International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 – 5161

Available at http://inpressco.com/category/ijcet

Research Article

Efficient Ranked Multi-Keyword Search using Machine Learning Algorithms

Namdeo S. Kedare and Chaitanya Mankar

Dept of Computer Engineering Dhole Patil College of Engineering Pune, India

Received 10 Nov 2020, Accepted 10 Dec 2020, Available online 01 Feb 2021, **Special Issue-8 (Feb 2021)**

Abstract

The increasing amount of documents in the search index of information retrieval system make the problem of ranking documents difficult. The updated state of the problem leads to the point where machine learning becomes the most effective way to optimize the ranking function. Keyword searching is an effective method to retrieve information from such important networks. The aim of keyword search is to find the answers covering all or part of the queried keywords. A challenge in keyword search systems is to ranking the answers according to their importance. This importance lies in the textual content and structural compactness of the answers. Classification is the process of groping the text documents based on phrase, word, & combination of them with respect to set of predefined categories. Data classification has many different applications such as mail routing, email filtering, content classification, news monitoring and narrow-casting. Keywords are retrieved from documents to classify the documents. Keywords are subpart of words that contain the most important information about the content of the document. Keyword retrieval is a process used to take out the important keywords from documents. With the help of proposed system keywords are extracted from documents using TF-IDF and Naïve Baye's algorithm. TF-IDF algorithm is used to select the different words. The words which have high similarity are taken as keywords. The research has been done using Naive Baye's algorithms and their performance is analyzed with help of on machine learning. This system uses keyword with top-k ranked search over secure server data. The system provides the accurate result ranking documents & search efficiency due to the use of tree based index and efficient search algorithm.

Keywords: *Keyword based search, machine learning, naïve bayes algorithm, TF-IDF algorithm, Ranking, classification.*

Introduction

During last decade, the number of digital documents available for various purposes has grown enormously with the increasing availability of high capacity storage hardware and powerful computing platforms. The realistic increase of documents requirement effectual organizing and retrieval methods mainly for large documents. Text classification is one of the important techniques in text mining to categorize the documents in a supervised manner. The processing of text classification involves two main problems they are the extraction of feature terms that become effective keywords in the training phase and then the actual classification of the document with the help of these feature terms in the test phase. Text classification can be used for document routing & filtering to topic specific processing mechanisms such as information extraction and machine translation. Various methods are used for document classification such as Naive Baye's, Support Vector Machine, K-Nearest Neighbor, Fuzzy C-means, Neural Networks, Decision trees and Rule based learning algorithms out sourcing.

Literature Survey

A. Ghanbarpour & H. Naderi [1] explains an attribute-specific ranking method based on language models to rank the different candidates answer according to their semantic information up to the important attribute level. This method answers using a powerful model enriched with attribute-specific sequences and integrating both the structure and content of answers. The proposed model structure is directly estimated on the sub-graphs (answers) and is defined such that it can preserve the importance of keywords in nodes.
Karl Severin, Swapna S. Gokhale Aldo Dagnino. [2] suggested a schemes supporting efficient ranked keyword search for achieving effective utilization of remotely stored encrypted data. In this structure, they used a feasible once-over to in addition enhance the intrigue suitability, and get the ostensibly debilitated

constrain framework to cover get the chance to instance of the clients demand. Security examination shows that our course of action can accomplish gathering of different records and reports, trapdoor confirmation, trapdoor un-likability, and hiding access instance of the intrigue client.

Vidhya. K. A, G. Aghila [3] showed a safe multi-catchphrase arranged look design over encoded cloud information, which meanwhile underpins dynamic fortify operations like destruction and development of reports. Naive Baye's works better for the data characteristics with certain deterministic or almost deterministic dependencies that are low entropy distribution, however the fact is that algorithm work very well even when the independence assumption is violated.

