INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATION AND MANAGEMENT

CONTENTS

CALL FOR MANUSCRIPTS

We invite unpublished novel, original, empirical and high quality research work pertaining to recent developments & practices in the area of Computer, Business, Finance, Marketing, Human Resource Management, General Management, Banking, Insurance, Corporate Governance and emerging paradigms in allied subjects like Accounting Education; Accounting Information Systems; Accounting Theory & Practice; Auditing; Behavioral Accounting; Behavioral Economics; Corporate Finance; Cost Accounting; Econometrics; Economic Development; Economic History; Financial Institutions & Markets; Financial Services; Fiscal Policy; Government & Non Profit Accounting; Industrial Organization; International Economics & Trade; International Finance; Macro Economics; Micro Economics; Monetary Policy; Portfolio & Security Analysis; Public Policy Economics; Real Estate; Regional Economics; Tax Accounting; Advertising & Promotion Management; Business Education; Business Information Systems (MIS); Business Law, Public Responsibility & Ethics; Communication; Direct Marketing; E-Commerce; Global Business; Health Care Administration; Labor Relations & Human Resource Management; Marketing Research; Marketing Theory & Applications; Non-Profit Organizations; Office Administration/Management; Operations Research/Statistics; Organizational Behavior & Theory; Organizational Development; Production/Operations; Public Administration; Purchasing/Materials Management; Retailing; Sales/Selling; Services; Small Business Entrepreneurship; Strategic Management Policy; Technology/Innovation; Tourism, Hospitality & Leisure; Transportation/Physical Distribution; Algorithms; Artificial Intelligence; Compilers & Translation; Computer Aided Design (CAD); Computer Aided Manufacturing; Computer Graphics; Computer Organization & Architecture; Database Structures & Systems; Digital Logic; Discrete Structures; Internet; Management Information Systems; Modeling & Simulation; Multimedia; Neural Systems/Neural Networks; Numerical Analysis/Scientific Computing; Object Oriented Programming; Operating Systems; Programming Languages; Robotics; Symbolic & Formal Logic and Web Design. The above mentioned tracks are only indicative, and not exhaustive.

Anybody can submit the soft copy of his/her manuscript **anytime** in M.S. Word format after preparing the same as per our submission guidelines duly available on our website under the heading guidelines for submission, at the email addresses: **1** or **info@ijrcm.org.in**.

GUIDELINES FOR SUBMISSION OF MANUSCRIPT

1. **COVERING LETTER FOR SUBMISSION**:

DATED: _____________

THE EDITOR

IJRCM

Subject: **SUBMISSION OF MANUSCRIPT IN THE AREA OF ____.**

(e.g. Computer/IT/Finance/Marketing/HRM/General Management/other, please specify).

DEAR SIR/MADAM

Please find my submission of manuscript titled '____' for possible publication in your journal.

I hereby affirm that the contents of this manuscript are original. Furthermore, it has neither been published elsewhere in any language fully or partly, nor is it under review for publication anywhere.

I affirm that all author (s) have seen and agreed to the submitted version of the manuscript and their inclusion of name (s) as co-author (s).

Also, if our/my manuscript is accepted, I/We agree to comply with the formalities as given on the website of journal & you are free to publish our contribution to any of your journals.

NAME OF CORRESPONDING AUTHOR:

Designation:

Affiliation with full address & Pin Code:

Residential address with Pin Code:

Mobile Number (s):

Landline Number (s):

E-mail Address:

Alternate E-mail Address:

