USING GOOGLE'S KEYWORD RELATION IN MULTI-DOMAIN DOCUMENT CLASSIFICATION

Ping-I Chen [1] and Shi-Jen Lin[2]

[1]Digital Education Institute, Institute for Information Industry, Taipei, Taiwan, ROC
be@iii.org.tw
[2]Department of Information Management,
National Central University, Chung-Li, Taiwan, ROC
sjlin@mgt.ncu.edu.tw

ABSTRACT

People can collect all kinds of knowledge from search engines to improve the quality of decision making, and use document classification systems to manage the knowledge repository. Document classification systems always need to construct a keyword vector, which always contains thousands of words, to represent the knowledge domain. Thus, the computation complexity of the classification algorithm is very high. Also, users need to download all the documents before extracting the keywords and classifying the documents. In our previous work, we described a new algorithm called "Word AdHoc Network" (WANET) and used it to extract the most important sequences of keywords for each document. In this paper, we adapt the WANET system to make it more precise. We will also use a new similarity measurement algorithm, called "Google Purity," to calculate the similarity between the extracted keyword sequences to classify similar documents together. By using this system, we can easily classify the information in different knowledge domains at the same time, and all the executions are without any pre-established keyword repository. Our experiments show that the classification results are very accurate and useful. This new system can improve the efficiency of document classification and make it more usable in Web-based information management.

KEYWORDS

Similarity distance, document classification, information retrieval, keyword sequence

1. INTRODUCTION

The search engine provides a brand new kind of information collection method to the company and to us. We can stay in the office and open a browser to search for any information we want to know in a few milliseconds. Thus, we do not need to send employees all over the world to collect documents or pictures for a company's decision making. But the new problem is how to deal with such a great amount of information and use it in an efficient way. A document classification system provides a useful way to automatically classify related documents into the same file. Thus, we can select a file in which we are interested and read highly relative information in it.

There are three main steps of document classification: keyword extraction, vector representation, and similarity measurement. Keyword extraction is about how to evaluate and find the potential keywords. Next, the system will use those extracted keywords to form the keyword vectors to represent the documents. Finally, we can use similarity measurement, such as cosine similarity or SVM, to evaluate how close two documents are and classify the similar ones. This three-step document classification has worked well in traditional information retrieval for several years. The only problem is that the keyword vector is always composed of at least thousands of words for

each knowledge domain. Thus, the computational cost is extremely high. Also, the vector will keep growing when there are new documents come in.

In Web information retrieval, the information can be collected from all kinds of resources, like webpages or web documents, and from different knowledge domains. For example, if a company wants to know if oil prices will increase or decrease, it will collect information about politics, the economy, and new technology. These three affecting factors have totally different representative keywords. If we use the traditional vector-based classification algorithms, the keyword vectors will become huge.

Individual users, nearly always read an article on a webpage and press the download bottom if the article seems worth saving. Using traditional classification algorithms requires users to download all of the documents and then use the system to classify similar documents into the same file. Users cannot download the documents and classify them to a repository at the same time because downloading too many documents which are in the same domain might contain only a few keywords which are the same, thwarting the traditional vector-based model.

In our previous work (Chen and Lin, 2011), we proposed a new algorithm, called "Google Core Distance" (GCD), to measure the relations between each two keywords using the number of search results of Google search engines. We also used the famous PageRank algorithm and combined it with the BB's graph-based clustering algorithm to find the keyword sequence which can be used to represent the keyword vector of each document. This idea is based on the sensor network's routing algorithm, and we have named it "Word AdHoc Network" (WANET). The advantage of the WANET system is that each document vector will have no more than four keywords. The experiment results showed that the 4-gram sequence of the determined keywords can identify the most relevant search results so that the representation of the keyword sequence is sufficient.

In this paper, we adapted the WANET system slightly so that the classification results will be better than before. We also proposed a new similarity measurement algorithm, called "Google Purity," to measure the similarity between the 4-gram extracted sequences to automatically classify the documents. Why did we not use the traditional similarity measurement algorithm? One reason is that the keyword sequences extracted by WANET rarely have the same keyword. Another is that we do not want to use long sequence vectors to represent the knowledge domain, to save computational costs. Therefore, if we use the traditional algorithms, there will be no classification results.

By using the Google Purity algorithm, users can download the documents and classify them immediately. The process only requires the user to assign some files, which represent a knowledge domain or specific classification, and choose a representative document for each file as the base sequence. The system will use the WANET algorithm to extract the 4-gram sequences to represent their vectors. While the user is reading a webpage and downloading the documents, WANET will also produce a 4-gram sequence and use the Google Purity algorithm to compare with each representative base sequence. In this way, WANET can easily decide which domain is the most similar and allocate the document into it. In most situation, our process can achieve high accuracy by using only one document, if its extracted sequence is representative.

The main contribution of this article can be concluded as follows:

- Smallest vector representation of documents
- Cross-domain ability
- Keyword extraction without pre-established keyword database
- Document classification with shortest vector

We will introduce our system and algorithm design in detail, and use the experiments to evaluate the accuracy of the system. The experimental results show that this system can achieve high accuracy in most situations. Also, if we add a new domain to the repository, the accuracy of the previous established domains will maintain high accuracy and stability. We think this kind of system not only can help companies collect and classify information more efficiently, but also can let individual users manage their own repository and knowledge in a more convient way.

2. RELATED WORKS

In this section, we will introduce some relative techniques which can help us classify documents automatically.

2.1. Word Sequences

As we know, the Google search engine can recognize the sequential order of keywords. Thus, if several keywords are entered into the search engine in different sequences, the search results for each will greatly differ. Also, if one wants to compare the similarities between webpages and documents, algorithms can be used to extract the keyword sequence and similarity measure methods can be used to cluster related information, as shown in Fig. 1.

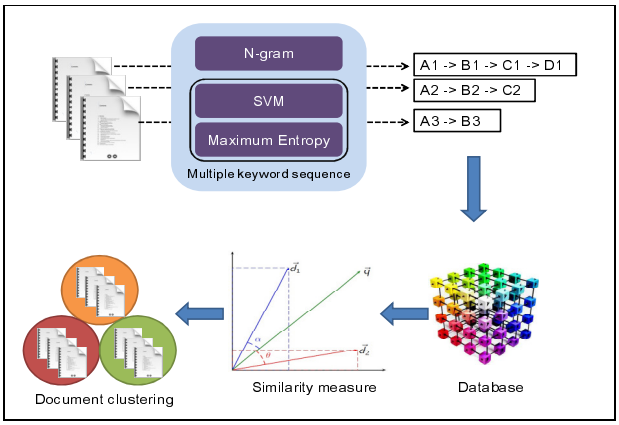


Figure 1. Keyword sequence and document clustering.

2.1.1. Multiple keyword sequence extraction

Sato and Saito (2002) proposed a method of extracting word sequences by using a support vector machine (SVM). They used this method to find relationships in documents. Its original design was intended to obtain the least-biased statistical inference when there was insufficient information, but it can also be used to extract bilingual word sequences. Feng and Croft (2001) used the maximum entropy (ME) algorithm to extract English noun phrases automatically. They used probability methods to find a sequence of keywords from one word to the next and also used it to realize article summarization. Li et al. (2003) used the same idea to extract a sequence of keywords from the news and clustered those related pieces of information to provide the user with a summary of search results. But they also said that the ME model required a large training data to evaluate the feature parameters.

