Concept Space Derivation and its Application in Query Categorization

Yashodhara Haribhakta
Research Scholar
Department of Computer Engineering and Information Technology, College of Engineering, Pune, India

Parag Kulkarni, PhD.
Adjunct Professor
Department of Computer Engineering and Information Technology, College of Engineering, Pune, India

ABSTRACT
Automatic text categorization (also known as text classification or topic spotting) is the activity of labeling natural language texts with thematic categories from a predefined set. For the purpose of classifying the text documents, there is a need for a set of features and a weighting model which gives relevance value to the features. Currently, bag of words(BOW) is found to be the most widely accepted text representation method . This representation has two major drawbacks. First, the amount of features is very large; second, there is no relatedness between the words. Topic Detection (TD) technique helps the BOW to handle the two drawbacks by detection features very relevant to the document in the space of concept. However, existing TD techniques were not designed for text categorization and often involve huge computational complexity and cost. This paper proposes a topic detection technique for relevant feature extraction. The TD technique extracts topics along with relevant features for each text document. It then finds relatedness between features for each topic. The features extracted for each topic are tightly related to the topic and accordingly the category label. The term frequency measure selects the appropriate features by finding frequency count for each extracted feature for each category label. Thus, the TD technique extracts the relevant features for the classifiers for classification. To evaluate the TD technique, a query categorization system is designed and proposed. The experiments were performed on three datasets (Reuters 21578, Ohsumed and 2G Scam) . The experimental results for TD technique show that the topics, along with the set of keywords, detected for documents are indeed relevant .Also, the query categorization system showed satisfactory performance in categorizing the queries using the TD technique.

General Terms
Text Mining, Text categorization, Query Categorization

Keywords
Concept Derivation, Topic detection, BOW representation, Performance Evaluation

1. INTRODUCTION
Today, if we look at the information on the web, specifically the text information, the text data is huge. Automatic categorization of text documents is the need for proper organization and efficient management of the text documents. Text categorization is a supervised learning approach which does automatic labeling of class labels or topics for new unlabelled documents based on the heuristics of maximum likelihood suggested by a training model of labeled documents (Yang and Liu, 1999).

Before the text documents are being processed by any machine learning algorithms, the text documents should be represented in some way essential for text categorization. Salton et al.(1975) has proposed vector space model(VSM) for text representation which proved to be very effective, after a lot of research, for text categorization. Using VSM, the text documents can be represented as vectors in feature space where features are extracted using a methodology of topic detection. Earlier, no attempt has been made for using the techniques in topic detection for feature extraction useful for text categorization. Usually, text data representation is done by performing two basic steps: feature extraction and feature selection using some weighting model. Feature extraction refers to identifying significant features which represents the text document and feature selection using some weighting model refers to assigning some appropriate weight values to the identified significant features of the text document.

2. Feature Representation
In most of the prior research for text categorization, the technique used for representation of the text data is bag-of-words(BOW). In BOW representation, the text documents are considered as a set of unordered words (Schutze et al. 1995, Xue and Zhou,2009) whereas the features are simple words. Depending on the presence or absence of a word in the document or the frequency of word in the document , weight values are assigned to each word of the document(Apte et al. 1994; Koller and Sahami 1997; Lewis and Ringuette 1994; Li and Jain 1998;Moulinier et al. 1996; Moulinier and Ganascia 1996; Schapire and Singer 2000; Schutze et al. 1995). To handle the issue of word co-occurrence and integrity of words , statistically derived word phrases (Cohen and Singer 1999; Mladeni´c 1998; Schapire et al. 1998;Caropreso et al 2001) , syntactical phrase derived from english language grammar(Fuhr et al. 1991; Lewis 1992; Tzeras and Hartmann 1993) are treated as features. But unfortunately, the performance improvement by these features was not very encouraging (Lewis 1992, Xue and Zhou, 2009). Thus, most of the research in text categorization is based on BOW representation which simply uses words as features.

The BOW provides a suitable way to turn text data into vectors but suffers from two major drawbacks. First , in BOW, the total number of words in entire corpus determines the document vectors. Unfortunately, the amount of words contained in the corpus is much more than a single text document thereby resulting in a very high vector space. High dimensionality of feature space is thus a major challenge for many learning algorithms as it leads to very high computational complexity. Second, the similarity between

different words is not feasible to be calculated as it is impossible to represent words as vectors in vector space . As a fact, it is not possible to handle the polysemy and synonymy by the BOW technique.