2. **INTRODUCTION**: Manuscript must be in British English prepared on a standard A4 size paper setting. It must be prepared on a single space and single column with 1" margin set for top, bottom, left and right. It should be typed in 8 point Calibri Font with page numbers at the bottom and centre of the every page.
3. **MANUSCRIPT TITLE**: The title of the paper should be in a 12 point Calibri Font. It should be bold typed, centered and fully capitalised.
4. **AUTHOR NAME(S) & AFFILIATIONS**: The author (s) full name, designation, affiliation (s), address, mobile/landline numbers, and email/alternate email address should be in italic & 11-point Calibri Font. It must be centered underneath the title.
5. **ABSTRACT**: Abstract should be in fully italicized text, not exceeding 250 words. The abstract must be informative and explain the background, aims, methods, results & conclusion in a single para.
6. **KEYWORDS**: Abstract must be followed by list of keywords, subject to the maximum of five. These should be arranged in alphabetic order separated by commas and full stops at the end.
7. **HEADINGS**: All the headings should be in a 10 point Calibri Font. These must be bold-faced, aligned left and fully capitalised. Leave a blank line before each heading.
8. **SUB-HEADINGS**: All the sub-headings should be in a 8 point Calibri Font. These must be bold-faced, aligned left and fully capitalised.
9. **MAIN TEXT**: The main text should be in a 8 point Calibri Font, single spaced and justified.
10. **FIGURES &TABLES**: These should be simple, centered, separately numbered & self explained, and titles must be above the tables/figures. Sources of data should be mentioned below the table/figure. It should be ensured that the tables/figures are referred to from the main text.
11. **EQUATIONS**: These should be consecutively numbered in parentheses, horizontally centered with equation number placed at the right.
12. **REFERENCES**: The list of all references should be alphabetically arranged. It must be single spaced, and at the end of the manuscript. The author (s) should mention only the actually utilised references in the preparation of manuscript and they are supposed to follow **Harvard Style of Referencing**. The author (s) are supposed to follow the references as per following:

- All works cited in the text (including sources for tables and figures) should be listed alphabetically.
- Use (**ed.**) for one editor, and (**ed.s)** for multiple editors.
- When listing two or more works by one author, use --- (20xx), such as after Kohl (1997), use --- (2001), etc, in chronologically ascending order.
- Indicate (opening and closing) page numbers for articles in journals and for chapters in books.
- The title of books and journals should be in italics. Double quotation marks are used for titles of journal articles, book chapters, dissertations, reports, working papers, unpublished material, etc.
- For titles in a language other than English, provide an English translation in parentheses.
- The location of endnotes within the text should be indicated by superscript numbers.

PLEASE USE THE FOLLOWING FOR STYLE AND PUNCTUATION IN REFERENCES:

BOOKS

- Bowersox, Donald J., Closs, David J., (1996), "Logistical Management." Tata McGraw, Hill, New Delhi.
- Hunker, H.L. and A.J. Wright (1963), "Factors of Industrial Location in Ohio," Ohio State University.

CONTRIBUTIONS TO BOOKS

- Sharma T., Kwatra, G. (2008) Effectiveness of Social Advertising: A Study of Selected Campaigns, Corporate Social Responsibility, Edited by David Crowther & Nicholas Capaldi, Ashgate Research Companion to Corporate Social Responsibility, Chapter 15, pp 287-303.

JOURNAL AND OTHER ARTICLES

- Schemenner, R.W., Huber, J.C. and Cook, R.L. (1987), "Geographic Differences and the Location of New Manufacturing Facilities," Journal of Urban Economics, Vol. 21, No. 1, pp. 83-104.

CONFERENCE PAPERS

- Garg Sambhav (2011): "Business Ethics" Paper presented at the Annual International Conference for the All India Management Association, New Delhi, India, 19–22 June.

UNPUBLISHED DISSERTATIONS AND THESES

- Kumar S. (2011): "Customer Value: A Comparative Study of Rural and Urban Customers," Thesis, Kurukshetra University, Kurukshetra.

ONLINE RESOURCES

- Always indicate the date that the source was accessed, as online resources are frequently updated or removed.

WEBSITE

- Garg, Bhavet (2011): Towards a New Natural Gas Policy, Economic and Political Weekly, Viewed on July 05, 2011 http://epw.in/user/viewabstract.jsp

AUTOMATIC INFORMATION COLLECTION & TEXT CLASSIFICATION FOR TELUGU CORPUS USING K-NN ALGORITHM

NADIMAPALLI V GANAPATHI RAJU
ASSOCIATE PROFESSOR
DEPARTMENT OF CSE
GOKARAJU RANGARAJU INSTITUTE OF ENGINEERING & TECHNOLOGY
HYDERABAD

VIDYA RANI V
P.G. STUDENT
DEPARTMENT OF MCA
GOKARAJU RANGARAJU INSTITUTE OF ENGINEERING & TECHNOLOGY
HYDERABAD

BHAVYA SUKAVASI
P.G. STUDENT
DEPARTMENT OF MCA
GOKARAJU RANGARAJU INSTITUTE OF ENGINEERING & TECHNOLOGY
HYDERABAD

SAI RAMA KRISHNA CHAVA
P.G. STUDENT
DEPARTMENT OF MCA
GOKARAJU RANGARAJU INSTITUTE OF ENGINEERING & TECHNOLOGY
HYDERABAD

ABSTRACT

Many algorithms have been implemented to the problem of Automatic Information Collection and Text Categorization. Most of the work in this area was carried out for the English corpus; on the other hand very few researches have been carried out for the Telugu corpus. In this project we have implemented the k- Nearest Neighbor (k-NN) algorithm, which is known to be one of top performing classifiers applied for the English text. The results show that k-NN is applicable to Telugu text.