2.2. Document Clustering and Classification

Document clustering and classification consists of three main stages: (a) extracting the keywords from documents; (b) extracting the sequence of keyword strings and transforming them into suitable data structures; and (c) using a similarity measurement algorithm to group similar

representations. Many articles focus on this research domain, and the methods used are all based on those we discussed in the previous section to extract keyword sequences. The following three methods are used to cluster similar documents in the same group: the vector space model, frequent itemset, and K-means and agglomorative. Also, by using the document clustering algorithm, we can determine the users' reading or browsing behaviors and then use this information to customize the search results that are most relevant to the user.

2.2.1. Vector space model

Salton et al. (1975) proposed the vector space model for automatic indexing. They thought that the document space will comprise one or more index terms, and those terms can be weighted according to level of importance. If one obtains the index vectors for two documents, the similarity between them can be easily calculated, and those documents with the highest similarity scores can be clustered. Most researchers use a cosine measurement to compute the cosine of the angle between these two vectors.

The Jaccard similarity coefficient (Jaccard index) is a statistic used to compare the similarity and diversity of sample sets. It compares the sum weight of shared terms to the sum weight of terms that are present in either of the two documents but are not the shared terms. Luo et al. (2009) used the cosine and Jaccard similarity coefficient to measure similarity to coordinate with the K-means algorithm for document clustering. They found that the cosine performs better than the Jaccard index for document clustering.

Pearson's correlation coefficient can be used to measure the correlation between two objects on all attributes. Shardanand and Maes (1995) used the standard Pearson r correlation coefficient to measure similarity between user profiles. Yang et al. (2002) thought that strong coherence may exist only on a subset of dimensions so that sometimes the correlation value will not be very high. The accuracy of this algorithm will be affected by the data because it lacks the ability to deal with attribute bias.

2.2.2. Frequent itemset mining

Association rule mining is a kind of data mining technique that uses frequent itemsets to find potential associations between individual words and thus provide users with more meaningful concepts. A frequent itemset is a set of frequently occurring items whose probability of co-occurrence in the database will be higher than that of the threshold.

Edith et al. (2006) proposed a method that used the maximum frequent itemset sequence that was not a subsequence of any frequent sequence as the representative term vector of documents. They then used the k-means algorithm to cluster similar documents and group them together.

Fung et al. (2003) used frequent words and the hierarchical clustering algorithm to cluster related information. Thus, users can easily obtain information by using the meaningful cluster labels. They thought that a frequent itemset could describe something common to all documents in a cluster, and these itemsets could be used to construct clusters into a topic hierarchy to achieve high accuracy and meaningful results. Most of the previous research used the TF-IDF algorithm from which frequent words can be easily extracted. Therefore, each document can be represented by a vector of weighted frequencies. But the authors used global frequent items instead of the TF-IDF to construct the initial clusters. Finally, they used their score function to measure the goodness of the initial clusters and tried to disjoint them to obtain the cluster tree.

Li et al. (2008) used a frequent word sequence and k-mismatch for document clustering. The problem with using k-mismatch is to find all occurrences of patterns in a text with the most k mismatches and then use them to compare the similarity between each document's frequent

itemsets. They believed that the sequence of words provides more valid information than individual words and also represents the topics very well. Thus, using this method, greater accuracy can be achieved than that produced by traditional text clustering methods that use the vector space model.

2.2.3. K-means and agglomerative

K-means is the simplest method of cluster analysis that aims to partition the data into several clusters. The way it works is to define k centroids, one for each cluster, and then take each point belonging to a given dataset and associate it to the nearest centroid. The weakness of k-means is that it is not easy to determine the number of clusters, and this deeply affects the results of the classification. Another kind of clustering method is the hierarchical clustering algorithm. Most researchers use the agglomerative algorithm to cluster similar information. Agglomerative hierarchical clustering is a bottom-up clustering method that treats each document as a single cluster and then continues to calculate and merge pairs of clusters until all clusters have been merged into a single cluster that contains all related documents.

Steinbach et al. (2000) used these two algorithms to conduct a document clustering system and evaluate its accuracy. They used k-means, bisecting k-means, and hierarchical clustering and found that bisecting k-means achieve the highest accuracy. They found that any two documents may contain many of the same words. Thus, understanding how to distinguish between these documents and allocate them into various different classes, especially the nearest-neighbour documents, is a difficult job. The agglomerative method will often put documents of the same class into the same cluster. But the k-means algorithm uses a global property approach, which computes the average similarity of all the documents to clustering so that it can overcome the mixed nearest-neighbour problem. Otherwise, the bisecting k-means also has the same advantage of the k-means, and its performance and execution results are better than those of k-means.

Web taxonomy is becoming more and more important in our daily life because it allows us to gather all kinds of information from the Internet. The most important step in constructing a taxonomy is to extract the domain-specific terminologies and determine the relationships of each. Webpages always contain many new terms, so a traditional knowledge database that has been created manually is not suitable since the cost of collecting the data manually is excessive. Thus, a system is needed that can perform this function automatically. Zhang and Lee (2004) used the TSVM algorithm to deal with this problem. The transductive SVM (TSVM) algorithm, first introduced by Joachims, exploits its prior knowledge to speed up performance of the classification, especially for small training examples.

Godoy and Amandi (2006) produced a document clustering algorithm that used the unsupervised concept learning over Web documents to acquire user profiles. This algorithm can extract semantic relations from a Webpage so that they can be integrated into ontology and thus provide more detailed information when searching with a search engine.

We think that the NGD-based algorithm can also be used to cluster related documents. We have successfully used the NGD algorithm combined with the PLSA to find the meaningful keyword sequences in a document. But the problem is that the relationships between single knowledge domains will be less complex than those found in multi-domains. If one attempts to cluster documents that contain different kinds of knowledge, the results will not be as good. Possibly the NGD algorithm could be adapted to create one that is more accurate in finding absolute relationships between two keywords.

3. PROPOSED METHOD

In this section, we will introduce our system in detail, describing our design and calculation methods. Our system is composed of two main parts:

- Sequence Extraction algorithm
- Google Purity algorithm (GP)

Figure 2 shows our system architecture. First, the keyword sequence will be extracted from the document.

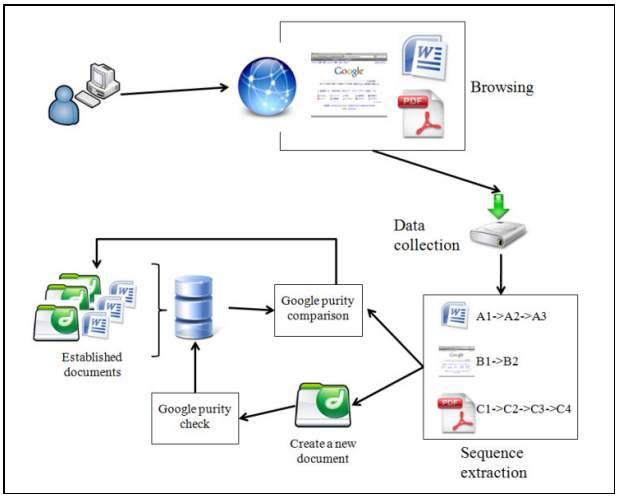


Figure 2. System flowchart.

3.1. Sequence extraction algorithm

In our previous work (Chen and Lin, 2011), we used the sequence extraction system, called WANET, to compute the word-by-word relations and find out the representative keyword sequence from the documents. This algorithm contains three main parts:

- Google core Distance (GCD)
- PageRank algorithm (PR)
- BB's graph-based clustering algorithm

We will introduce our previous work briefly for further introduction of our new classification system design.