2.1 Existing Topic Detection Techniques

The prior work of topic detection is discussed as follows. Actually, there is very little work for feature extraction using topic detection methodology. A very common approach to topic detection focuses on modeling of the basic generative process of textual data. The basic idea used is to build a probabilistic model which details out how the different language units within the text are generated. Then such probabilistic learnt models are used for generating topic for the text. This basically involves inferring over words the probability distribution associated with each topic, and the probability distribution over the topics for every document . The earliest Probabilistic topic detection model [1997] called Latent Semantic Analysis (LSA) detects latent topics in the textual data and has been applied with remarkable success in a number of domains, but it had a list of deficits, mainly due to its poor statistical foundation. Since the PLSA model is based on the likelihood principle, it has a good statistical foundation, and therefore defines a more proper generative model of the textual data. Since the PLSA [18, 19, 20, 21, 2, 3, 24] model does not make any assumptions about the generation of the mixture weights, it is difficult to test the generalizability of the PLSA model to new documents. Adding a Dirichlet prior to PLSA resulted in a new generative model called Latent Dirichlet Allocation (LDA) in which the result is a smoothed topic distribution. All these prior work involves probabilistic approach for topic detection which requires a high computational cost, domain language and copyright which is not cost effective for most of the other related research.

2.2 Topic Detection (TD) Model

Existing topic detection techniques such as LSA,PLSA and LDA are designed for applications involving natural language processing, text retrieval but not specifically designed for text categorization. These techniques try to interpret the meaning of words in a complex information space. Consequently to implement these techniques, a huge computational cost is required. In this work, a non-probabilistic approach for topic detection is proposed. It is simple but more efficient for TD . It is based on the idea of word decomposition which works in two phases. The first phase deals with the construction of confident unigram , bigram and trigram words and the second phase deals with derivation of topics using the hypothesis of word decomposition.

The contribution of this paper are threefold. First, a new methodology to extract concept from the information of the text is proposed. The concepts are the topics defined for the text. The methodology is simple but more efficient and is designed specifically for application involving text categorization, query categorization , information retrieval, etc.,. Secondly, a new weighting model is proposed for finding the relatedness between the words and their concepts. The new method takes into consideration the occurrence frequency of each word in its related concepts. Finally, the proposed approach is evaluated on a query categorization system. The evaluation is performed on three different corpora.

Experimental performance and analysis shows that the proposed methodology of TD performs better and comparable to , but not worse , than the bag-of-words representation with different datasets. It requires much less computing time thereby reducing the computational cost drastically. Also, the query categorization system for local search performs better using the proposed TD technique.

The rest of the paper is organized as follows: Section 3 discusses about how to build the concept space. Section 4 discusses about the relatedness between the words and concept space in relation to documents . Section 5 proposes a query categorization system which uses the derived concept space . Section 6 reports the experimentation, results on three datasets and analysis of the results with discussion. Finally, section 7 concludes the paper.

3. Concept Space Building

What defines the concept in the building model of concept space is very important. For different applications, the requirements of concept are different. It would be impossible to define a universal concept set once and for all. Text Categorization is purely a statistical problem and it would be very difficult to identify the relatedness between the words and their concepts. Since the corpus is the only knowledge repository, initially without using any external knowledgebase the concept space is builded. So, defining concept set for each document and then the corpus would be a more preferable approach.

In prior research (Salton et al.,1975;Lan et al., 2009; Xue and Zhou, 2009), no attention has been given to the class labels and building of concept sets. The class labels are treated as meaningless symbols. In fact, the class labels and the words (either unigram, bigram, trigram) are often of great significance. Treating the class label as meaningless and not focusing on n-gram words, one will never be able to determine meaningful concepts and relatedness of n-gram words with the concepts. For example, consider the Reuters 21578(Lewis, 1995) top 10 categories {acq, corn, crude, earn, grain, interest, money-fx, ship, trade, wheat}. Each of these categories has a class label which has a defined meaning and it would be easy to predict the relatedness of words with the class labels. Given a word "money", it can be seen that the word is closer to the class label "money-fx" and less close to class labels "acq", "wheat" or "grain". Given a word "maize", it is more close to class label "corn" and maybe relative with class labels "grain" and "wheat" but has rare relatedness with class label "earn" or" acq". Also, the uni-gram words which are the components of the n-gram words which defines the concepts can be justified on similar grounds. This shows that it is feasible to interpret words (unigram, bigram or trigram) in the concept space derived from class label. Also, concepts derived from the document header and the document text would be meaningful in finding relatedness between the words in the concept space derived from document header and document text with the words in the concept space derived from class labels . Then how to derive concepts from the class labels, document header and the document text for a corpus ?

3.1 Derivation of Concept Space for Document text , Document header and Class Labels

This approach starts by treating class label of each document as first available concepts for the class. Usually the class label i.e., name of the document conveys what the document is about. It means, it conveys important information about the topic of the document, so the term features in the document name represents significant features which helps in concept space construction for class labels. Then, it applies a simple methodology for deriving concept space for document text

and document header. The concepts derived are the topics defined for the text. So, concepts will be referred as topics hence further. The topic derivation is based on the idea of word decomposition which works in two phases. The first phase, as discussed in Algorithm 1, deals with the construction of confident single-value(unigram) words and confident multi-value(bigram, trigram) words and the second phase, as discussed in Algorithm 2, deals with derivation of topic using the hypothesis of word decomposition.

