ACCURATE AND EFFICIENT NEAREST KEYWORD SET SEARCH USING TERM FREQUENCY AND INVERSE DOCUMENT FREQUENCY IN DATASETS

K.Manjusha[1], C.Shoba Bindu[2,] P.Dileep Kumar Reddy[3]
[1]M.Tech (SE), [2] Professor, [3]Research Scholar
[1]Department of CSE
JNTUACE, Anantapur, India

Abstract: In the present computerized world the measure of information is developing step by step. It is hard to investigate the huge dataset for a given query as well as to accomplish more accuracy on user query. Frequently query will search on dataset for accurate keyword match and it does not discover the nearest keyword set for accuracy. Vishwa karma et.al. Proposed a strategy called ProMiSH (Projection and Multi Scale Hashing) that utilizes random projection and hash based index structures to accomplish high scalability and speedup. But the memory usage of Random projection and multi scale hashing increases when the number of dimensions of data points increases. So, there is a require to assign some weights to the keywords to find the nearest keyword set for accuracy. The Term Frequency (TF) and Inverse Document Frequency (IDF) weighting technique are utilized to discover the efficient nearest keyword set search. Performance of those techniques provide better performance in terms of accuracy, dimension reduction rate and response time.

INDEXTERMS - Querying, Datasets, Indexing. Hashing, TF-IDF

I. INTRODUCTION

Objects (e.g., images, chemical compounds and documents) are often described by a collection of suitable features, and are normally represented as points in a multi-dimensional feature space. For example, images are represented using colour characteristic vectors, and documents are represented by using frequently have descriptive text information like tags or keywords related to objects. Every data point in the dataset has a set of keywords, for example , Fig.1 shows the representation of a NKS query having more than a set of two-dimensional data points. Each point is tag with a position of keywords query Q = {a, b, c} the set of points {7, 8, 9} contain all the query keywords {a, b, c} and forms the tightest group of similar items compared with any other set of points, cover all the query keywords. Therefore the set {7, 8, 9} is the top results for the query Q. NKS queries are used for various applications such as location specific information using keyword queries on the web and graph pattern search and sharing photos in social websites.

Photo sharing in social websites (Face book) where images are tagged with locations and people name. These images can be set in a high-dimensional characteristic space of stability, colour, or shape. Using query we can find a group of same images which contains a set of people.

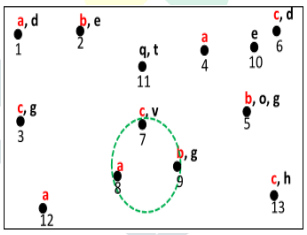


Fig .1.An example of NKS query on a keyword tagged dataset

Vishwa karma et.al.[1] proposed ProMiSH (Projection and Multi-Scale Hashing), which is capable for preparing the Nearest watchword set queries fastly. ProMiSH ensures that it finds the best possible top-k results. ProMiSH is more effective in terms of response time and accuracy and is capable of obtaining near-best results. ProMiSH-E that discovers best subset of points and ProMiSH-A which searches near optimal results with better effieciency. Nearest keyword queries always need coordination for queries, so it is difficult to develop a capable method to solve NKS queries. This paper proposes a scoring method for position of the result sets. It includes design and implementation of a TF - IDF algorithm that can assign weights to the keywords of point. It also scores the document using cosine similarity model. The proposed system provides a considerably better performance in terms of exactness, dimension reduction rate and response time. The paper is organised as follows. Review of literature can be summarized in section 2. The proposed work is described in section 3. Section 4 demonstrates Experimental results. The conclusion is followed in Section 5.

II. RELATED WORKS

A selection of queries semantically changed from NKS queries, have been studied on text rich spatial datasets. By Long et.al [2] Location- definite keyword queries on the web and in the geographic information systems [7],[8],[9]were previously answered using a combination of rectangular tree and inverted list. Felip et. al.[9]developed Information retrieval tree to place objects from spatial datasets based on a grouping of their distance to the query locations and the significance of their text descriptions to the query keywords.Cong et al.[6]included rectangular tree and inverted list file to answer a query similar to Felipe et al.[9]using a different position function. Zhou et al.[7]calculated text importance and location proximity independently, and then joined the ranking scores.

Cao et al.[5] and Long et al.[2] projected algorithms to improve a group of latitude and longitude web objects such that the group of keywords cover the query keywords and the objects in the groups are nearby to the query location and have the lowest inner-object spaces.