Pawar Supriya, Dr. S. A. Ubale [4] proposed a useful and gainful multi-catchphrase break with word ask for over blended cloud information by gaining best k scored records. The vector space model and TF-IDF demonstrate are utilized to gather record and question time. The KNN algorithms calculation used to scramble record and demand vectors and develop a unique tree called Balanced M-way Search Tree asking for and propose a Depth First Search Technique figuring to complete the reasonable multi-catchphrase proportionally suitable word arranged for search. The precision & effectiveness of Depth First Search Technique estimation are addressed with a case, BMS tree, it takes sub-straight time multifaceted nature.

Alexander Ratner et al. [5] has proposed the paradigm for the program aticcreation of training sets called data programming in which users express weak supervision strategy, which are programs that label subsets of the data, but that are noisy and may conflict, which gives high quality with the increasing availability of powerful computing platforms & high capacity storage hardware. The vivid increase of documents demands effectual organizing and retrieval methods mainly for big documents. Text classification is one of the important techniques in text mining to categorize the documents in a supervised manner. The processing of text classification considers two main problems that are the extraction of feature terms which become effective keywords in the training phase and then the actual classification of the document using this features terms in the test phase.

Pinal shah et al. [6] has proposed web search personalization with the help of semantic data. The current search engines retrieve results are sometimes not of user relevance due to keyword based search, so to fill the gap between the user interest and retrieved search results, personalized web search needs to be evolved. For example, a programmer and a biologist may use the same query "mouse" with different search context, but the search systems would return same results.

H. Han, D. Zhu, and X. Wang. [7] Has showed Semi-supervised text classification from unlabeled documents using class associated words. A learning algorithm, based on the combination of Expectation-Maximization (EM) and a Naive Baye's classifier, is introduced to classify documents from fully unlabeled documents using class associated words. Experimental results show that it has good classification capability with high accuracy, especially for those categories with small quantities of samples.

Luis A. Trindade, Hui Wang, et al. [8] gives the classification of text using word sequence kernel methods. They evaluate the two kernels for the task of text classification, in particular to see how their difference affects their performance. The kernels are employed as similarity functions in an algorithm which is then tested on two text classification tasks: the determination of whether sentences in a corpus are correctly formed (word permutation task) and the determination of whether sentences in another corpus are subjective or objective (subjectivity classification task). We use classification accuracy on test data set as a measure of their performance.

Ranjitha K. V. & Venkatesh [9] they show classification and optimization scheme for text data using naïve baye's algorithm. This is compelling machine learning method which uses Naïve Bayes classifier for text data. In Machine Learning approach, the classifier is built automatically by learning the properties of categories from a set of pre-defined training data. Hence, it can process complex furthermore, multi assortment information in dynamic situations. Naïve bayes classifier which scales directly with number of indicators and data points which can be used for both binary and multiclass classification problems. Wei Zhang, Feng Gao [10] they shows the importance of Naive Bayes classifier which is widely used in machine learning for its simplicity and efficiency. However, most of the existing work on naïve Bayes focused on improving the Bayes model itself or whether the "naïve assumption" is satisfied. The performance of naïve bayes in text classification is analyzed and the corresponding results from different points of view is proposed, then an improving way for text classification with highly asymmetric misclassification costs is provided.

Text classification is useful for document filtering and routing to topic specific processing mechanisms such as machine translation & information extraction. Various methods are used for document classification such as Naive Baye's, Support Vector Machine, K-Nearest Neighbor, Fuzzy C-means, Neural Networks, Decision trees and Rule based learning algorithms out sourcing.

Proposed Methodology

The proposed system consists of user interface, pre-processing, feature classification using Naïve Baye's classifier & frequency calculation by TF IDF algorithm for more accurate categorization of words. This system useful to solve irrelevant data and more accuracy by associating TF-IDF with Naïve Baye's Classification algorithm.

Naive Baye's (NB):
Naive Baye's algorithm uses Baye's Theorem, which finds the probability of an event given the probability of another event that has already occurred. Naive Baye's algorithm performs very well for a problem which is linearly separable and even for problems which are non-linearly separable it performs reasonably well.