3.1.1. Google Core Distance (GCD)

Cilibrasi et al. created the Google similarity distance algorithm in 2007. This algorithm is used to calculate the relationship between two words by using the keywords' number of search results in Google search engine.

In our previous work (Chen and Lin, 2010), we used the algorithm to automatically combine some single-word keywords into multi-word keywords to make the meaning of keywords more precise than single keywords. We will have a set of single keywords (S), and a set of multiple keywords (M) which contains one or more than one single keywords from S, where $S \subseteq M$, $aM \subseteq M$, $\forall a \in S$ and $aM = \{am | m \in M$,am is a sequence of two consecutive words$\}$. Thus, we can use each multiple keyword set to get its number of search results ($g(m)$) from Google to

calculate the relationship between each two multiple keywords by using the *GCD* algorithm. Fig. 3 presents a simple example of our idea. Suppose that circle "*a*" represents the total number of search results in Google. Circles "*b*" and "*c*" represent the number of search results for the two keywords that we want to measure. We can use the radius of circle a (*R*), subtracting the radius of circle b to get the value of *r1*. In the same way, we can find the value of *r2*. Thus, the distance of the two centers will be as follows:

$$GCD(m,n) \le \sqrt{(R-r_1)^2+(R-r_2)^2}$$
$$= \sqrt{\left(\sqrt{\frac{\log(google)}{\pi}}-\sqrt{\frac{\log(g(m))}{\pi}}\right)^2+\left(\sqrt{\frac{\log(google)}{\pi}}-\sqrt{\frac{\log(g(n))}{\pi}}\right)^2} \quad (1)$$

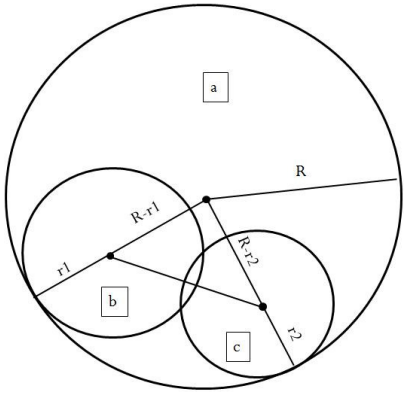


Figure 3. Distance-based Google similarity distance-without outlier.

The official data for the number of Web pages shows that about 8,058,044,651 pages are within the circle area of a. The number of search results for keywords "RFID-BASED" and "GENETIC ALGORITHM" are 535,000 and 3,000,000 respectively, and that number represents the circle areas of *b* and *c*. The number of search results is so huge that we will use their logs to calculate the results. Finally, the similarity score of these two keywords is about 0.55. The lower the value of the *GCD*, the better. This indicates that the two keywords always appear in the same knowledge domain or document. The calculation process is as follows:

$$\pi R^2 = \log(google) = \log 8058044651 = 9.906$$
$$\pi r_1^2 = \log(g(m)) = \log 535000 = 5.728 \quad \text{//RFID-BASED}$$
$$\pi r_2^2 = \log(g(n)) = \log 3000000 = 6.477 \quad \text{//GENETIC ALGORITHM}$$
$$\therefore R = 1.78, r_1 = 1.351, r_2 = 1.436$$
$$GCD(m,n) \le \sqrt{(1.78-1.351)^2+(1.78-1.436)^2} = 0.55$$

But sometimes, one of the *r1* or *r2* will become the hypotenuse of the triangle because the number of search results is too small. These keywords which have fewer search results are almost always specific keywords created by the user. Therefore, if we consider these outlier keywords which we eliminated before, and adjust the algorithm to let them join in the keyword sequences, we might be able to find stronger sequences to represent the document because those keywords are highly related to this article. Sometimes we can use only a few specific keywords to find the most relevant information. As we can see in Fig. 4. We suppose that the number of search results of c is very small.

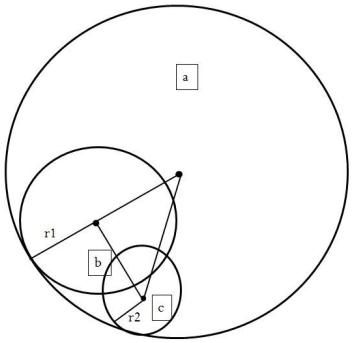


Figure 4. Distance-based Google similarity distance-with outlier.

In this way, when $(R-r_2)/(R-r_1) \geq 2$, the *GCD* algorithm will become as follows:

$$GCD(m,n) \leq \sqrt{(R-r_2)^2-(R-r_1)^2}$$
$$= \sqrt{\left(\sqrt{\frac{\log(google)}{\pi}}-\sqrt{\frac{\log(g(n))}{\pi}}\right)^2-\left(\sqrt{\frac{\log(google)}{\pi}}-\sqrt{\frac{\log(g(m))}{\pi}}\right)^2} \quad (2)$$

We named this function the adaptive GCD algorithm (AGCD) which can give those outlier keywords the opportunity to be joined in the keyword sequence to represent the documents.

3.1.2. PageRank algorithm

After we use the *GCD* to measure the similarity distance between each two keywords, we can imagine that those keywords in a document or webpage can form a fully-connected network structure. In our previous work (Chen and Lin, 2011), we used the threshold α, which represents the top 50% smallest *GCD* scores, to restrict and filter out some unimportant relations, and use them to extract the keyword sequence. In our experience, if we use 75% as the threshold, in some situations we can extract out longer n-gram sequences. But the accuracy of the experiment results will be almost the same. Also, the computational cost will increase. But here we will also consider whether using the top 50% largest *GCD* scores will improve the results or not. In other words, there are four totally different situations before we get to start to use the PageRank algorithm. We use table 1 as illustration and the keywords' relative distributions in the Google are shown in Fig. 5.

Table 1. Four kind of situations of the keyword relations.

	Without outlier	With outlier
Strongest keyword sequence	(a) GCD, ranking from small to large	(c) AGCD, ranking from small to large
Largest distribution sequence	(b) GCD, ranking from large to small	(d) AGCD, ranking from large to small

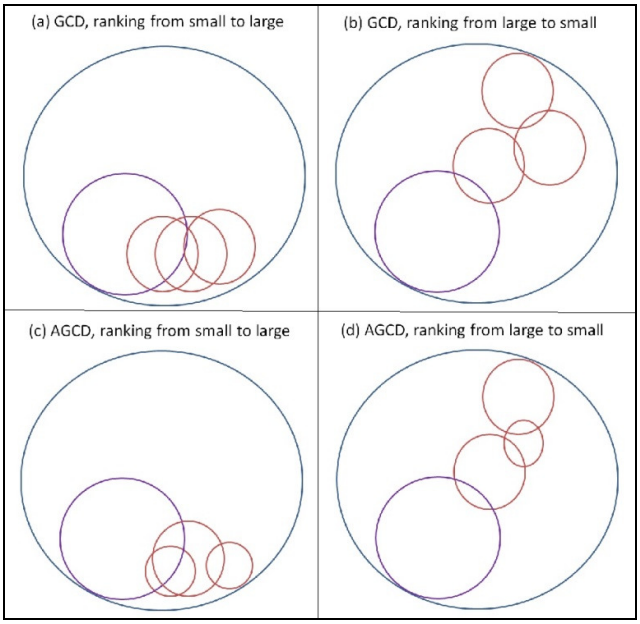


Figure 5. Distribution of the keyword sequence.

