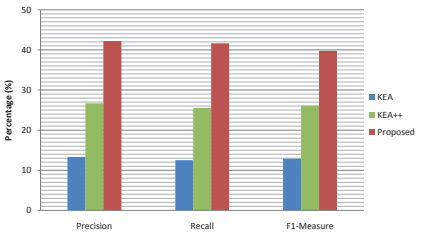
(a) Comparison results for top-3 keyphrases.



(b) Comparison results for top-5 keyphrases.



(c) Comparison results for top-7 keyphrases.



(d) Comparison results for top-9 keyphrases.

Figure 7. Comparative performance of the proposed system, KEA and KEA++ over agricultural documents.

its ability to identify keyphrases without requiring a pre-annotated set of training documents. The increasing performance of our system can be attributed to the proposed candidate phrase extraction and filtering mechanisms, in addition to using Markov clustering for grouping related sentences together. Presently, we are testing our system on documents from other general-purpose domains like news documents, Wikipedia documents, etc. and exploring some more NLP techniques to increase the precision and recall values of the system.

Bibliography

[1] M. Abulaish, Jahiruddin and L. Dey, A statistical approach for automatic keyphrase extraction, in: *Proceedings of the 5th Indian International Conference on Artificial Intelligence* (IICAI'11), Tumkur, India, to appear.

[2] J. Allen, *Natural Language Understanding*, 2nd ed., Pearson Education, Singapore, Indian branch, 2004.

[3] K. Barker and N. Cornacchia, Using noun phrase heads to extract document keyphrases, in: *Proceedings of the 13th Canadian Conference on Artificial Intelligence* (LNAI 1822), Montreal, Canada (2000), 40–52.

[4] G. Berend and R. Farkas, SZTERGAK: Feature engineering for keyphrase extraction, in: *Proceedings of the 5th International Workshop on Semantic Evaluation* (ACL'10), Uppsala, Sweden (2010), 186–189.

[5] A. Berger and V. Mittal, OCELOT: A system for summarizing web pages, in: *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Athens, Greece (2000), 144–151.

[6] D. M. Blei, Y. N. Andrew and I. J. Michael, Latent dirichlet allocation, *Journal of Machine Learning Research* **3** (2003), 993–1022.

[7] D. B. Bracewell, J. Yan and F. Ren, Single document keyword extraction for internet news articles, *International Journal of Innovative Computing, Information and Control* **4** (2010), 905–913.

[8] S. van Dongen, A cluster algorithm for graphs, University of Utrecht, 2000.

[9] G. E. Forsythe, M. A. Malcolm and C. B. Moler, *Computer Methods for Mathematical Computations*, Prentice Hall Professional Technical Reference, 1977.

[10] E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin and C. G. Nevill-Manning, Domain-specific keyphrase extraction, in: *Proceedings of the 16th International Joint Conference on Artificial Intelligence* (IJCAI'99), Stockholm, Sweden (1999), 668–673.

[11] C. Gutwin, G. W. Paynter, I. H. Witten, C. G. Nevill-Manning and E. Frank, Improving browsing in digital libraries with keyphrase indexes, *Journal of Decision Support Systems* **27** (1999), 81–104.

[12] K. M. Hammouda, D. N. Matute and M. S. Kamel, CorePhrase: Keyphrase extraction for document clustering, in: *Proceedings of the 4th International Conference on Machine Learning and Data Mining* (MLDM'05), Leipzig, Germany (2005), 265–274.

[13] J. D. Kim, T. Ohta, Y. Teteisi and J. Tsujii, GENIA corpus – a semantically annotated corpus for bio-textmining, *Bioinformatics* **19** (2003), 180–182.

[14] B. Krulwich and C. Burkey, Learning user information interests through the extraction of semantically significant phrases, in: *Proceedings of the 13th National Conference on Artificial Intelligence* (AAAI'96), Portland, Oregon, USA (1996), 110–112.

[15] O. Medelyan and I. H. Witten, Thesaurus based automatic keyphrase indexing, in: *Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries*, Chapel Hill, North Carolina, USA (2006), 296–297.

[16] O. Medelyan, I. H. Witten and D. Milne, Topic indexing with wikipedia, in: *Proceedings of the AAAI Workshop on Wikipedia and Artificial Intelligence* (WIKIAI'08), Chicago, USA (2008), 19–24.

[17] R. Mihalcea and P. Tarau, Textrank: Bringing order into texts, in: *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (EMNLP'04), Barcelona, Spain (2004), 401–411.

[18] C. Q. Nguyen and T. T. Phan, An ontology-based approach for key phrase extraction, in: *Proceedings of the ACL-IJCNLP'09*, Suntec, Singapore (2009), 181–184.

[19] T. D. Nguyen and M. Y. Kan, Keyphrase extraction in scientific publications, in: *Proceedings of the International Conference on Asian Digital Libraries* (ICADL'07), Hanoi, Vietnam (2007), 317–326.

[20] C. Pasquier, Task 5: Single document keyphrase extraction using sentence clustering and latent dirichlet allocation, in: *Proceedings of the 5th International Workshop on Semantic Evaluation* (ACL'10), Uppsala, Sweden (2010), 154–157.

[21] K. Sarkar, Automatic keyphrase extraction from medical documents, in: *Proceedings of the 3rd International Conference on Pattern Recognition and Machine Intelligence* (PReMI'09), LNCS 5909, IIT Delhi, India (2009), 273–278.

[22] T. Tomokiyo and M. Hurst, A language model approach to keyphrase extraction, in: *Proceedings of ACL Workshop on Multiword Expressions*, Sapporo, Japan (2003), 33–40.

[23] P. D. Turney, Learning to extract keyphrases from text, Technical Report ERB-1057, National Research Council, Institute for Information Technology, 1999.

[24] P. D. Turney, Learning algorithm for keyphrase extraction, *Journal of Information Retrieval* **2** (2000), 303–336.

[25] X. Wan and J. Xiao, Single document keyphrase extraction using neighborhood knowledge, in: *Proceedings of the 23rd National Conference on Artificial Intelligence*, Chicago, Illinois (2008), 855–860.

[26] W. Zhang, T. Yoshida, T. B. Ho and X. Tang, Augmented mutual information for multi-word extraction, *International Journal of Innovative Computing, Information and Control* **5** (2009), 543–554.

Received June 15, 2011.

Author information

Muhammad Abulaish, Center of Excellence in Information Assurance,
King Saud University, Riyadh, Kingdom of Saudi Arabia.
E-mail: mabulaish@ksu.edu.sa

Jahiruddin, Department of Computer Science, Jamia Millia Islamia, New Delhi, India.
E-mail: jahir.jmi@gmail.com

Lipika Dey, Innovation Labs, Tata Consultancy Services, New Delhi, India.
E-mail: lipika.dey@tcs.com