TF-IDF Algorithm:
TF-IDF stands for Term Frequency Inverse Document Frequency. The TF-IDF weight is often used in information retrieval & text mining. TF-IDF Variation weighting scheme are often used by search engines to scoring and ranking a document's relevance. This weight is important and is a statistical measure used to evaluate importance of a word to a document in a collection or corpus of data.

A. System Architecture
Basically Proposed System is consists of 3 modules

1.User
This module helps clients to enter their query keyword to get the most important documents from set of the uploaded documents. This module recovers the documents from cloud storage which coordinates the query keyword.
2.Data Owner
After expansion of keywords the data owner assist data with multiple keywords the document utilizing based on machine learning Algorithm and after that upload the document to store the database.

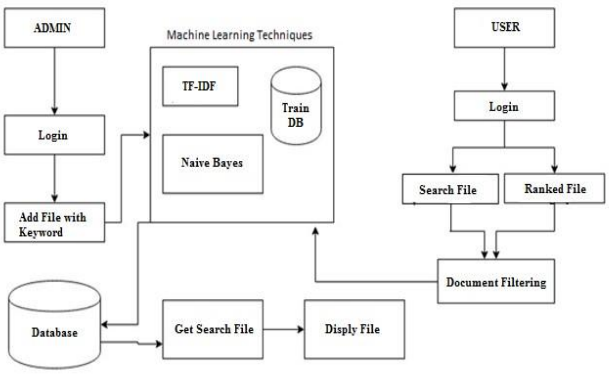

Fig I. Architecture of System

Ranked Results

Clients/user can download the resultant arrangement of documents just if he/she is approved client who has allowed consent from data owner to download specific document. Here user can get the ranked and mostly search records from the ranking system to get exactly data to the all user.

B. Algorithms

1. Naive Baye's Algorithm
Naive Baye's algorithm calculates the probability of an event in the following 4 steps:
Step 1: Calculate the prior probability for given class label. Step 2: Find related (Likelihood) probability with each attribute for each class.
Step 3: Put these value in Baye's Formula and calculate posterior probability.
Step 4: Calculate which class has a highest probability, given the input belongs to the higher probability class.

To know the Naïve Baye's classifier, complete survey is required in which the application of various Machine Learning
Techniques to the text categorization problem like in the field of e-mail filtering, medicine, including rule learning for knowledge base systems has been explored. The survey is tends to the various probabilistic approach of Naïve Baye's ML approach for which the text categorization aims to classify the document with maximum accuracy.
Probability is checked for condition by Naïve Baye's classifier which comes from well known statistical system approach 'Baye's Theorem', where as Naïve refers to "assumption" that all different attributes of the examples are not depends on each other, given the context of the category. Because of the independent assumption the parameters for each attribute can be learned separately or differently and this greatly simplifies learning especially when the number of attributes is large.during this text classification, the probability of a document d belongs to class c is calculated by using Baye's theorem as follows.

$$P(c/d) = \frac{p(d/c)/p(c)}{p(d)}$$

Baye's Theorem provides a way which calculates the probability of a piece of data belonging to a given class, given our prior knowledge.
Baye's Theorem is stated as:
P (class | data) = (P (data | class) * P (class)) / P (data)
Where,
P (class | data) is the probability of class given the required data.

2. TF-IDF
Term frequency-inverse document frequency (TF-IDF) is a statistical measure that evaluates how relevant a word is to a document from collection of documents. This is done by multiplying two metrics: how many times a word appears in a document and the inverse document frequency of the word across a set of documents.
It has many uses, most important in automate text analysis, and is very useful for scoring words in machine learning algorithms for Natural Language Processing.
TF-IDF was searched (invented) for document search and information retrieval. It works effectively by increasing proportionality to the number of times a