Consider documents to be bag of words (ordering of words is maintained). Upper case letters are used to represent sets and lower case letters are used for elements of the set. D = {d_1, d_2,...,d_m} represent the document set of the input corpus. W={w_1,w_2,…….,w_n} represents the set of all the different term features in D. T= {t_1,t_2,……..,t_p} is the concept or topic space. C={c_1,c_2,…….,c_m} is the class label. Let tf(d_i,w) denote the frequency of term feature $w \in W$ in the document $d_i \in D$. F = {f_1, f_2, …., f_m} be the set of names of files such that f_i is the filename of document $d_i \in D$. Also, DH ={dh_1, dh_2, ……, dh_m} bet the set of document header of files $f_i \in F$.

Algorithm 1: Concept_Space_Derivation(CSD)

Input : The corpus D = {d_1, d_2,...,d_m}

Output : Unigram, bigram, trigram vectors of document text for each $d_i \in D$ and for the entire corpus

for each $d_i \in D$ do

/ initially obtain the term feature vector for each $d_i \in D$

$\overrightarrow{t}_{d_i}$= { w_1, w_2 , ……, w_m} where m is size of d_i

// Maintain two instances of the vector $\overrightarrow{t}_{d_i}$, such as $\overrightarrow{t\_first}_{d_i}$ and $\overrightarrow{t\_second}_{d_i}$

$\overrightarrow{t\_first}_{d_i}$= { $w_1(d_i)$, $w_2(d_i)$, ……., $w_m(d_i)$ }

= $\overrightarrow{t\_second}_{d_i}$

// Preprocess $\overrightarrow{t\_first}_{d_i}$, that is, perform removal of stop-words, punctuations,

// perform stemming . Now obtain the term frequency of each feature, w_j, in $d_i \in D$.

tf(w_j, d_i) = $\sum_{\forall w_j}$ tf(d_i, w_j)

//The preprocessed term vector is now denoted as

$\overrightarrow{tp\_f}_{di}$ = {tf(d_i,w_1), tf(d_i,w_2), ……., tf(d_i,w_m)}

// Sort the preprocessed term vector, $\overrightarrow{tp\_f}_{d_i}$, consisting of unigram words

// The sorted pre-processed term vector is,

$\overrightarrow{tp\_f}_{s\_di}$ = { $tf_{max}(d_i,w_1)$, ……., $tf_{min}(d_i,w_m)$ }

enddo

Top 10 term features from the sorted vector are finally selected as strong or significant term features for each d_i . The rationale behind discarding the term features below 10 is that those features were not helping with identifying appropriate classes but they contributed more in form of noise thereby degrading the overall performance.

The selected top 10 term features from each $d_i \in D$ are stored in a unigram word vector, called $\overrightarrow{U}$. The vector $\overrightarrow{U}$ is sorted and finally top 20 features are maintained.

$\overrightarrow{U}$ = $\cup_{i=1}^{D}$ $\overrightarrow{tp\_f}_{s\_di}$ // sorted and top 20 are maintained if they exist or all

// Now, obtain bigram, and trigram word vectors for each document

Consider the second instance, $\overrightarrow{t\_second}_{d_i}$, to obtain the trigram vector

// Let T_r denote the threshold on trigrams

for each $d_i \in D$ do

Let LU_i = $\overrightarrow{t\_second}_{d_i}$ // list of unigrams in d_i where ordering of words is maintained

Let P_i = empty & Q_i=empty

for j = 3 .. number of unigrams in LU_i do

Let tr_j = (LU_i[j-2], LU_i[j-1], LU_i[j]) be the j-th trigram over LU_i

Score(tr_j, d_i) = 1

P_i = P_i U tr_j

enddo

for each trigram tr_k in P_i

Score(tr_k, P_i) = $\sum_{\forall tr_k}$ Score(P_i, tr_k)

if (Score(tr_k) > Tr) then

Q_i = Q_i U tr_k

endif

enddo

// trigram for each document d_i

$\overrightarrow{tr}_{d_i}$= sort Q_i and store top 10 trigrams (if they exist) or all in the trigram vector

enddo

The Top 10 trigrams from the sorted list Q_i are finally selected as strong or significant trigrams for each d_i

// The trigram vector for D , called $\overrightarrow{TRI}$, is

$\overrightarrow{TRI}$ = $\cup_{i=1}^{D}$ $\overrightarrow{tr}_{di}$ // sorted and top 20 are maintained if they exist or all

Again, consider the second instance, $\overrightarrow{t\_second}_{d_i}$, to obtain the bigram vector

// Let T_b denote the threshold on bigrams

for each $d_i \in D$ do

Let LU_i = $\overrightarrow{t\_second}_{d_i}$ // list of unigrams in d_i where ordering of words is maintained