Other correlated queries comprise collective nearest keyword search in spatial databases[3], top-k privileged query[9], top-k sites in a spatial data based on their ability on characteristic points[8], and best possible location queries[10],[9] are explained. Tree-based indexes, such as R-Tree[4]and M-Tree[10],have been widely investigated for nearest neighbour search in high dimensional spaces. These indexes neglect to scale to dimensions more than 10 due to the curse of dimensionality [10].

Recently vishwa karma et. al.[1] suggests a new method called projection and multiscale hashing based on random projection and hashing. By using this index they developed ProMISH, it finds the best separation of points exactly. and it searches near best results with better efficiency. They proved that ProMISH is quicker than tree based technique, with multiple orders of magnitude performance development. And the memory usage of this technique grows slowly when the number of dimensions in data points increase.

However, their work does not support scoring schemes for ranking the result sets. This paper presents a term frequency and inverse document frequency that needs assigning weights to the keywords of points. Then, each group of points can be scored based on distance between points and weights of keywords.

III. PROPOSED MODEL

The proposed work assigns weight to the keywords of points using Term Frequency and Inverse Document Frequency algorithm. Where each group of points can be scored based on distance between points and weights of keywords. The Term frequency and inverse document frequency weight method is constructed with two words: the first word is Term Frequency (TF) and second word is Inverse Document Frequency(IDF). We give high score to the word which is important and that appears frequently in a document. We give low score to the word means which is not a uncommon identifier, and that appears in many documents. Some Common words like "the" and "for" which are shown in many documents. We can compute the term frequency as the number of times keyword appear in a document divided by total number of words in a document. Inverse Document Frequency provides how much information the word provides that is either common or rare across all the document. We can compute inverse document frequency as the total number of documents by number of documents that containing a term 't'. For example, document having 100 words, the term 'rat' appear 13 times, the Term frequency of the word 'rat' is TF(rat) = 13/100 i.e. 0.13. IDF (Inverse document frequency) of a word is measure of how significant that term is throughout the web. For example, say the term 'rat' appears 10 million times in the whole corpus (i.e. web). Let's assume there are 0.4 million documents that contain such a huge number of 'rat', then the IDF (i.e. log {DF}) is given by the total number of documents divided by the number of documents containing the term 'rat'.

IDF (Inverse document frequency) of a word is measure of how significant that term is throughout the web. For example, say the term 'rat' appears 10 million times in the whole corpus (i.e. web). Let's assume there are 0.4 million documents that contain such a huge number of 'rat', then the IDF (i.e. log {DF}) is given by the total number of documents divided by the number of documents containing the term 'rat'.

IDF (rat) = log (10.000.000/*400*.000) i.e. 1.63

∴ W rat = (TF*IDF) rat = 0.13* 1.63 = 0.2119.

The proposed system work as follows:

Null Stemmer: A dummy stemmer that performs no stemming at all.

TF: Sets whether if the word frequencies in a document should be transformed into$f_{ij} \times \log$(no of documents with word i)where f_{ij} is the frequency of word i in document (instance) j.

IDF: Sets whether if the word frequencies in a document should be transformed into $\log(1+f_{i,j})$where f_{ij} is the frequency of word i in document (instance) j.

Algorithm:

Input: Dataset D={D_1,$D_{2....}D_n$}

Where D->Data set

D_i ->Document

Output: F={F_1,F_2 ...F_n}

Where F->A set of selected features from D

F_i->is the selected features

Begin

Load D

For(i=1;i≤N)//Where N is no of documents

{

Perform tokenizing [D_I]

Append the tokens T_{list}

}

WC=find length (T_{list})

Sc=find (SW_{list})

For(i=0;i<wc)

For (j=0;j≤sc)

If(SW_{list}(j)==T_{list}(i))

S_{list}={S_1, $S_{2,...}$ S_n}

Eliminate T_{list}

Else

Append R_{list}

X=length of NS

For(i=0;i<x)