KEYWORDS

Text Collection, Text classification, Term Weighting, Similarity Measuring, Telugu Script, Unicode, k-NN Classifier.

INTRODUCTION

Information gathering process produces a collection of data (corpus). A corpus is a large and representative collection of language material stored in a computer process able form. Corpus provides the basic language data from which a variety of lexical resources can be generated. We have collected Telugu script corpus from www.uni.medhas.org which provides Unicode Telugu data.

Huge amount of electronic textual information is increasingly available through the internet and organizations, making the process of retrieving data and information turns into a real problem without good indexing and summarization of documents contents. Text or document categorization is one solution for the problem. Many statistical learning methods have been applied in the field of text categorization in the recent years [1] [4], this includes regression models, nearest neighbor classifiers, Bayes belief networks, decision trees rule learning algorithms, neural networks, and inductive learning techniques. In this project we chose the nearest neighbor machine learning approach for its simplicity, effectiveness, and also because of its applicability with small number of training patterns.

RETRIEVAL - NEED OF CLASSIFICATION

Information retrieval (IR) corresponds to representation, storage, organization, and access to information items. IR has no restriction on the format. But typically, retrieval systems include letters, documents of all sorts, newspaper articles, books, research articles etc. Usually the IR system when ever finds a query with less number of relevant documents it retrieves the other members of class having relevant documents which increase precision and recall.

In a classification task, the precision for a class is the number of **true positives** (i.e. the number of items correctly labeled as belonging to the positive class) divided by the total number of elements labeled as belonging to the positive class (i.e. the sum of true positives and **false positives**, which are items incorrectly labeled as belonging to the class). Recall in this context is defined as the number of true positives divided by the total number of elements that actually belong to the positive class (i.e. the sum of true positives and **false negatives**, which are items which were not labeled as belonging to the positive class but should have been).

Often, there is an inverse relationship between precision and recall, where it is possible to increase one at the cost of reducing the other. For example, an information retrieval system (such as a search engine) can often increase its recall by retrieving more documents, at the cost of increasing number of irrelevant documents retrieved (decreasing precision).

In the context of classification tasks, the terms **true positives**, **true negatives**, **false positives** and **false negatives** are used to compare the given classification of an item (the class label assigned to the item by a classifier) with the desired correct classification (the class the item actually belongs to). This is illustrated by the table below:

Precision and recall are then defined as:

TABLE 1: THE RESULT OF CLASSIFICATION

Expected Result		Positive	Negative
obtained result	Positive	**tp** (true positive)	**fp** (false positive)
	Negative	**fn** (false negative)	**tn** (true negative)

Precision=tp/(tp+fp) Recall=tp/(tp+fn)

AUTOMATIC INFORMATION COLLECTION

There are great amounts of textual information on the internet and in computerized systems of different associations and companies; it is very difficult to collect manually this huge amount of information for that it is very time consuming. Automatic Information Collection helps in reducing the time needed to collect hundreds or thousands of daily arrived documents.

There are no tools existed for the collection of data from the WWW automatically. This is necessary for a classification tool for dynamic acceptance Telugu test document. The system can automatically collect Telugu data from the Internet by the specified link to use it as document to be classified.

Downloading a web file sends a "request" to a web server using the standard HTTP 1.1 protocol. The server processes your request and sends you a "response". The response's "header" tells you the file's size, last modified date, MIME type, and other useful information. Finally, the response's payload is the file itself.

TEXT CATEGORIZATION: OVERVIEW

The goal of this clustering method is to simply separate the data based on the assumed similarities between various classes. Thus, the classes can be differentiated from one another by searching for similarities between the data provided.