Let P_i = empty , Let Q_i=empty

for j = 2 .. number of unigrams in LU_i do

Let b_j = (LU_i[j-1], LU_i[j]) be the j-th bigram over LU_i

Score(b_j, d_i) = 1

P_i = P_i U b_j

enddo

for each bigram, b_z, in P_i

Score(b_z, P_i) = $\sum_{\forall b_z}$ Score(P_i, b_z)

if (($b_z \notin$ Powerset($\overrightarrow{tr}_{di}$)) and score(b_z >Tr)) then

Q_i = Q_i U b_j

endif

enddo

// bigram for each document d_i

$\overrightarrow{b}_{d_i}$= sort Q_i and store top 10 bigrams (if they exist) or all in the bigram vector

enddo

The Top 10 bigrams from the sorted list Q_i are finally selected

as strong or significant bigrams for each d_i.
The bigram vector for D , called $\overrightarrow{B}$, is
$\overrightarrow{B} = \cup_{i=1}^{p} \vec{b}_{di}$ // sorted and top 20 are maintained if they exist or all
Finally, k-gram vector for D, called $\overrightarrow{K}$, is obtained by the union of bigram and trigram vectors.
$\overrightarrow{K} = \overrightarrow{B} \cup \overrightarrow{TRI}$.

The CSD Algorithm 1, discusses the methodology of concept space construction for document text, document header and class label of each document. Usually, the first few sentences in any document tend to give information regarding the specific topic that is discussed in the document. It is very rare situation that the topic which is discussed through the document is not spoken of or named in the first few lines, so the first few lines in the document represent significant features which will help in topic derivation. So, consider the starting 20 words separately as an document header. Similarly to obtaining vectors for each $d_i \in D$, obtain unigram, bigram and trigram vectors for each $f_i \in F$ and each document header $dh_i \in f_i$. Thus, referring to algorithm 1, obtain the following vectors,

$\overrightarrow{DN}_{f_i}$ = contains unigrams of filename $f_i \in F$
$\overrightarrow{DN}_{b_i}$ = contains bigrams of filename $f_i \in F$
$\overrightarrow{DN}_{tr_i}$ = contains trigrams of filename $f_i \in F$
$\overrightarrow{DH}_{d_i}$ = contains unigrams of document header dh_i of filename $f_i \in F$
$\overrightarrow{DH}_{b_i}$= contains bigrams of document header dh_i of filename $f_i \in F$
$\overrightarrow{DH}_{tr_i}$ = contains trigrams of document header dh_i of filename $f_i \in F$

After obtaining the unigram, bigram and trigram word vectors for the collection D ,class labels and document headers , the second phase is of topic derivation. For topic derivation, maintain two vectors , $\overrightarrow{U}$ and $\overrightarrow{K}$.

Algorithm 2 : Topic Derivation (TDR)

Input : Unigram, bigram, trigram term vectors for each $d_i \in D$, for each $f_i \in F$ and for each $dh_i \in f_i$ and the corpus

Output : topic or concept for each $d_i \in D$ and set of confident features for the topic

// Compare unigram vectors $\overrightarrow{DN}_{f_i}$ with $\overrightarrow{U}$, $\overrightarrow{DH}_{d_i}$ with $\vec{U}$ and $\overrightarrow{DH}_{d_i}$ with $\overrightarrow{DN}_{f_i}$ for each $d_i \in D$ and store the matched result of all in $\overrightarrow{U}_{all}$
// Similarly, compare the bigram term vectors $\overrightarrow{DN}_{b_i}$ and $\overrightarrow{DH}_{b_i}$ with $\vec{b}_{di}$, and trigram term vectors $\overrightarrow{DN}_{tr_i}$ and $\overrightarrow{DH}_{tr_i}$ with $\vec{tr}_{di}$ for each $d_i \in D$ and store the matched result in $\overrightarrow{V}_{all}$

$\overrightarrow{DNU} = \overrightarrow{DN}_{f_i} \cap \vec{U}$
$\overrightarrow{DHU} = \overrightarrow{DH}_{d_i} \cap \vec{U}$
$\overrightarrow{DHN} = \overrightarrow{DH}_{d_i} \cap \overrightarrow{DN}_{f_i}$
$\overrightarrow{U}_{all} = \overrightarrow{DHU} \cup \overrightarrow{DHN} \cup \overrightarrow{DNU}$
$\overrightarrow{V}_{all} = (\overrightarrow{DN}_{b_i} \cap \overrightarrow{DH}_{b_i} \cap \vec{b}_{di} ,) \cup (\overrightarrow{DN}_{tr_i} \cap \overrightarrow{DH}_{tr_i} \cap \vec{tr}_{di})$
// Initialize an empty list, TopicDoc
for each $d_i \in D$ do
  TopicDoc = empty
  for each $v_j \in \overrightarrow{V}_{all}$ do
    if (PowerSet (v_j) ϵ $\vec{U}_{all}$) then
      TopicDoc = TopicDoc $\cup$ {v_i }
    // obtain the score of each feature, $td_i \in TopicDoc$
      score($td_i, TopicDoc$) = score(td_i) in $\overrightarrow{V}_{all}$
    endif
  enddo
if (TopicDoc$_{di}$ is not empty) then
    topic$_{di}$ = max(score($td_i, TopicDoc$))
else
  if (($\overrightarrow{DNU}$ is not empty) then
    topic$_{di}$ = max(tf($\overrightarrow{DNU}$, dnu_i))
  else
  if (compare ($\overrightarrow{U}_{all}$, $\vec{U}$) then
    topic$_{di}$ = matched unigram with highest term frequency
  endif
  endif
endif
Confident_Feature_Set$_{di}$ = top 5 terms from $\vec{U}$
enddo