The situation (*a*) is the original algorithm which we used in our previous work (Chen and Lin, 2011). The idea is based on extracting the most frequent and highly relative sequences to represent the document. Thus, those keywords with a fewer number of search results will not be considered. The situation (*b*) used the GCD algorithm and uses the top 50% largest GCD scores. Thus, the distribution will not be as highly closely related as (*a*). But those keywords' coverage range in the whole space should be greater than original one. We think that if we adapt the ranking method, the keyword which we found will be more general and it might be able to represent each knowledge domain more easily, with better resulting accuracy. Situation (*c*) and (*d*) use the adaptive GCD algorithm to replace the GCD algorithm to take more consideration of the outlier keywords.

After we use the four different methods we introduced earlier, our next problem is how to extract a sequence of keywords from the WANET.

In order to find the keyword sequence, we first use the PageRank algorithm, which has been used to analyze the cross-relations of the Webpages for several years, to evaluate the linking status of the keywords. The one which has the highest PageRank score in the "keyword network" is the core of the network.

Here, we only let the PageRank algorithm execute the iteration once. The first reason is that it was original designed for calculating millions of the relationships between webpages. But in our WANET, we only have a few important keywords. If we use 100 iterations as the algorithm's original design, most of the *PR* score will be nearly so close that the difference will not be significant. The second reason is that the computation cost will be lessened by only executing a few times.

3.1.3. BB's Graph-Based Clustering algorithm

The BB's graph-based clustering algorithm (Beeferman and Berger, 2000) is designed for measuring the similarities between the query terms and the search results. We adapt this algorithm to weight our keywords and their relative next-keywords in order to make the keyword

sequences more reliable. Also, we can make the relationship between each two keywords stronger. This is the BB's graph-based clustering algorithm:

$$BB(m,n)=\begin{cases}\dfrac{|\{l|(m,l)\in E \text{ and } (n,l)\in E\}|}{|\{l|(m,l)\in E \text{ or } (n,l)\in E\}|}, & if\ |\{l|(m,l)\in E \text{ or } (n,l)\in E\}|>0,\\ 0, & otherwise,\end{cases} \quad (3)$$

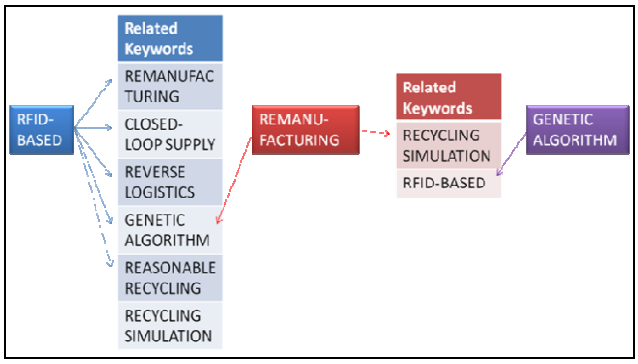


Figure 6. BB algorithm.

As we can see in Fig. 6, "RFID-BASED" and "REMANUFACTURING" have six relative keywords, and there is one keyword that is relative to both of them. Thus, the BB score will be 0.167. Also, the BB score between "REMANUFACTURING" and "GENETIC ALGORITHM" will be zero. By using the BB algorithm, we can make the keywords' relations stronger. Without the BB algorithm, the extracted sequence will always be longer and the representative ability will not be sufficient.

3.1.4. Hop-by-Hop Routing algorithm(HHR)

We can combine the algorithms which we introduced before into one synthesized algorithm as follows:

$$D(m_i \rightarrow m_{i+1}) = BB(m_i, m_{i+1}) * PR(m_{i+1}) * (GCD(m_i, m_{i+1}))^2$$
$$D(m_i \rightarrow m_{i+1} \ldots \rightarrow m_{i+n}) = (D(m_i \rightarrow m_{i+1}) + D(m_{i+1} \rightarrow m_{i+2}) + \ldots + D(m_{i+n-1} \rightarrow m_{i+n}))/n \quad (4)$$

In Fig. 7, the 3-gram keyword sequence is used to provide an example. According to our HHR algorithm, the value of the Top-1 to -3 sequence will be *(D(m0, m1) + D(m1, m2))*/2. We use the same algorithm to calculate the HHR value and to find the D(m0, m2). If the 3-gram HHR value is more than the *D(m0, m2)*, then we treat this 3-gram sequence as valid. Therefore, the keyword sequence will become *m0* -> *m1* -> *m2*. In the same way, the system will repeat this process and continue to check until the threshold is reached.

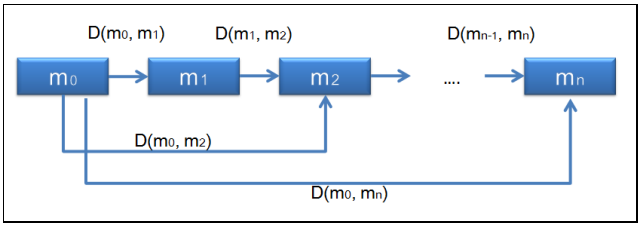


Figure 7. Expansion score checking.

Here, we can get some different lengths of sequences of each document or webpage. The definition of the n-gram sequence can be shown as follows:

$$m_1, m_2 \text{ is a } 2-gram\left(\text{denoted as } m_1 \to m_2\right) \text{ if } (m_1, m_2) \in \mathrm{E}$$

$$m_1, m_2, \mathrm{m}_3 \text{ is a } 3-gram\left(\text{denoted as } m_1 \to m_2 \to m_3\right)$$
$$\text{if } m_1 \to m_2 \text{ and } (m_2, m_3) \in \mathrm{E} \text{ and } (m_1, m_3) \in \mathrm{E}$$
$$\text{and } D(m_1 \to m_2 \to m_3) > D(m_1 \to m_3)$$

$$m_1, m_2 \ldots \mathrm{m}_n \text{ is an } n-gram\left(\text{denoted as } m_1 \to m_2 \to \ldots \to m_n\right)$$
$$\text{if } m_1 \to m_2 \to \ldots \to m_{n-1} \text{ and } (m_{n-1}, m_n) \in \mathrm{E}$$
$$\text{and } (m_1, m_n) \in \mathrm{E} \text{ and } D(m_1 \to m_2 \to \ldots \to m_n) > D(m_1 \to m_n)$$

3.2. Similarity Measurement using Google Purity

In order to classify the documents, we should create a "specific category" (*sc*) as the knowledge domain and select a representative document for each of them. Then, we can use the document's "typical sequence" (*ts*), where ts is a longest n-gram of the document, to represent the vector of each category. When we are downloading some new and unclassified documents, we can measure the similarity of its ts with each categorys' representive sequence (*rs*) to decide which is the closest category and put the document into it.

Here are two main questions:

- Can any document be the representative document for each knowledge domain?
- How to measure the similary of documents only using the n-gram sequence?

The ts is a sequence which is combined with no more than four multiwords. Thus, it will be a great challenge to measure the similarity using only four-by-four keyword realtion. In traditional analysis, we have to collect all of the keywords in each knowledge domain as vectors. There will be thousands of keywords and the vectors will be huge. But using the sequence which we introduced in the previous sections, there are only four keywords for the extracted documents and the base documents. Also, those keywords in most situations are totally different, so we cannot use the traditional vector-based similarity measurements such as cosine similarity to measure our n-gram sequence. We will introduce our new method, named "Google Purity Score" (*GPscore*) to solve the two problems which we mentioned earlier.