The k-Nearest Neighbor is suitable for data streams. kNN does not build a classifier in advance. When a new sample arrives, k-NN finds the k neighbors nearest to the new samples from the training space based on some suitable similarity or distance metric.

k-NN is a good choice when simplicity and accuracy are the predominant issues. k-NN can be superior when resident, trained and tested classifiers has a short useful lifespan, such as in the case with the data streams where new data is added rapidly and the training set is ever changing. k-NN does not rely on prior probabilities, and it is computationally efficient. The main computation is the sorting of the training documents in order to find out the k nearest neighbors for the test document.

k-Nearest Neighbor is useful when there are less than 20 attributes per instance, there is lots of training data, training is very fast, learning complex target functions and don't want to lose information.

The disadvantages of using such a function are that it is slow in sorting out queries and irrelevant attributes can fool the neighbor.

DATA PREPROCESSING AND INDEXING

Data pre-processing comprises six sub-components including document conversion function word removal, word stemming, feature selection, dictionary construction, and feature weighting. The functionality of each component is described as follows:

a. Document converting –converts different types of documents such as XML, PDF, HTML, DOC format to plain text format.

b. Function word removal –removes topic-neutral words such as articles, Prepositions, conjunctions etc. from the documents

c. Word stemming –standardizes word's suffixes (e.g., labeling -- label, introduction -introduct).

d. Indexing tables – KNN algorithm for classification is based on Statistics like weights for that the creation of tables for each document and calculation of TFIDF values.

e. Term weighting – For most existing document clustering algorithms, documents are represented by using the vector space model. In this model, each document d is considered as a vector in the term-space and represented by the term frequency (TF) vector:

$$d_{tf} = [tf_1, tf_2, \ldots, tf_D]$$

where tfi is the frequency of termi in the document, and D is the total number of unique terms in the text database. Normally there are several preprocessing steps, including the removal of stop words and the stemming on the documents. A widely used refinement to this model is to weight each term based on its inverse document frequency (IDF) in the document collection. The idea is that the terms appearing frequently in many documents have limited discrimination power, so they need to be deemphasized .This is commonly done by multiplying the frequency of each term i by $log(n/df_i)$, where n is the total number of documents in the collection, and dfi is the number of documents that contain term i (i.e., document frequency).

Thus, the tf–idf representation of the document d is:

$$d_{tf-idf} = [tf_1 \log(n/df_1), tf_2 \log(n/df_2), \ldots, tf_D \log(n/df_D)]$$

when the document vectors are normalized, only the vector product has to be calculated, which can be done faster than the computation of the Euclidian Distance, especially for sparce vectors. The cosine similarities measure the cosine of the angle between two vectors. The bigger the value, the smaller is the actual angle and the more similar are the two vectors. This makes the distance metrics also independent of the length of the documents, as document vectors of different length, but with the same angle to each other, will have zero distance.

For example, given term frequency (TF) weights for science document:

TABLE 2: TERM WEIGHTS FOR A DOCUMENT

sports2				
ID	Word	Count	Tf	tfidf
35	మైలేజీ	1	0.007	25
44	జూలైలో	2	0.015	10
59	బియ్యం	1	0.007	25
61	అధికమై	1	0.007	50
107	లాభా	1	0.007	16.666
116	వ్యవహ	1	0.007	50
179	ప్రస్తుత	14	0.105	350
198	పట్ట	1	0.007	50
226	పుంతలు	1	0.007	0.7
227	పుంజుకొంటోం	2	0.015	0.375
236	తొక్కిస్తాం	1	0.007	4.545
275	కాలంలో	1	0.007	0.7

IMPLEMENTATION

We used Python and Java programming languages for implementation of this work. Python is an interpreter, interactive, object-oriented, extensible programming language. There are two types of strings in Python: byte strings and Unicode strings. Python handles Unicode strings same as that of byte strings. Unicode strings are encoded in UTF-8 format. Python has codecs module which convert UTF-8 encoded byte strings to Unicode strings. We used Java (JDBC) to make database operations using MS Access.

The system can be divided into following modules:

- Automatic Information Collection from web
- Preprocessing and extraction of words from corpus
- Applying stop word removal
- Developing a new N-gram based technique for stemming of the data set.
- Developing the Index tables
- Dimensionality reduction
- Implementing the k-NN Classifier
- Performance measures

AUTOMATIC INFORMATION COLLECTION FROM WEB

- Here we get the data from the news-papers which are available online in the internet. There are some websites providing us the daily news papers in the Telugu script like www.uni.medhas.org.
- This data is called as corpus which will be useful to study the canonical structure of the script. These files are saved in UTF_8 format so that the content in the files can be visible.