The TDR algorithm describes the way in which $\overrightarrow{V}_{all}$ and $\overrightarrow{U}_{all}$ are compared to extract the confident terms which are saved as the potential candidates for topics (in TopicDoc). Now the highest score candidate from TopicDoc is chosen as topic of the document. If there are two candidates having the same highest score then any one of them is selected randomly, or otherwise both of them are selected as topic of the document. The algorithm handles the situation of not finding any potential candidates as topics for a document. Also, the top five unigrams from $\vec{U}$ are selected as the most relevant keywords related to the extracted topic. The related words help identify the context to which the topic belongs. Thus, for a given document a set of keywords and a topic is derived.

4. Relatedness between the extracted keywords and topics

The hypothesis on which the relationship between keywords and topics can be identified is as follows. Every authentic document would have an appropriate topic and confident keywords which explains the topic which signifies the direct relationship between topic and keywords. The objective is to extract the best keywords, which have occurred more frequently as these would define the relationship with the topic.

Now for every document in the corpus, there is a topic and a set of related keywords. This information is analyzed to extract similar topics across the corpus. The similar topics are then merged by combining the set of keywords of that topic in all the documents and increasing the value of each topic maintained by a rating factor. Initially for every topic, the rating factor is set to 1. Every other occurrence of the topic in the corpus increases the rate value by one. Similar keywords for the same topic across the corpus are handled by adding their term-frequency value. Thus, finally there is a list of probable topics for a particular corpus and the keywords related to that topic. From this list, top 5 highly rated topics are selected along with their keywords as most relevant or significant for the corpus.

5. Performance evaluation of topic detection using Query categorization system

To evaluate the performance of TD , an query categorization system is designed which uses the feature set derived by TD. The Figure 1, shows the structure of proposed query categorization system. Available is the corpus containing the pre-classified text documents into pre-defined categories. Using the CSD algorithm, the concept space for the input collection is derived. Then, the TDR algorithm is used for finding the keywords and topic for each document and for the entire category. The keywords belonging to a category forms the class vector for that category. The synonyms of these keywords also relate to the class. So, the class vectors for all categories are then expanded using ,WordNet, to get the expanded class vectors.

User gives the (text) query as input. This query is pre-processed, expanded and disambiguated. The expanded query and expanded class vectors are given as input to query categorizer module which categorizes the query into one of the pre-defined categories. WordNet [16, 23] is an extensive online lexical database for English language which describes word relationships in three dimensions of Hypernym, Hyponym and Synonym. Here, are focusing on just considering synonym word relationship.

5.1 Query Processing

User gives the text query as the input to the system. The maximum size of the query that is allowed is five words. This query is preprocessed i.e. stopwords are eliminated and remaining words are stemmed. Then, the query is expanded using WordNet. WordNet is a large and comprehensive thesaurus which is manually constructed at Princeton University. It can be used as the lexical reference aid. It models the lexical knowledge of English language. WordNet is organized into the network of synonym sets (which are called synsets). A Synset in the WordNet is a group of words that are synonymous i.e. they can be interchanged without changing the meaning of the sentence under a particular context. Each sysnet represent one underlying lexical concept. These synsets are interconnected with a variety of relations (which are semantic relations between the concepts) within the open class categories of noun, verb, adjective and adverb[15]. The semantic relations for nouns include synonymy, hyponymy, troponymy, meronymy and their corresponding counterparts[16]. All synsets are strictly organized using the lexical relations and they only differ from each other by POS (part of speech) and the number of senses associated with each of the POS. A concept of a word is represented by three elements, the word itself, its part of speech and the POS num. It is denoted by a triple (word, POS, POS num).

WordNet returns the synonyms of the words and the related gloss definition for each of the available senses of the word. The words from the synonyms set and gloss definition of the query word are compared to the synonyms and gloss definition of the next word in the query. If the match is found, then the query is expanded using the sense that matched. If the query consists only of one word, then it is expanded using all the senses of that word. The algorithm 3 describes the process of query expansion using WordNet taking into account the sense of query words.