word appears in a document, but it is offset by the number of documents that contain the word. So, words that are common in every document, & if rank low even though they may appear many times, since they don't mean much to that document in particular.
However, if the word 'hi' appears many times in a document, while not appearing many times in others, it probably means that it's very relevant. For example, if what we're doing is trying to find out which topics some NPS responses belong to, the word 'hi' would probably end up being tied to the topic Reliability, since most responses containing that word would be about that particular topic.
TF-IDF for a word in a document is calculated by multiplying two different matrices values. The term frequency (TF) of word in a document. There are several ways of calculating this frequency, with the simplest being a preliminary count of instances a word appears in a document. There are ways to adjust the frequency, by length of a document, or by the raw frequency of the most frequently appear word in a document. The inverse document frequency (IDF) of the word across a set of documents. This means that, how common or rare a word is in the entire document set. The closer it is to 0, then more common a word is. This metric can be calculated by taking the total number of documents, dividing it by the number of documents that contain a word, & calculating the logarithm. So, when the word is very common and appears in many documents, this number will approach 0. Otherwise, it will approaches to 1.
When multiplying these two numbers those results into the TFIDF score of a word in a document. The higher the score, then more relevant that word is in that particular document.
By puttting it in more formal mathematical terms, the TF-IDF score for a word t in the document d from the document set D then it is calculated as follows:

Tf*idf [t,d,D] = tf[t, d] * idf[t, D] Where:
T f [t, d] = log [1 + freq (t, d)]
idf [t, D] = log[N/((count(d €D: t € d)]

Machine learning with natural language is faced with one major hurdle – its algorithms usually deal with numbers, and natural language is, well, text. So we need to transform that text into numbers, otherwise known as text vectorization. It's a fundamental step in the process of machine learning for analyzing text, and different vectorization algorithms will drastically affect end results, so you need to choose one that will deliver the results you're hoping for. Once you've transformed words into numbers, in a way that's machine learning algorithms can understand, the TF-IDF output values can be fed to algorithms such as Naive Baye's greatly improving the results of more basic methods like word count. Simply put, a word vector represents a document as a list of numbers, with one for each possible word of the corpus. Vectorizing a document takes the text and creating one of these classes, and the numbers of the vectors somehow represent the content of the text. TF-IDF enables us to gives us a way to associate each word in a document with a number that represents how relevant each word is in that document. Then, documents with similar & relevant words will have similar vectors, which is what we are searching for in a machine learning algorithm.
It's important to use TF-IDF for text extraction so that you can gain a better understanding of how machine learning algorithms function. Machine learning classifier algorithms traditionally work better with numbers; TF-IDF algorithms help them to decipher words by allocating them a numerical value or vector. This has been revolutionary for machine learning, especially in fields related to NLP such as text analysis.
In text analysis with machine learning, TF-IDF algorithm is helpful to sort data into categories, as well as extract keywords. This means that simple, monotonous tasks, like inputting data, tagging support tickets or rows of feedback can be done in seconds.
Every wondered how Google can serve up information related to your search in mere seconds? Vectorization transforms text within documents into numbers, so TF-IDF algorithms can rank articles in order of their relevance.

Result and Discussions

Naïve Baye's generally outperforms for large datasets in text classification problem in spite of the Naïve independence assumption but as of small data sets Naïve Baye's doesn't show promising results in accuracy or performance. [11] Even though Naïve Baye's classifier achieves better accuracy, to fine tune the classification accuracy combined with the other machine learning technique like SVM, neural networks, decision trees for unsupervised data.
The existing system is based on graph structure so it is less efficient and effective. To overcome this problem, keyword search is an effective method to retrieve information from bundle of data with tree structure. This system uses naïve bayes & TFIDF algorithms to provide accurate results. The aim of keyword search is to find a set of answers covering all or part of the queried keywords.

Table I - Efficient Ranked Multi-Keyword Search System.

Sr No	Keyword	Count	Efficiency
1	Hi	4	100
2	Hello	3	75
3	…	….	….