Getting a web file using these classes always includes these steps:

- Create a URL object from a URL string.
- Open a URLConnection object from the URL object.
- Set up the web server request by calling set* methods on the URLConnection object.
- Send the request to the web server by calling connect () on the URLConnection object.
- Get the web server response by calling get* methods on the URLConnection object.
- Decode the file content based upon the content type.

PREPROCESSING AND EXTRACTION OF WORDS FROM THE CORPUS

- In this step, the files in the corpus are preprocessed where preprocessing is the process by which we will remove or ignore the characters which are other than in the Telugu language.
- All the numbers, Roman characters and any unwanted letters except "space" will be removed.
- This can in process described below.
    - First, we read all the files of the category.
    - Then, read each file into a string variable.
    - Next step is identifying each individual character.
    - Remove each character which is unwanted.
    - Final step is to write the string to a file.
- From the preprocessed file we will extract the words using a space identifier which separates each word.

APPLYING STOP WORD REMOVAL

Many times, it makes sense to not index "stop words" during the indexing process. Stop words are words which have very little informational content. These are words such as: **గురించి**, **ఎంత**, **ఎందుకు**, **ఒక**, **అవి**, **వాళ్లు**, **వారు** etc.

Studies have shown that by removing stop words from the index, you may benefit with reduced index size without significantly affecting the accuracy of a user's query. Care must be taken however to take into account the user's needs.

- Here we will remove the words such as articles, Prepositions, conjunctions etc. from the documents.

STEMMING OF WORDS USING N-GRAM DATA STRUCTURES

Stemming is common form of language processing in most, "A failure to process morphological variants results in retrieving only 2% - 10% of the documents retrieved with such processing".

It is the step we develop three dictionaries having key as stem (bigram/trigram/quadgram) and value as words for those corresponding N-grams.

- Here we will give priority order as higher priority to quadgram, priority levels towards trigram and bigram having less priority.
- Starting with quadict if a key is quadict having two or more words as value then we will consider the key as stem and push into a stemmed dictionary.
- The above step is repeated for tridict and bidict.
- Now stemmed dictionary containing some key value pairs by giving higher priority to quadgram we will remove if any other values in the dictionary contains same words which under come in the values of quadgram.
- If a word is exactly of length 2 or 3 or 4 and doesn't have any morphological variants then that word itself is considered as root.

DEVELOPING THE INDEX TABLES

k-NN algorithm for classification is based on Statistics like weights for that the creation of tables for each document and calculation of TFIDF values are necessary.

- For the table corresponding to a file to have values first we should have another file called count with frequency each stem word for that file.
- These values are then inserted into the file by systematically processing count file.
- Once all the training documents have its associated initial tables we can calculate Inverse Document Frequency (IDF) and Term Frequency Inverse Document Frequency (TFIDF).

DIMENSIONALITY REDUCTION

- Feature selection is performed here using Document Frequency Thresholding
- Words doesn't occur in just one document are removed based on the assumption that rare words do not affect category prediction.

FINDING TERM WEIGHTS & IMPLEMENTING THE CLASSIFIER

Classification is the process which assigns one or more labels or no label at all to a new (unseen) document. There are many machine learning algorithms which have been applied to the problem of text categorization, ranging from statistical methods. There is already one classifier implemented in Text Categorization using k-NN, the k-nearest neighbor classifier.

K-NEAREST NEIGHBOR

The cosine similarity measure is commonly used in Information Retrieval [Sal89] and hence is adopted as the basic similarity measure in k-NN. The weighted cosine measure between document X and Y with weight vector W and set of terms (or words) T as

$$sim(di, dj) = \frac{\sum_{k=1}^{t} w_{di} . w_{dj}}{\sqrt{\sum_{k=1}^{t} (w_{di})^2 . \sum_{k=1}^{t} (w_{dj})^2}}$$

Where w_{di} and w_{dj} are normalized TF of word t for di and dj, respectively. Select k nearest training documents, where the similarity is measured by the cosine between a given testing document and a training document.

- Using cosine values of k nearest neighbors and frequency of documents of each class i in k nearest neighbors, sort the classes based on their cosine values.
- Assign (i.e., classify) the testing document a class label which has maximum k value.