3.2.1. The Google Purity

Before we start to introduce our proposed similarity measurement method, we have to introduce the search results filtering function which is provided by Google in advance. The Google search engine provides two types of automatic filters:

- Duplicate Snippet Filter (*DSF*)
- Duplicate Directory Filter (*DDF*)

The *DSF* is used to filter out those documents which have the same titles, and provide the webpages or documents which have the highest ranking score. The *DDF* is used to filter out the results which belongs to the same Web directory. For each Web directory, the Google search engine will only select two results to provide to the user. By using these two filtering methods, the number of search results will be highly restricted so that users can read the information in a more efficient way. Also, the content is more important and accurate to the users' requirements. In this way, we found that perhaps we could use this specific feature to calculate the similarity between the sequences.

We can input the whole ts into Google, where g($m_1 \to m_2 \to ... \to m_n$)=g($m_1$ *or* m_2 *or...or* m_n), and get the total number of search results about ts in the first page (*page(g(ts)*,1)). The sequence „$m_1 \to m_2 \to ... \to m_n$" is a *ts* of the document. But when we keep reading the results which the Google search provided to us, the number of search results will become less and less. Finally, the Google will provide a page and show that some search results are omitted because they are too similar to the results of previous pages. Thus, we can use the last page's number of search results (*page(g(ts)*,end)) to represent the number of important individual search results. We can use the formula as follows to calculate how many individual results there are.

$$GP_{score}(ts) = \frac{page(g(ts), end)}{page(g(ts), 1)} \qquad (5)$$

But we quickly found a problem: how to extract the last page of the search results? Some sequences may contain a lot of individual results, so that it will not be easy to get the last page. Thus, we will use the *10th, 20th, 30th, 40th,* and *50th* pages' number of search results to calculate five *GPscore*, and summarize them into one synthesized value as the similarity measurement. In our experience, if two sequences do not belong to the same domain, their number of search results will drop down very fast, so that the last page which Google will provide to us will be no more than *50th*. The algorithm is as follows:

$$GP_{score}(ts) = \sum_{i=1}^{5} GP(ts)_i = \sum_{i=1}^{5} \frac{page(g(ts), i*10)}{page(g(ts), 1)} \qquad (6)$$

The $page(g(ts), i*10)$ represents the number of search results of the Google's i*10th results page by using ts. We will use the example to show how we calculate the GP_{score} and how we can use this simple algorithm to find the similarity between each two sequences in the following sections.

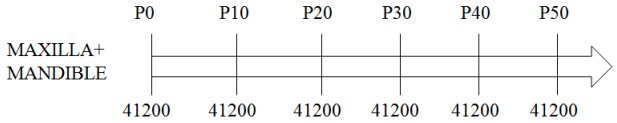
3.2.2. Base Sequence Selection

The proposed similarity measurement is based on each knowledge domain's rs compared with the ts of the unclassified document. Thus, we can imagine that choosing a most representative sequence is an important issue. This raises the question of whether / every document's sequence can be used to represent the domain knowledge.

The answer to the question is definitely "No." The sequence should be checked by using the *GP* algorithm to make sure that the sequence is pure enough to represent the knowledge domain. It means that those keywords in the same base sequence should not contain too much information about other knowledge domains. We can conclude the rs as follows:

rs : is a representive sequence of sc
if there exists a document in sc
and rs is a typical sequence of the document
and rs satisfies with $GP_{score}(rs) \geq 4$

For example, if we enter a small sequence "Maxilla + Mandible" to the Google, the number of search results is as follows:



After we get the number of search results, we can use the *GP* to check whether this is a pure sequence or not. The calculation is as follows:

$$GP_{score}(ts) = \frac{41200}{41200} + \frac{41200}{41200} + \frac{41200}{41200} + \frac{41200}{41200} + \frac{41200}{41200} = 5$$

Our experience is that the sum of those five *GPscore* should be more than four. If the sum is less than the threshold, the accuracy of classification will be affected by the impure base sequence. By using the *GP* algorithm and the checking threshold, we can easily choose these pure base sequences to classify the documents. Therefore, we can use the *GPscore* to find out whether these two sequences are pure enough and belong to the same knowledge domain or not

3.2.3. Similarity measurement

After the system chooses the best pure base sequence for each knowledge domain, we can start to automatically download the documents and classify them into their relative domains. But the problem is how to measure the similarity by using only n-gram sequence. The only solution for this problem is also the Google Purity ratio. We use Fig. 8 as an simple example to introduce our proposed method.

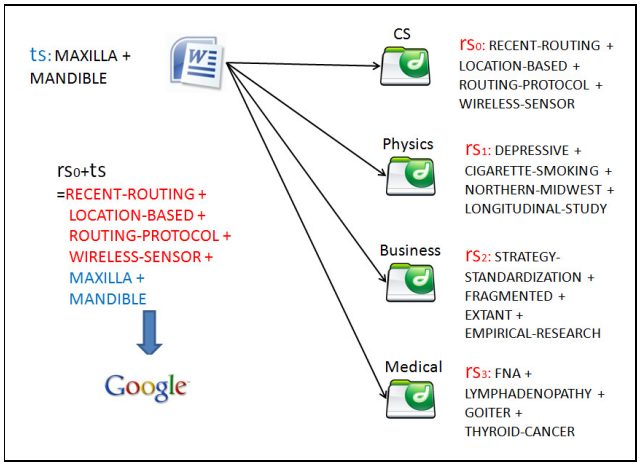


Figure 8. Document classification method.

Suppose we have several rs and an unclassified ts, which is "Maxilla + Mandible," ready for classifying. We will first combine each of the rs with the ts, and send the whole combined sequence into Google to calculate the *GPdistance* for each category. The method for combining the sequence to get the number of search results is as follows:

$$g(n_1 - gram, n_2 - gram) = g(m_{11}\ or\ m_{12}\ or ... or\ m_{1n}\ or\ m_{21}\ or\ m_{22}\ or ... or\ m_{2n}) \quad (7)$$

Thus, we can find a category which has the highest *GPdistance* score to allocate the document to it. The algorithm of *GPdistance* is as follows:

$$GP_{distance}(ts,rs)=\sum_{i=1}^{5}\frac{page(g(ts,rs),i*10)}{page(g(ts,rs),1)} \quad (8)$$

We use the sum of the *GP(ts,rs)* instead of only the last page divided by the first page because in some situations, the value of the *GP(ts,rs)* will be the same or very close. Thus, using the sum of the *GP(ts,rs)* can make sure that only one knowledge domain will be chosen to allocate the document. But there is a special case: If the *GPdistance* values in different knowledge domain are the same, we will simply use the total number of search results to make a decision.

We can use an example to introduce this method in more detail to realize the comparing procedure. The *GPdistance(ts, rs0)* is composed of *rs0* which is from the computer science domain and ts which is extracted from a medical domain article. This combined sequence will then be inputed into the search engine and use the Google purity algorithm to get five different *GP* scores. Another combined sequence is *GPdistance(ts, rs3)* which is composed of a medical domain sequence *(rs3)* and the ts. Thus, we can also get five *GP* scores for this sequence. As Fig. 9 shows, the value of the *GPdistance(ts, rs3)* is higher than another one. Thus, this unallocated document will be classified into the medical domain.