Basically Naïve Baye's work on conditional probability derived from the idea of Baye's theorem which is modified according to the application of Naïve Baye's for text classification. To evaluate the text classifier system with the Naïve Baye's approach there are two metrics factor, recall, precision and F1measure can be used to find the effectiveness of document classifier

which is given by, tp (True Positive): The number of documents that are classified correctly to that class. tn (True Negative): The number of documents that are correctly rejected from that class. fp (False Positive): The number of documents rejected incorrectly from that class.
fn (False Negative): The number of documents classified incorrectly to that class. P: Precision= tp/(tp+fp)
R: Recall= tp/(tp+fn)
F1 (Measure) = 2(P*R)/(P+R)

The formulas for precision, recall and F-measure is given in the performance of Naïve Baye's Machine learning technique when combined with the other method shows better performance. The discussion about the Naïve Baye's classifier performance with the micro F1-measure the values for multinomial methods available from the paper [6] such that the variants of the classifiers significantly outperfor the old multinomial Naive Baye's at least when the 20 - Newsgroup is used. In the graph showing the micro_F1 values, SRF_l at of 0.2 achieves the best performance. The RF_u & SRF_u gives better performance than baseline performance & less than the Rf_l or SRF_l, but trivial. It means that there is no significant difference between using the number of tokens and the number of unique terms in the document.
The big difference between the micro_F1 & macro_F1 is that the performance increase by the normalization over the baseline is much greater in the case of macro_F1 (0.2238 for the baseline versus 0.5066 for RF-l). Macro_F1 values in the Reuters 21578 collection tend to be dominated by a large number of small categories, which have a small number of training documents [6], From the above survey it is understood that the proposed normalization methods are quite effective, particularly in the categories where the number of positive training documents is small where the traditional Naïve Baye's algorithm fails, after doing subsequent experiments and found that this method is quite effective. While categorizing text there are various benchmarking datasets available like Cora, Reuters 21578, 20 Newsgroup and Web KB. Reuters 21578 and 20 Newsgroup datasets are designed with either set of long or short documents. There are predefined categories where the hierarchy structure of each category is specified. The dataset Web KB is basically preferred for spam mail classification simulation. However the results of the Naïve Baye's along with the other hybrid methods for text document classification with these datasets and feature selection technique is shows in the following Table1. Performance of Naïve Baye's classifier when combined with other methods. Following table shows Document Classification and Naïve Bayes Machine Learning Approach.

Table II - Document Classification and Naïve Bayes Machine Learning Approach.

Naïve Bayes Model with Noun Phrase approach	User defined Feature selection template	Training material comes from four sections (15-18) of the Wall Street Journal (WSJ) part of the Penn Treebank- II corpus.	93.7%
Naïve Bayes with Probability estimation Tree	Small Size Data –No Feature Selection Required	Experiments on 9 UCI Datasets are conducted	On Average 87%
Naï ve Bayes with Support Vector Machine	TF-IDF (Term Frequency and Inverse Term Frequency Method)	20 Newgroup & Prepared own Dataset for testing.	Flat Ranking – 88.89% Flat Ranking with High Ranking Keyword – 90.00%
Naïve Bayes with Active Learning Boosting Method	Weightage Scheme	6 Datasets from the UCI Machine Learning Repository	Achieved Higher Accuracy by 0.05% compared to Adaboost
Naïve Bayes with Generative/Discriminative Technique	Wavelet transformation Feature subset of Documents	Reuters21578, Cora, WebKB and 20Newsgroup Dataset	92.5% on Average
Naïve Bayes for Learning Object Identification	Weightage Scheme, Normalized Statistics	Set of own data files	Good Learning Object Identification is achieved.
Naïve Bayes for E-mail Spam Filtering	Mutual Information Gain	Lingspam corpus and PUI corpus	Multivariate – 98.86% Accuracy ,Multinomial- 98.06%
Naïve Bayes with Multivariate and Multinomial Distribution	New Feature Weightage scheme was proposed and tested	Reuters21578 and 20Newsgroup	F1-Measure is compared for arious weightage scheme. 0.5066 Multinomial- 0.2238

TF-IDF

TF*IDF classifier is very important information retrieval technique that weighs a term's frequency (TF) and its inverse document frequency (IDF). Each word or term has its respective TF and IDF value. The product of the TF and IDF values of a term is called the TF*IDF weight of that term. This term weighting suggests that the valuable terms will have low frequency on the outside and a high frequency in particular document. Two main components of the term weight must be distinguished: term frequency (tf) factor and inverse document frequency (idf). The term frequency (tf) is the number of times a term appears in document.