PERFORMANCE MEASURES

Precision and recall alone do not say much about the effectiveness of the classifier. Hence, it is necessary to compute different standard values which combine precision and recall, to derive a robust measure of the effectiveness of the classifier.

ALGORITHMS USED IN IMPLEMENTATION

1. Algorithm for Automatic Information Collection.
2. Algorithm for Text Classification using k-NN algorithm.

ALGORITHM FOR AUTOMATIC INFORMATION GATHERING

```
Input website URL
Read the data from url and write it into web_url file
Preprocess the web_url file
Copy all the web links from web_url
        Save the links into url_links file
Read the each link from url_links
For each link
If ascii > 255 && ascii==10 && ascii==32
        Read the data from site
        write it into sites folder as Telugu file
        read all the web links from current web page
        write it into sites folder as Telugu_url file
repeat this process until EOF
```

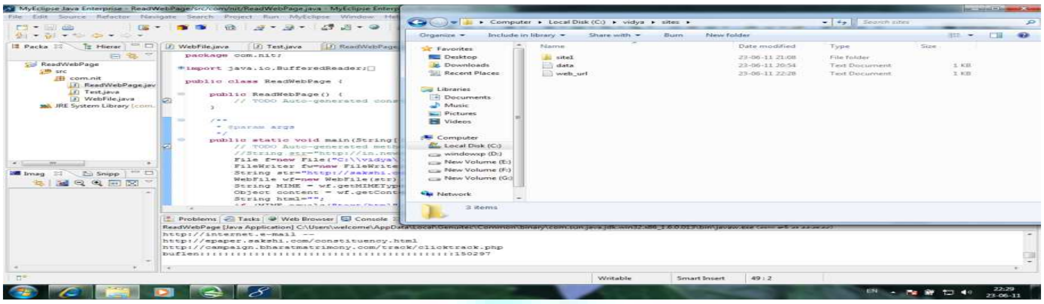
ALGORITHM FOR TEXT CLASSIFICATION USING K-NN ALGORITHM

```
Read the file from web
Preprocess the file
Apply stop word removal
Apply stemming process
Index the stemmed words
Calculate DF value
If DF==0
Eliminate the word from index table
Find the IDF value of test document
Find the associated TF*IDF value of test document
Calculate cosine similarity measure with each trained set
Assume the k value
Take the k highest similarity values
Consider the class which has maximum number of k- nearest neighbors in it
```