{

Perform stemming R_{list}(i)and store V_{list}

}

Calculate TF, IDF{

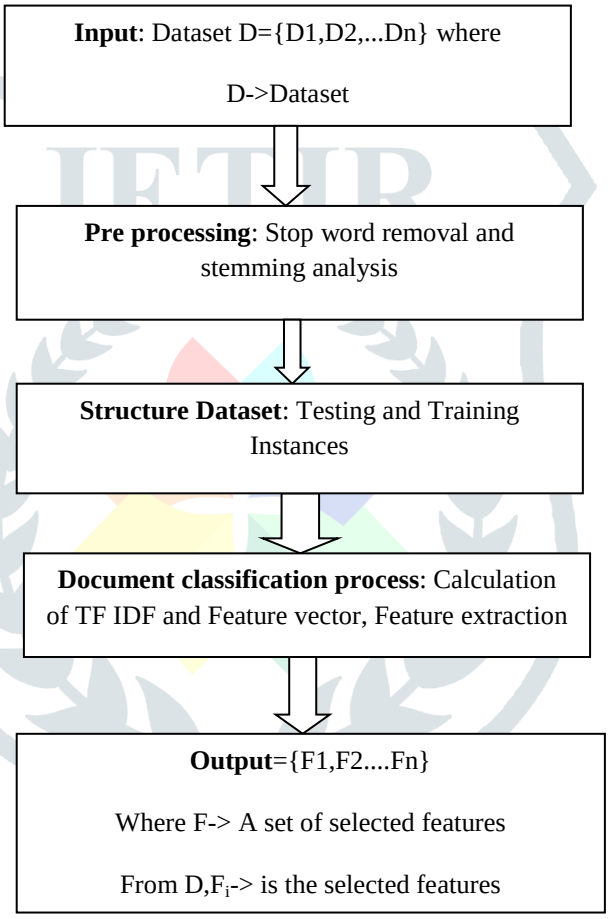
Append the terms to TF_{list}

}

tf-idf is tf-idf(t,d)=tf(t,d)×idf(t)
Idf(t)=log2 |D|{d:t£d}
Preparation of feature vector an append FV_{list}
For(i=0;i<v)
{
Perform feature vector creation (FV_{list})
}
Firstly the Reuter dataset is loaded, followed by tokenizing. In this progression each word in the report is part into tokens, and afterward put away it as Tlist. After that, prepare the stop word list and afterward stored it as SWlist.Apply the for loop condition to check the stop words. If the stored word is available in the document, at that point erase the word from the document. After that stage the stop word removal, do stemming in stemming process apply null stemmer to change the words into their root arrangement. Then stored it as R list. At that point both TF and IDF utilizing tf-idf is tf-idf(t,d)=tf(t,d)×idf(t),Idf(t)=log2|D|{d:t£d}.Based on the term frequency ,form the attribute vector of the document followed by accumulate it as FV list.

Feature extraction process: Feature extraction is a particular appearance of dimensionality reduction. Extract features from the document during the estimate of their weights in Different related domains. The feature extraction is a pre-processing phase of the knowledge discovery. This pre-processing step aim at change the free-text review sentences into a set of words and, at the same time, elevating their semantic importance.

Below flow chart represent the steps for document classification,



IV.EXPERIMENTAL RESULTS

This section give a performance study on Reuters dataset(shown in Table 1). Reuters dataset is a text categorization dataset, Reuters dataset is taken for extracting the features. After extracting the process the objects are split into two subsets: one for training (or development) and the other one for testing (or for performance evaluation).

S.no	Dataset	Document	Classes
1	ReutersCorn	1553	2

Table 1.details of dataset

Experimental procedure: following steps are performed for conduction of this experiment
1) Firstly, we can upload the Reuters dataset.
2) Remove the stop words from the dataset (i.e is, the, on ...etc)
3) Do stemming word analysis for removing the common meaning and flexional endings from the words.
4) Compute the term frequency(TF) and inverse document frequency (IDF).
5) remove features based on the TF and IDF.
6) Calculate the accuracy for the extracted features using stemming algorithm.
In order to justify the performance of proposed method the test is conducted using Reuters dataset and that results are tabulated in Table2.

Methods	Extracted number of features	Accuracy	Response time
TF-IDF with stemmer	1182	98.3%	3.26 sec
Null stemmer	2022	92%	4.21 sec

Table 2.Feature extraction method against no. of features, accuracy and runtime of Reuters dataset

Accuracy: It refers to the nearness of a measured value to a standard or known value.In this experimentation, we first utilize tf and idf stemmer and null stemmer, tf-idf stemmer is an computerized process which produces a base string in an attempt to represent a correlated words, null stemmer means null or nil means no value or not applicable it means "no explicit value". Null stemmer gives the overfilling meanings, it is little difficult sometimes encounter problems related to confusion about the meaning of null. All the 50 queries graph are shown in Figure 2 The input factors are fixed as 98.3 and 92. The accuracy results are shown in Figure 2.