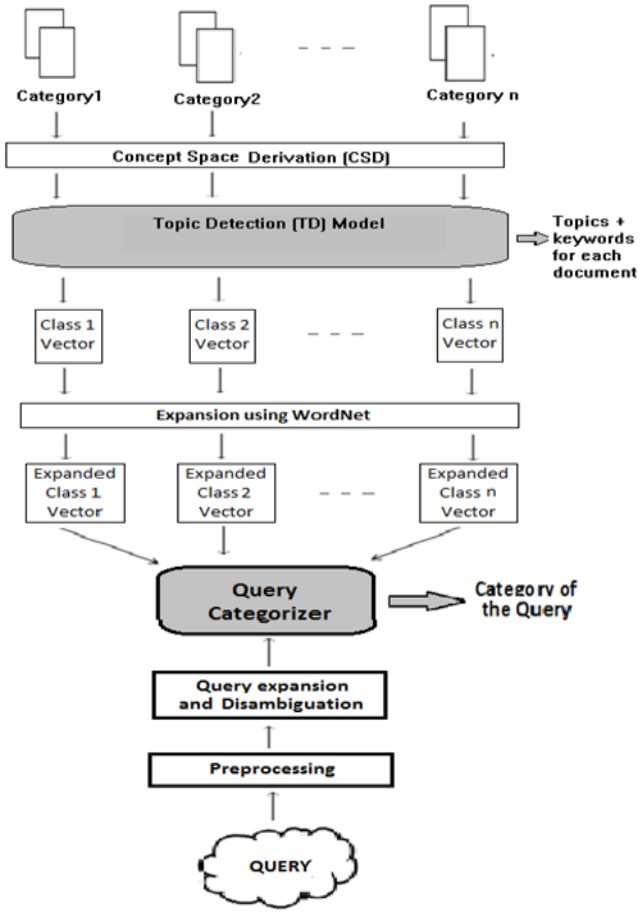

Figure 1 : Proposed Query categorization Model

Algorithm 3 : Query Expansion using Wordnet

```
Input : text query
Output : expanded query
// declare String vectors word1, word2, word3, word4, word5,
finalQuery;
// declare array of Synsets synsets[] and String str, str1;

if  (num of query words greater than 5 ) then
        print error;
else
  for i=1 to num of query words do
     synsets[wi] = getSynsets of wi;
     for j = 0 to synsets.length do
        wordList = all word forms of wi for sense j ;
        str = gloss definition of wi for sense j;
        preprocess gloss definition (str); // i.e.tokenize,
remove stopwords, symbols and change
                                          // each token to lower
case;
        wordi[j] = tokens from gloss definitions and the
wordlist (for sense j);
    enddo
 endo

// handling query of length 1
 if  (num of query words = 1) then
      finalQuery is all the tokens from word1;
 else
 for i=1 to num of query words-1 do
   for j = 0 to wordi.size-1 do
    str = wordi[j];
    tokenize str;
    for each token t in str do
      for k = 0 to word(i + 1).size-1 do
       str1 = word(i + 1)[k];
       tokenize str1;
       for each token t1 in str1 do
       if  ( t matches with t1) then
         include all tokens from str and str1 into finalQuery;
        endif
     enddo
    enddo
   enddo
 enddo
enddo
endif

finalQuery contains the tokens for the expanded query
```

5.2 Query Categorizer

The expanded query and the class vectors of all the considered categories are given to query categorizer. It categorizes query into one of the pre-defined categories using the similarity measures for the vectors like cosine similarity or measures like Euclidean distance.

For finding the cosine Similarity, suppose given two vectors containing same attributes, A and B, the cosine similarity, cos (θ) , is represented using a dot product and magnitude as follows:

$$\text{cosine similarity} = \cos(\theta) = \frac{A.B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}}$$

The resulting similarity ranges from -1 which means that there is no similarity between the two vectors, to 1 meaning both the vectors are exactly the same. For the proposed model, the cosine similarity between two vectors, namely, the expanded class vector and the query vector is determined.

Similarly, finding the euclidean distance between two points p and q is equal to the length of the line segment connecting them i.e. ($\overline{pq}$).

In Cartesian coordinates, if p = (p_1, p_2,..., p_n) and q = (q_1, q_2,..., q_n) are two points in Euclidean n dimensional space, then the distance from p to q, or from q to p is given by :

d (p, q) = d (q, p).

$$= \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

For the proposed model, the euclidean distance is the distance between the class vector and the query vector. The number of features in the class vectors defines the dimensions of the space.

6. Experimental Results

The experimentation on the topic detection model is performed using different datasets like 2G-Scam, Reuters 21578, Ohsumed. As a case study, the results for 2G-Scam are discussed. Also, the query categorization system is experimented on Reuters 21578 dataset and compared with Google desktop search tool.

6.1 Topic Detection

To present the results, TDR algorithm is applied to corpus containing documents about 2G-SCAM. Some of the topmost topics (i.e. topics having high score in corpus) are shown in table 1. After checking the results manually, the topics detected for documents are indeed found to be relevant. There are no datasets available with topic labeling along with keywords. The experimentation were also carried out on different datasets like Reuters 21578, Ohsumed. Manually analysis of the results proved that the topic derived for a class or category were relevant. Also, the set of keywords for each topic was found to be very related to the topic.