Classify the test document into that class

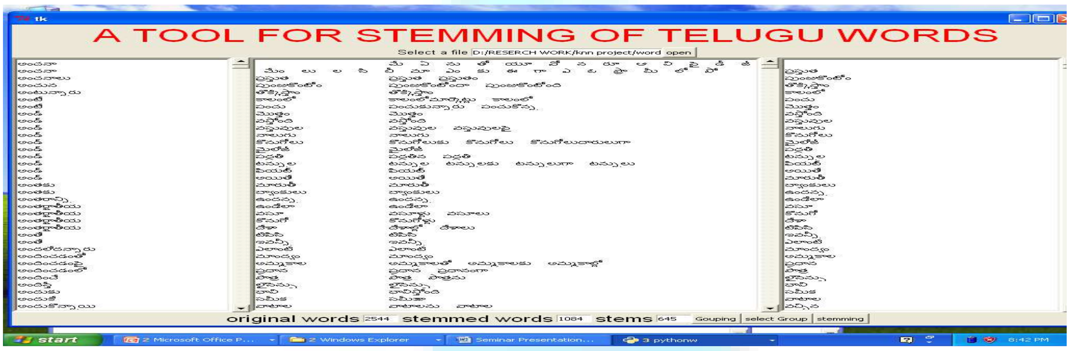
SCREENS

FIG 1: AUTOMATIC INFORMATION COLLECTION



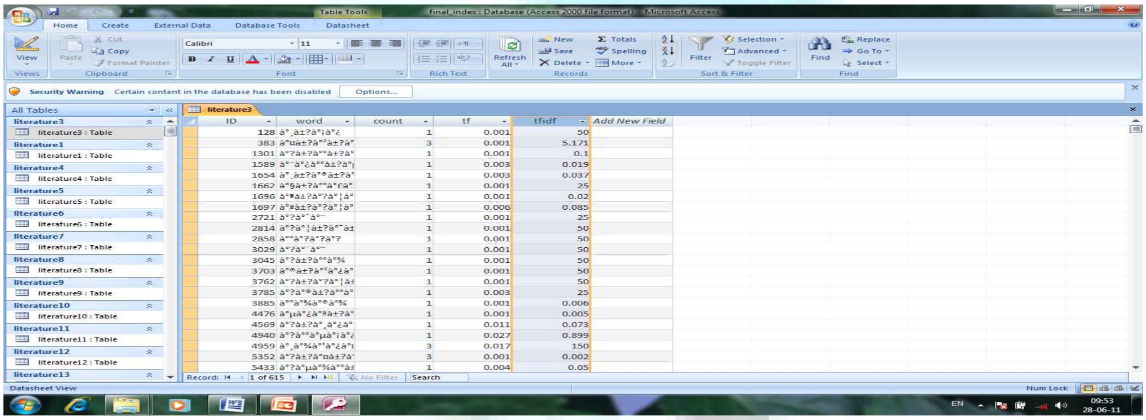
This screen shot shows the source code for information collection tool and the folder containing collected data.

FIG 2: GUI OF N-GRAM BASED STEMMING



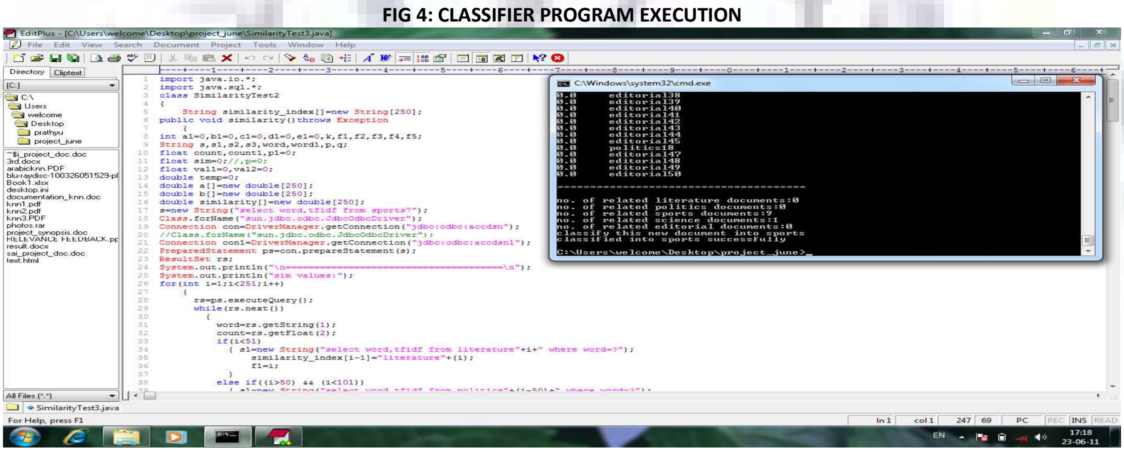
The above screen shot shows the three columns each column showing sorted words, stem word and their associated morphological words, and stem words respectively. The screen shot is obtained when the program for stemming is executing. For the Stemming the input is all the words collected from all file in corpus.

FIG 3: A SAMPLE INDEX FILES IN DATABASE



The above screen shot shows the index file generated in MS – Access in which for each word TF-IDF values are calculated and are inserted into the file.

FIG 4: CLASSIFIER PROGRAM EXECUTION



The above screen shot shows source code and execution of the classifier program in which cosine similarity is calculated in between test document and each training document for finding k-nearest neighbors.

RESULTS

In the training phase of the system we trained 250 documents belonging to 5 different classes business, science, sports, politics and rivers. Then during testing we took a Telugu document containing information related to sports. Then we executed classifier program at different values of K and the results are analyzed. Results are drawn for different values of k for k-Nearest Neighbor Classification for Telugu documents. The results are tabulated as the following table.

Table 3: Effect of value of K on Classification result

Value of k	Number of Similar Documents from Each class					Classified To
	Business	Science	Sports	Politics	Rivers	
1	0	0	1	0	0	Sports
5	0	1	4	0	0	Sports
10	0	1	9	0	0	Sports
20	2	4	11	2	1	Sports
50	10	14	12	5	9	Science

We can clearly conclude that as the value of k increases up to a certain limit maximum correct similar documents are from SPORTS. If the k value increases above this limit the proportionality of correct similar documents decrease. For our classifier the limit is 10.