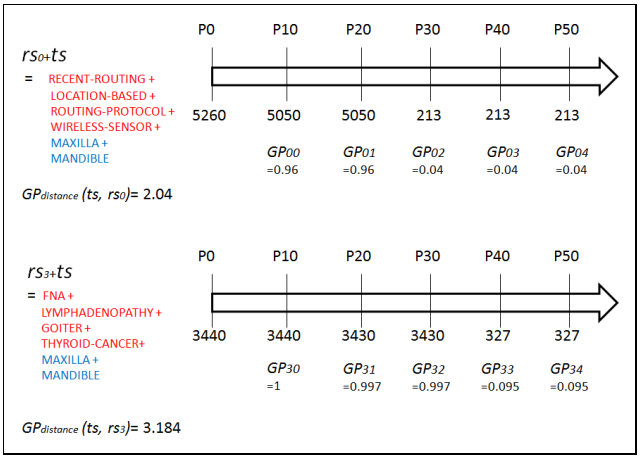


Figure 9. Google distance calculation.

We examine the keyword sequences and check each search results. We find that if the ts and rs belong to the same category. those search results which Google provides will contain almost all of the multiple keywords in these two sequences. But if they do not belong to the same domain, it will only contain most of the multiple keywords from only one sequence; the multiple keywords from another sequence will rarely appear. In this way, we can understand that if the multiple keywords from ts and rs both appear in more search results, the similarity between these two sequences will be higher.

But here comes another problem: "Can we use only one base sequence to represent the whole knowledge domain?" As we know, there are different kinds of knowledge in a single domain. For example, the domain of computer science may contain the knowledge of artificial intelligence, network security, software engineering and much more. Thus, if we get a base sequence from an article which is about artificial intelligence, the extracted sequence from network security can be successfully classified into the computer science domain. The answer is "Yes." The rs and ts may contain totally different keywords, but, when we combined them and searched in Google, the *GP* score will also be higher than the combined sequence when the rs and ts are from different

domains. The branches of knowledge in the same knowledge domain will appear in more webpages than those in different knowledge domains.

4. EXPERIMENTAL RESULTS

In this section, we conducted experiments that used the abstracts of 260 research papers to parse the most important sequential keywords and then use the Google purity measurement to automatically classify them into their relative knowledge domains. These papers were randomly selected from the Elsevier Web site. We first chose three knowledge domains and several journals for each domain. In each journal, we chose the 10 most-downloaded articles as our dataset. Then we added a new knowledge domain to see whether it would reduce the accuracy of the original three domains or not. Our system's design goal is to create files for the users so that the number of files in the repository should increase as time goes by. The newly added base sequence can not effect the accuracy of the previous base sequences.

4.1. Pure Sequence Rate

In our system design, the user can choose a document to represent a new knowledge domain. The biggest question was whether it was possible to use every document's sequence as the base sequence for each domain. At first, we thought it might be possible because if the sequence can fully represent the document, the sequence should be strong enough to represent the domain to which it belongs. But we quickly found a problem after we combined some sequences with the extracted sequences in the same domain. The calculated GPdistance will sometimes be less than the ts if combined with another domain's base sequence. Thus, the accuracy will be not good enough and the ts will be allocated into the wrong domain.

We did a lot of experiments and found that only some stronger sequences, which have higher GP_{score}, can be used to represent the knowledge domain. Therefore, the system should check whether the sequence can become a base sequence or not. The users need to create some documents in the same file so that the system can choose the best sequence to represent the domain. That raises the issue of whether we could get the strongest representative sequence from documents in the same file. We use all of the extracted sequences to examine their GP_{score}. Our goal is to measure the percentage of the "Pure Sequence" in each knowledge domain. This should help us decide how many documents should be collected in advance to increase the ability to find a pure sequence. As Fig. 10 shows, we can see that in most situations, the GCD algoritm using the largest distribution method can get a purer sequence than the closest related method. In the computer science (CS), business and medical domains, there was more than a sixty-five percent chance to find a pure sequence. We can put a few documents into the file, and the pure sequence will soon be decided. In the CS domain, almost every document which we selected can serve as the representative document. But, in the psychology domain, the pure sequence was hard to find. We examined the psychology sequences' Google search results, and found that half of the extracted keywords in the sequences are similar to the business domain's keywords. The research articles in business management often measure the user behavior to realize the effectiveness of the business or system operation performance. Also, the number of search results in the business domain is far larger than psychology's. Therefore, the number of pure sequences to represent the domain's knowledge were not easy to find.

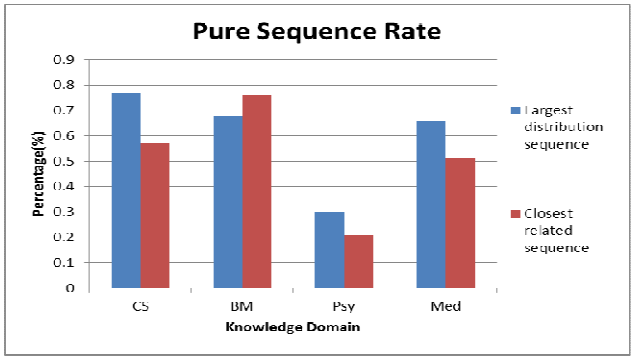


Figure 10. Pure sequence measurement.

4.2. Classifying Accuracy Measurement

First, we will use three knowledge domains to evaluate the accuracy of the system. We chose computer science, psychology, and business as our test domains. The reason for choosing these three domains is that they are highly cross-related. Research articles in information management can belong to computer science, business, and psychology. We think that if we can successfully classify the articles in these three knowledge domains accurately, we can say that our proposed method is useful and can achieve high accuracy.

As we mentioned in the previous section, we considered four different situation for using GCD and AGCD combined to extract those top 50% smallest relations (closest related method) or top 50% largest relations (largest distribution method). When we did the experiment, we quickly found that the experiment results which used the AGCD algorithm were notably worse. Almost all the documents were classified to the computer science domain. We believe this is because the classification should use those keywords which are more general in their knowledge domain, instead of the specific keywords in a single document. Otherwise, the keywords in rs which belong to the computer science domain will always contain more search results. Thus, if there are specific terms in the ts sequence, the ts will be closed to the rs which is in computer science. But if we want to use the sequence to automatically find out some relevant information about this article, we can use the AGCD algorithm to get better results than our previous work. In this paper, we only want to focus on the classification results of our new proposed method. Therefore, our experiment results will only display the two kinds of results using the GCD algorithm.

As Fig. 11 shows, we selected seven different journals in the computer science domain. We randomly chose these journals to test our proposed system's classification ability. Those journals which have higher unicity can achieve high accuracy by using our system. We evaluated the extracted keyword sequences of the an, aes, and cn. Most of the composed keywords were terminology, so they will rarely be used by other knowledge domains. In this way, the accuracy of these three journals is very high. The last four journals contain some cross-domain research articles, especially the dss and eswa, and most of the articles were classified to the business domain. Actually, in our experience, some of the dss articles should be classified into the business domain because most of the research in this journal is focused on information management problems. But the Elsevier Website put this journal in the sub-page of computer science. Thus, we use it as the guideline to measure how our system might sometimes mismatch with our common sense. Also, the accuracy of the execution results will be affected. The journal articles in eswa which we extracted all contains fewer specific terms, and some of them are focused on management issues. Thus, the execution results are much worse because a portion of the extracted keywords contains management terminology. For this reason, they were classified into the wrong domain.

Using only a four-gram keyword sequence to represent the document and measuring the similarity with another four-gram sequence is not easy. But if we ignore the classification of the Elsevier Website and analyze each article in detailwe can see that the accuracy is very high.