Table 1 : Some of the topics and related keywords found from 2G -Scam corpus

Topic	Keywords
reliance official	reliance, official, bail, role, scam, swan, plea, director, accuse, jail, enter, massive, court, telecom
scam chargesheet	chargesheet, mauritius, delphi, scam, information
home minister	chidambaram, minister, swamy, cbi, raja
minister Chidambaram	minister, pac, probe, chidambaram, report
telecom chairman	telecom, ngo, cbi, ambani, ruia
official allege	official, radia, cbi, allege, spokesperson

6.2 Query expansion

Query expansion module expands query using WordNet and while expansion it considers the senses of each word of the query. Table 2 shows some queries and their expanded versions. The above expansion shows that the senses of query words are indeed taken into consideration and the query gets expanded accordingly.

Table 2 : Examples of Query Expansion Method

Original Query	Expanded Query
river bank	river, bank, water, large, natural, stream, larger, creek, sloping, land, body
bank transactions	bank, account, transactions, activities, act, dealings, transacting, groups, carrying, commercial, written, transpired, banking, company, financial, institution, accepts, deposits, channels
erosion on the bank	erosion, bank, water, condition, earth's, surface, worn, action, sloping, land, slope, body
tree in data structures	tree, data, structures
tree cutting	plant, tree, part, root, slip, leaf, bud, removed, propagate, rooting, grafting, main, branches, tall, perennial, woody, trunk, forming, distinct, elevated
mining data from corpus	data, corpus, information, collection, derived, mining, facts, conclusions, drawn, writings, point, item, factual, measurement, research, principal, sum, capital
mining minerals from ores	mineral, mining, minerals, ores, solid, homogenous, substances, occurring , nature, definite, chemical, composition, ore, metal, valuable, mined

The working of query categorization system has been experimented using Google Desktop search tool. As corpus, reuters 21578 dataset is used. Google Desktop is a popular freeware desktop search tool offered by Google [17]. It has a simple Web interface which is similar to the Google.com search interface which makes it possible to use ones browser to search for information on the local computer. Google Desktop can index and manage a large variety of resources including Office documents, media files, email, zipped archives, Web history cache, and chat sessions.

Google Desktop also tracks the user's activity while viewing and editing files, reading and writing email, and surfing the Web. It creates cached copies of the tracked information, allowing the user to access it afterwards. For this reason, it is possible to search and access data, from the cache, even after the original email or file no longer exists on the system. The Google Desktop application runs a local Web server which is bound to port 4664 on the localhost network interface. For security purposes, it responds only to requests originating from the local computer.

There are no available datasets of labeled queries for a corpus. So for experimentation, manually some queries were collected related to the Reuters 21578 dataset. Tables 3 and 4 shows the output of proposed query categorization system for some sample queries. Table 3 shows the result for some queries which produce similar results for queries after expansion and before expansion. Table 4 shows the results of the queries for which the expansion procedure enhances the result. This demonstrate the usage of query expansion for the end result of the query categorizer.

Table 3 : Sample queries which give same result with and without expansion

Query	Output with and without expansion (gives same output for these queries)
export of wheat crop	wheat
export of crop	grain
import of crop	grain
carriers for load transport	ship
defense against missile attack	ship
gulf war zone	ship
trade union strike	trade
large farm	corn
subsidy for export of crop	wheat
price rise impact on export	crude
soft flour	wheat
deficit incurred	Trade
market analyst	moneyfx
mission of economic growth	trade
dollar to yen conversion	moneyfx
high rate of inflation	interest
asset of 10 min	earn

Table 4 : Sample queries where the proposed model of categorization (with expansion) gives better results

Query	Output with expansion (Output of proposed system)	Output without expansion
loading at the dock	ship	not exist
gantry cranes	trade	not exist
unload containers at dock	ship	not exist
subsidy for export of crop	grain	wheat
cultivate food	corn	acq
atmospheric requirements for cultivation	wheat	not exist
survey of fertile soil types	wheat	not exist
Insecticide and pesticide	acq	not exist
cereals cultivation	wheat	not exist
pasture for animals	grain	not exist
bread making	grain	not exist
Cakes and pastries	grain	not exist
dough and batter	wheat	not exist
processed food	corn	acq
barter method	trade	not exist
swiss capital export	trade	wheat
income tax waiver	earn	crude
treaty for reduction tariffs	trade	not exist
commercial buying and selling	trade	ship
breach of law	interest	not exist
governing body	trade	not exist
legal documents	trade	not exist
recession	trade	not exist
return on equity	moneyfx	not exist

Table 5 : Comparison of results given by the proposed model and google desktop using same sample queries

Query	Output of proposed system using derived concepts	Output using Google desktop search
export of crop	grain	corn
defense against missile attack	ship	not exist
vessel stationed at the dockyard	ship	not exist
rotterdam port issues	ship	not exist
rotterdam port	ship	ship
trade union strike	trade	not exist
trade union	trade	corn
market analyst	moneyfx	acq
mission of economic growth	trade	not exist
economic growth	trade	trade
dollar to yen conversion	moneyfx	not exist
dollar to yen	moneyfx	moneyfx
gatt trade rules	trade	not exist
gatt trade	trade	trade
japanese semiconductor industry	trade	not exist
japanese semiconductor	trade	trade
merchandise organization	trade	not exist
federal budget deficit	trade	trade
high rate of inflation	interest	interest
net profit in transaction	earn	not exist
net profit	earn	earn
revenue distribution	earn	not exist