For the above table we calculated precision and recall to determine best value of k which optimizes them. The calculated precision, recall are tabulated as below for easy evaluation of the result.

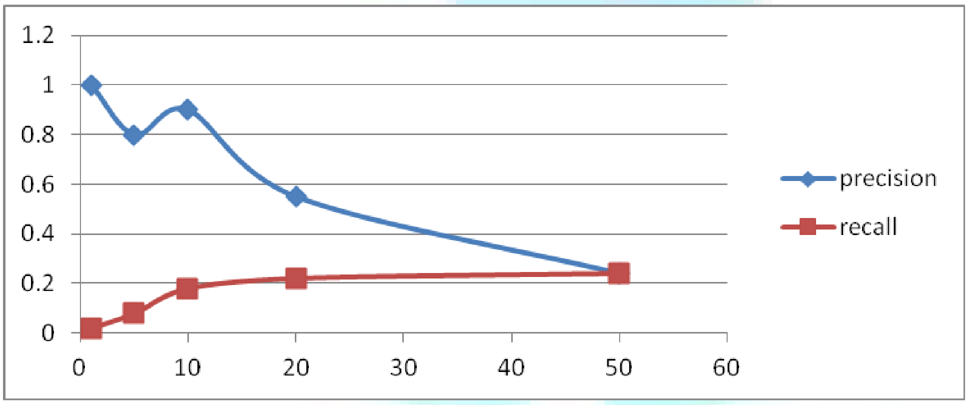
TABLE 4: PRECISION AND RECALL AT DIFFERENT VALUES OF K

K	Precision	Recall
1	1	0.02
5	0.8	0.08
10	0.9	0.18
20	0.55	0.22
50	0.24	0.24

A graph is drawn for precision and recall at different values of k and is shown below.

FIG 5: GRAPH OF PRECISION AND RECALL AT DIFFERENT VALUES OF K

Precision or recall



Value of k

The precision, recall graph must conclude the best value of k for k-NN classification. The precision value is high at k=1 but recall is low. The optimum precision and recall are obtained at k=10. At k=20 recall has its highest value but the precision has fallen sharply. So the conclusion is 10 is the best value for k in k-NN classification for Telugu document.

CONCLUSION

Our Automatic Information Collection can only eliminate English characters from the links but not other. We can extend the filtering for other languages also. The system can be extended for any other language collection. Fortunately, there is still a lot of work to do. New corpora and weighting functions can be implemented. It is also possible to export the document vectors after the dimensionality reduction to other data formats, which can then be applied to several other classifiers.

REFERENCES

1. Li Baoli1, Yu Shiwen1, and Lu Qin2 An Improved k-Nearest Neighbor Algorithm for Text Categorization
2. Gongde Guo1 , Hui Wang1 , David Bell 2, Yaxin Bi2, and Kieran Greer An k-NN Model-based Approach and Its Application in Text Categorization
3. Eui-Hong (Sam) Han George Karypis, Vipin Kumar Text Categorization Using Weight Adjusted k-Nearest Neighbor Classification
4. Fabrizio Sebastiani Text Categorization
5. Renato Fernandes Corrêa, Teresa Bernarda Ludermir Automatic Text Categorization
6. Gerarld Salton and Christopher Buckley Term-weighting approaches in automatic text retrieval
7. Aigars Mahinovs and Ashutosh Tiwari Text Classification method review
8. Gerald J.Kowalski, Mark T. Maybury Information Storage and Retrieval Systems Theory and Implementation
9. Williams B. Frakes and Ricardo Baeza- Yates Information Retrieval: Data Structures & Algorithms

REQUEST FOR FEEDBACK

Dear Readers

At the very outset, International Journal of Research in Computer Application and Management (IJRCM) acknowledges & appreciates your efforts in showing interest in our present issue under your kind perusal.

I would like to request you to supply your critical comments and suggestions about the material published in this issue as well as on the journal as a whole, on our E-mails i.e. **infoijrcm@gmail.com** or **info@ijrcm.org.in** for further improvements in the interest of research.

If you have any queries please feel free to contact us on our E-mail **infoijrcm@gmail.com**.

I am sure that your feedback and deliberations would make future issues better – a result of our joint effort.

Looking forward an appropriate consideration.

With sincere regards

Thanking you profoundly

Academically yours

Sd/-

Co-ordinator