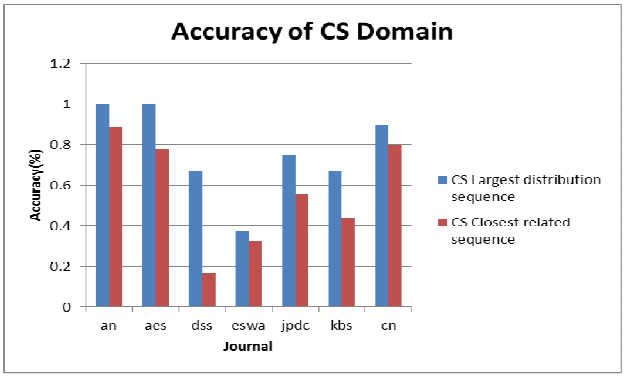


Figure 11. Classification results of the computer science domain.

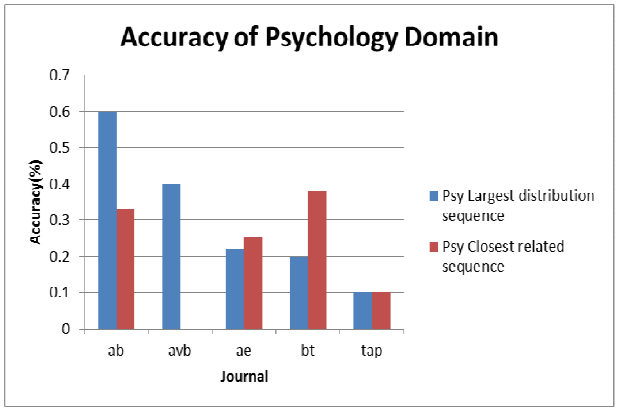


Figure 12. Classification results of the psychology domain.

In the beginning, we did not know that the purity of the base sequence would deeply affect the accuracy of the system. Thus, we randomly selected a sequence to represent the psychology domain, and the results were frustrating. The accuracy was only about thirteen percent. After we used the *GP* algorithm to evaluate and find a representative pure sequence, the accuracy became much higher than before, as we can see in Fig. 12. Also, the execution results show that using the largest distirbution method can be more accrate than the old closest related method. The keywords in this domain have always been used by the business domain, so some articles will be mis-classified to the business domain. Also, some research dealt with the psychological effects of information technology. Therefore, these kinds of articles will also be mis-classified. The accuracy of this domain when we used the largest distribution method is better, but, the overall accuracy is not good enough to classify the documents. Perhaps we should use longer keywords or other techniques to solve this problem in the future.

The last knowledge domain is business management, as shown in Fig. 13. Most of the mis-classified documents are allocated to computer science and psychology. But in most situations, we could achieve more than sixty percent accuracy. Some journals' accuracy was about eighty precent. The journal which had the lowest accuracy rate is iam (Information and Management). It is also the journal which focuses on information technology management's problems and user behavior relating to new technology. Thus, it will not be easy to allocate each document in this journal to the business domain. But here we can find that using the closest related method will be

a little bit better than the largest distribution. The experiment results show that the difference between our two proposed methods is not too large.

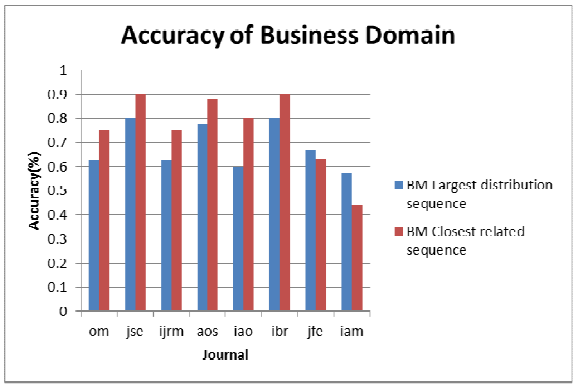


Figure 13. Classification results of the business management domain.

Finally, we added a new domain, medical, to measure whether it would affect the original three domains' accuracy. We chose the medical domain and randomly chose six journals because some of the articles may contain keywords relating to computer science, psychology, and management. Thus, if our Google purity algorithm is not effective enough, the accuracy of the classification in the original domains will be lower.

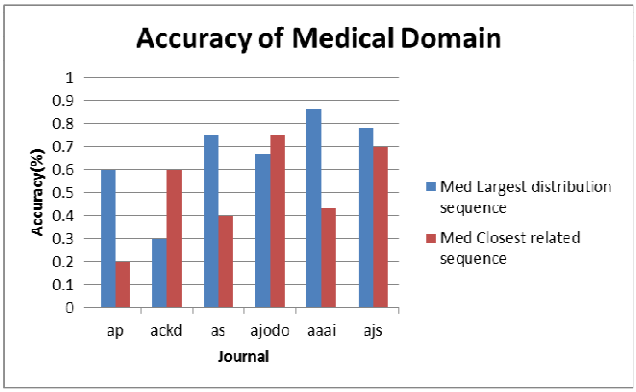


Figure 14. Classification results of the medical domain.

After the experiment, we found that the previous three domains can maintain the same accuracy and will not be affected by the newly added domain. Also, in the medical domain, the accuracy using the largest distribution method will be better than the closest related method, as Fig. 14 shows. In most situations, the accuracy was more than sixty percent. The keywords which the system extracted in this domain were more specific and most of them will not appear in other domains. We repeated the experiment several times and sometimes the accuracy of these six jouanals achieved almost ninety percent. But for this paper, we wanted to use the same representative sequence to compare the two algorithms. Thus, the results in Fig. 13 do not contain the best results of our system. The accuracy of this domain can be extremely high. Also, using the largest distribution method can get better classification results.

Table 2. Vector-size comparison of three different methods.

Method	Knowledge Domain			
	Computer Science	Business Management	Psychology	Medical
GCD+Cosine	46 words	54 words	75 words	261 words
All+Cocine	458 words	332 words	454 words	365 words
Our Method	4 words	4 words	4 words	4 words

We also conducted the experiments using the traditional vector-based method to classify the documents and compare them with our newly, proposed method. To construct the representative vectors, we randomly selected 25% of the documents in each knowledge domain as our training dataset. The first experiment compared the accuracy of Google purity with the cosine similarity measurement. We used the *GCD* algorithm to extract the important keywords from the training dataset, and the cosine algorithm to classify the documents. In this way, we were able to understand whether or not our proposed similarity measurement algorithm was useful. The second experiment used all of the keywords, except for stop words, to construct the representative vectors and classify the documents. Most vector-based document classification methods use this type of method; many studies have proved that this method can achieve high accuracy in classification problems.

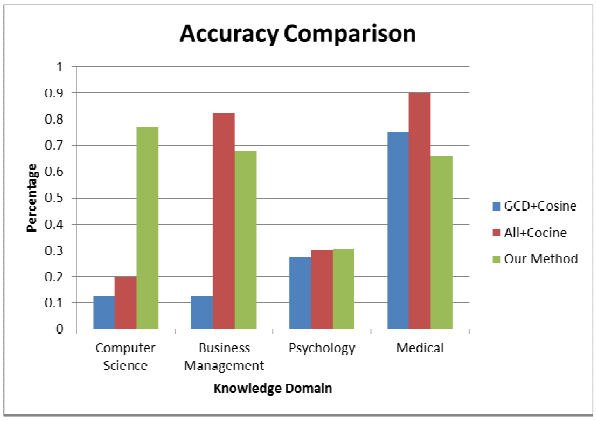


Figure 15. Classification Accuracy Comparison of three different methods.