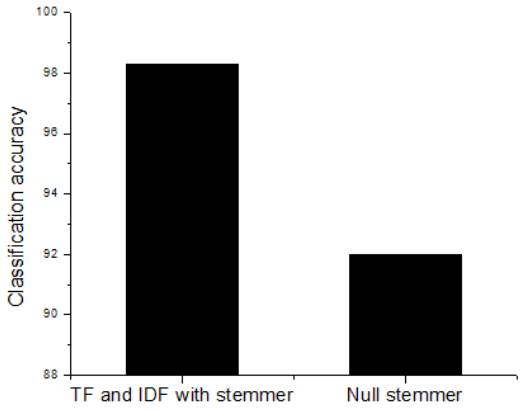


FIGURE 2:classification of accuracy for the Reuters dataset

Efficiency: Efficiency is defined as the capability to create a time with smallest amount of effort. Response time as the metric to assess the efficiency of different algorithms. In this experimentation using Reuters dataset. Given a set of queries, the response time is describe as the normal amount of time that use in processing queries. Figure 3 shows the Response time for searching the queries result. In this experiment, input parameters are fixed as tf and idf stemmer is 3.26 seconds and null stemmer is 4.21 seconds.

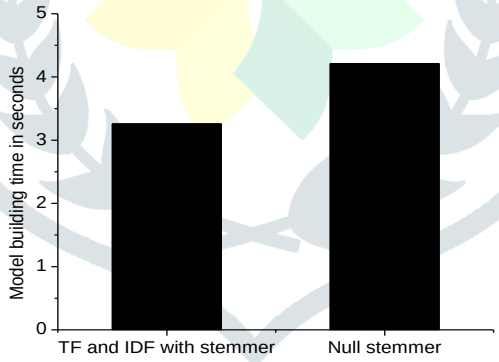


Figure 3: Response time of model building for the reuters dataset

CONCLUSION

In this paper we proposed solutions to the assigns weights to the keyword of points by using term frequency and inverse document frequency . Then, each group of points can be score based on distance between points and weights of keywords. The results clearly indicate that suggest algorithm provides a considerably better performance in terms of accurateness, dimension reduction rate and response time. In future we expect to expand the examination of Random projection and Multi scale hashing to disk.

REFERENCES

[1] VishwakarmaSingh,B.Zong,AmbujK.Singh,"Nearest Keyword Set Search in Multi-Dimensional Datasets",vol.28,no.3,March 2016.

[2] C. Long, R. C.-W. Wong, K. Wang, A. W.-C. Fu, "Collective spatial keyword queries: A distance owner-driven approach," in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2013, pp. 689–700.

[3] Z.Li, H.Xu, Y.Lu and A.Qian,"aggregate nearest keyword search in spatial databases, "in Proc .12th Int.Asia Pacific Web Conf., 2010, pp.15-21.

[4] A.Guttman,"R-Trees:A dynamic index structure for spatial searching, "in Proc.ACM SIGMOD Int.Conf. Manage. Data Eng.,1984,pp.47-57.

[5] X.Cao,G.Cong,C.S.JensenandB.C.Ooi,"Collective spatial keyword querying, "in proc.ACM SIGMOD Int.Conf.Manage.Data, 2011, pp.373-384.

[6] M.Datar,N.Immorilica,P.Indyk,andV.S.Mikkorkni,"Locality sensitive hashing scheme based on p-stabledistributions,"inProc.20th Annu.Symp.Comput.Geometry, 2004,pp.253-262.

[7] Y.Zhou, X.Xie, C.Wang, Y.Gong, and W.-Y.ma,"Hybrid Index structure for location-based websearch,"inproc.14th ACMInt.Conf.Inf.Knowl.Manage. 2005, pp.155-162.

[8] R.Hariharan,B.Hore,G.Li,andS.Mehrotra,"processing spatial keyword queries in geographic information retrieval(GIR)system," in proc.19th int.Conf.Sci.StatisticalDatabaseManage.,2007,p.16.

[9] S.Vaid,C.B.Jones,H.Joho,andM.Sanderson,"Spatio-textual indexing for geographical search on the web," in Proc.9th Int.Conf.Adv.Spatial Temporal Databases, 2005, pp.218-235.

[10] I.DeFelipe, V.Hristidis, andN.Rishe,"Keyword search on spatial databases, "in Proc.IEEE 24th Int.Conf.Data Eng., 2008, pp.656-665.