$TF(ij) = F(ij) / L(i)$

Where F(ij) is the frequency of term j in document i, and L(i) is total number of keywords in the document i.

There are various weighting schemes to discriminate one document from others. This factor is called inverse document frequency IDF.

$IDF(i) = \log(n/n)+1 \qquad n(j) > 0$

Where n is the number of documents and n(j) is the number of documents in which term j. The concepts of term frequency and inverse document frequency (tf-idf) are combined, to produce a composite weight for each term in each document.

TF – IDF =TF*IDF

Conclusion

This system is designed keyword with top-k ranked search over secure server data. The system provides the accurate result ranking documents & search efficiency due to the use of tree based index and efficient search algorithm. For future work there are many challenges in symmetric searchable encryption scheme. As it is assumed that all the data users are trustworthy, but in practical, the dishonest data user may distribute his secure keys to unauthorized users

References

[1]. A. Ghanbarpour, H. Naderi, IEEE Transactions On Knowledge & Data Engineering, 2018. "An Attribute Specific Ranking Method Based on Language Models for Keyword Search over Graphs."

[2]. Karl Severin, Swapna S. Gokhale Aldo Dagnino. 2019 IEEE43rd Annual Computer Software and Applications conference (COMPSAC), pp: 978-1-7281-2607-4. "Keyword Based Semi-Supervised Text Classification."

[3]. Vidhya.K.A, G.Aghila (IJCSIS) International Journal of Computer Science and Information Security, Vol. 7, No. 2, 2010. "A Survey of Naïve Bayes Machine Learnin approach in Text Document Classification."

[4]. Pawar Supriya, Dr. S. A. Ubale.International Journal for Research in Applied Science & Engineering Technology (IJRASET) Volume 5 Issue VII, July 2017. "Multi Keyword Top-K Ranked Search over Encrypted Cloud."

[5]. Alexander Ratner, Sen Wu, Christopher De Sa, DanielnSelsam, Christopher RéStanford University, Pages 3567–3575, 2016. "Data Programming: Creating Large Training Sets, Quickly."

[6]. Pinal shah, Jay patel, kamlesh makhawana, parth shah "Review on web search personalization through semantic data."

[7]. H. Han, D. Zhu, and X. Wang, "Semi-supervised text classification from unlabeled documents using class associated words."

[8]. Luis A. Trindade, Hui Wang, William Blackburn, Niall Rooney, "Text classification using word sequence kernel methods."

[9]. Ranjitha K. V. & Venkatesh, "Classification and optimization scheme for text data using machine learning naïve bayes classifier."

[10]. Wei Zhang, Feng Gao, "Performance Analysis and Improvement of Naïve Bayes in Text Classification Application".

[11]. Sang-Bum kim, Kyong-soo Han, Hae-Chang Rim, Sung Hyon Myaeng "Some Effective techniques for Naïve Bayes Text Classification" IEEE Transactions on Knowledge and Data Engineering -2006.

[12]. Siham JABRI, Azzeddine DAHBI, Taoufiq GADI, Abdelhak BASSIR, "Ranking of Text Documents using TF-IDF Weighting and Association Rules mining."

[13]. Xinggao Cai & Shujin Cao, "A keyword extraction method based on learning to rank."

[14]. S.V. Semenikhin, L.A. Denisova "Learning To Rank Based on Modified Genetic Algorithm." Tian Weixin & Zhu Fuxi "Learning to Rank UsingSemantic Features in Document Retrieval."