The cosine similarity algorithm is defined as:

$$\cos\theta = \frac{d \cdot v}{\|d\| \|v\|} \qquad (9)$$

where $d \cdot v$ is the intersection of the document and the training data vector, $\|d\|$ is the norm of vector *d*, and $\|v\|$ is the norm of vector *v*. The representative vectors of our proposed method are only four keywords. On the other hand, if we use all of the keywords as our vectors, vector size increases and becomes very large as time goes by. Even by using the *GCD* algorithm to filter out some unimportant keywords from the documents, vector size remains more than ten times that of our method. Fig. 15 illustrates the accuracy measurement, which shows the use of the same *GCD* keyword extraction method and only the evaluation of the accuracy of the Google purity algorithm and the cosine similarity measurement. The accuracy of using our proposed method

will be better than the traditional method in every knowledge domain, except for the Medical domain. The Medical domain contains a lot of potential keywords, enabling it to achieve high accuracy by using *GCD + cosine* similarity.

Although there are different kinds of research issues in each domain, the accuracy is only dependent on the pure base sequence. We can see that the results will not be affected even though we added a new knowledge domain. We think this Google purity algoirthim is useful in real life. In our daily life, we constantly added new files in our computer's repository. Thus, the documents which we collected will be classified by the original base sequencs, and the addition of a new base sequence should not affect the whole system's accuracy.

5. CONCLUSIONS

In this paper, we proposed a new document classification system which can use only 4-gram keyword sequences to represent each document and use the new Google Purity algorithm to measure the similarity of the sequence. Therefore, the computational cost will be greatly reduced compared to the traditional document classification methods. Also, all the computations in this system are based on the similarity measurement which uses the co-occurrence probability between the keywords in the Google search engine. We adapted our previous WANET system and prove that the classification results will be better by using this new method. The advantage of this system can be concluded as follows: (a) it extracts the keywords without a pre-established keyword repository; (b) it has the smallest keyword vectors; (c) it can classify the documents in real-time. The experiments show that the accuracy will be very high if we can find a pure sequence to represent the domain knowledge. Also, if we add a new domain into the test set, the whole accuracy of the previous added knowledge domain will not be affected by the new one.

We believe this system can be useful in Web information retrieval. The knowledge will be gathered from all kinds of resources and knowledge domains to help a company's decision making. It will also help users, especially researchers engaged in broad searches of information from the Web to manage downloaded documents. The system can immediately calculate and find the most similar file, then allocate the document into it. The only weakness of this system is that the execution time of the sequence extraction is too long.

We will continue to reduce the execution time of this system and make it more suitable to real life. Also, we are trying to automatically conduct the ontology from each document so that after the system has classified those documents into the knowledge domain, we can use the hierarchical structure to classify the documents in each knowledge domain in more detail. For example, the computer science domain might be further automatically sub-divided into IR, AI, Network, Security, and so on. But the computational cost is still too high and needs to be solved.

Recently, we have tried to use cache methods to improve the execution efficiency. By constructing a small search engine to transfer most of the queries from Google to our private search engine, the system can be more usable in our daily life. We will keep trying to improve the cache method and find other techniques to use the search engine based keyword similarity measurement in further IR research.

ACKNOWLEDGEMENTS

This study is conducted under the "III Innovative and Prospective Technologies Project" of the Institute for Information Industry which is subsidized by the Ministry of Economy Affairs of the Republic of China.

REFERENCES

[1] Beeferman, D. & Berger, A. (2010) "Agglomerative Clustering of a Search Engine Query Log, " Proc. ACM SIGKDD.

[2] Chen, P.I. & Lin, S.J. (2010) "Automatic Keyword Prediction Using Google Similarity Distance." Expert Systems with Applications, Vol. 37, No. 3, pp 1928-1938.

[3] Chen, P.I. & Lin, S.J. (2011) "Word AdHoc Network: Using Google Core Distance to Extract the Most Relevant Information." Knowledge-Based Systems, Vol. 24, No. 3, pp 393-405.

[4] Cilibrasi, R.L. & Vitanyi, P.M.B. (2007) "The Google Similarity Distance." IEEE Transactions on Knowledge and Data Engineering, Vol. 19, No. 3,pp 370 – 383.

[5] Edith, H., Rene, A.G., Carrasco-Ochoa, J.A. & Martinez-Trinidad, J.F. (2006) "Document Clustering Based Onmaximal Frequent Sequences." Proceedings of the FinTAL 2006, 4139, pp 257–267.

[6] Feng, F. & Croft, W. (2001) "Probabilistic Techniques for Phrase Extraction." Information Processing and Management, Vol. 37, pp 199–200.

[7] Freund, G.E. & Willett, P. (1982) "Online Identification of Word Variants and Arbitrary Truncation Searching Using a String Similarity Measure." Information Technology: Research and Development, Vol. 1, No. 3, pp 177-87.

[8] Fung, B., Wang, K., & Ester, M. (2003) "Hierarchical Document Clustering Using Frequent Itemsets," Proceedings of the 3rd SIAM International Conference on Data Mining.

[9] Godoy, D. & Amandi, A. (2006) "Modeling User Interests by Conceptual clustering." Information Systems, Vol. 31, No. 4-5, pp 247-265.

[10] Google Search Appliance, Creating the Search Experience: Best Practices, https://developers.google.com/search-appliance/documentation/50/admin_searchexp/ce_improving_search

[11] Li, Y.J., Chung, S.M. & Holt, J.D. (2008) "Text Document Clustering Based on Frequent Word Meaning Sequences." Data & Knowledge Engineering, Vol. 64, pp 381-404.

[12] Li, S. J., Wang H. F., Yu S. W. & Xin C. S. (2003) "News-Oriented Keyword Indexing with Maximum Entropy Principle [J]." Proceedings of PACLIC17, pp 277-281.

[13] Lin, C.Y. & Hovy, E.H. (2003) "Automatic Evaluation of Summaries Using n-gram Co-occurrence Statistics." Proceedings of Human Language Technology Conference.

[14] Luo, C., Li, Y. & Chung, S. M. (2009) "Text Document Clustering Based on Neighbors." Data & Knowledge Engineering, Vol. 68, No.11, pp 1271-1288.

[15] Papineni, K., Roukos, S., Ward, T. & Zhu, W.J. (2001) "BLEU: a Method for Automatic Evaluation of Machine Translation." IBM Research Report RC22176 (W0109-022).

[16] Salton, G., Yang, C. S. & Yu, C. T. (1975) "A Theory of Term Importance in Automatic Text Analysis." Journal of the American Society for Information Science, Vol. 26, No. 1, pp 33-44.

[17] Sato, K. & Saito, H. (2002) "Extracting Word Sequence Correspondences With Support Vector Machines." Proceedings of the 19th International Conference on Computational Linguistics, Taipei, Taiwan, pp 1-7.

[18] Shardanand, U. & Maes, P. (1995) "Social Information Filtering: Algorithms for Automating Word of Mouth." Proceedings of the Computer-Human Interaction Conference (CHI'95).

[19] Steinbach, M., Karypis, G. & Kumar, V. (2000) "A Comparison of Document Clustering Techniques." Proc. KDD-2000 Workshop TextMining..

[20] Yang, J., Wang, W., Wang, H. & Yu, P.S. (2002) "Delta-clusters: Capturing Subspace Correlation in a Large Data Set." Proceedings of the ICDE, pp 517-528.

[21] Zhang, D. & Lee, W. S. (2004) "Web Taxonomy Integration Using Support Vector Machines." Proceedings of the 13th International World Wide Web Conference.