The table 5 shows the comparison of the results using the proposed query categorization system and Google Desktop. The query is given to the google desktop search and the category to which the majority of the retrieved documents belong to, is given as the category of the query. The search is restricted to only one folder which contains all the text documents belonging to the corpus. It can be seen from the table 5 that the results given by the proposed model are better than that given by google desktop search. Also, the proposed query categorization system covers more queries i.e. the system gives correct results for some of the queries for which google desktop search give the result as 'query does not exist'. Experimental results show that the proposed system for query categorization works fairly well for queries on static corpus.

7. Conclusion

The proposed and implemented query categorization system shows satisfactory performance in categorizing the queries. The topic detection model is the most important module of the system. The algorithm proposed for topic detection works well for majority of the documents. Apart from just finding the topics, the keywords and topics found can be useful for some of the text mining applications. One, as discussed previously is text categorization. The other application where the topic detection is useful is document retrieval. For a given query, if the objective is at retrieving relevant documents from the corpus, then knowing the topic and keywords of the documents will certainly help in improving the retrieval accuracy. Also, knowing the probable topics for a corpus containing documents of a certain category, and thus knowing the important topics and keywords for a particular category can also help in the task of query categorization. Also the query expansion approach used, takes into consideration the senses of query words and expands the query accordingly. Finally, the proposed approach is one of the few approaches that do not use any web applications like search engines (unlike most of the approaches studied in literature survey) and thus can be used where privacy is a concern.

8. REFERENCES

[1] Yaming Yang and Xin Liu " A reexamination of text categorization methods" .In annual ACM conference on Research and Development in Information Retrieval . pp 42-49, 1999.

[2] Q. Mei, C. Liu, H. Su, and C. Zhai. A probabilistic approach to spatiotemporal theme pattern mining on weblogs. In *Proceedings of WWW '06*, pages 533-542, 2006.

[3] Q. Mei and C. Zhai. A mixture model for contextual text mining. In *Proceedings of KDD '06*, pages 649-655, 2006.

[4] David D. Lewis " Reuters 21578" dataset.

[5] G. Salton, A. Wong and C.S. Yang "A Vector Space Model for Automatic Indexing" Communication of the ACM , November 1975 vol 18 number 11.

[6] Xiao-Bing Xue, Zhi-Hua Zhou ," Distributional Features for Text Categorization, IEEE Transactions on Knowledge and Data Engineering, Volume 21 Issue 3, March 2009 ,Pages 428-442.

[7] C. Apte, F. Damerau, and S.M. Weiss, " Automated Learning of Decision Rules for Text Categorization", ACM Transactions on Information Systems, 1994.

[8] D. Koller and M Sahami. 1997." Hierarchically classifying documents using very few words". In *Proceedings of the International Conference on Machine Learning (ICML)*.

[9] Lewis, D., and Ringuette, M. 1994. A comparison of two learning algorithms for text categorization. In *Third Annual Symposium on Document Analysis and IR*.

[10] Y. H. LI and A. K. JAIN, " Classification of Text Documents., The COMPUTER Journal, Vol. 41, No. 8, 1998.

[11] Robert E. Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, December 1999.

[12] Moulinier et.al , Text categorization : A symbolic approach. In annual symposium on document analysis and Information retrieval (SDAIR),1996.

[13] Moulinier and Ganascia J. , " Applying and an existing machine learning algorithms to text categorization" , in Connectionist, statistical, and symbolic approaches to learning for NLP, springer verlag, 1996.

[14] Schutze H, Hull D. and Pederson J," A comparison of classifiers and document representation for the routing problem". ACM SIGIR conference on research and Development in IR, 1995.

[15] George A. Miller, Claudia Leacock, Randee Tengi, Ross T. Bunker., A Semantic Concordance., Proceedings of the 3rd DARPA Workshop on Human Language Technology, 1993.

[16] Miller, G., R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. ,Five papers on WordNet., CSL Report 43, Cognitive Science Laboratory, Princeton Uni versity, 1990.

[17] Yair Amit, Danny Allan, Adi Sharabani, Overtaking Google Desktop – A Security Analysis, A whitepaper from watch_re, 2007.

[18] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of SIGIR '99*, pages 50-57, 1999.

[19] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993-1022,2003.

[20] C. Zhai, A. Velivelli, and B. Yu. A cross-collection mixture model for comparative text mining. In *Proceedings of KDD '04*, pages 743-748, 2004.

[21] M. Steyvers, P. Smyth, M. Rosen-Zvi, and T. Griffths. Probabilistic author-topic models for information discovery. In *Proceedings of KDD '04*, pages 306-315, 2004.

[22] W. Li and A. McCallum. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of ICML*, pages 577-584, 2006.

[23] Miller, G. A., Wordnet : A lexical database for English,2010