


Clustering of Relevant Documents Based on Findability Effort in Information Retrieval


Prabha Rajagopal, Monash University, Malaysia

 <https://orcid.org/0000-0002-5734-3691>

Taoufik Aghris, EMINES-School of Industrial Management, Mohammed VI Polytechnic University, Morocco

Fatima-Ezzahra Fettah, EMINES-School of Industrial Management, Mohammed VI Polytechnic University, Morocco

Sri Devi Ravana, University of Malaya, Malaysia*

 <https://orcid.org/0000-0002-5637-9158>

ABSTRACT

A user expresses their information need in the form of a query on an information retrieval (IR) system that retrieves a set of articles related to the query. The performance of the retrieval system is measured based on the retrieved content to the query, judged by expert topic assessors who are trained to find this relevant information. However, real users do not always succeed in finding relevant information in the retrieved list due to the amount of time and effort needed. This paper aims 1) to utilize the findability features to determine the amount of effort needed to find information from relevant documents using the machine learning approach and 2) to demonstrate changes in IR systems' performance when the effort is included in the evaluation. This study uses a natural language processing technique and unsupervised clustering approach to group documents by the amount of effort needed. The results show that relevant documents can be clustered using the k-means clustering approach, and the retrieval system performance varies by 23%, on average.

KEYWORDS

Clustering, Evaluation, Findability Effort, Information Retrieval, Machine Learning, TREC, Unsupervised Learning, Word2Vec

INTRODUCTION

Information retrieval (IR) is the science of searching information in documents relevant to a given query, from within large stored collections. The fundamental challenge of an information retrieval system (IRS) resides in matching between an information requirement statement, precisely a user's query, and a collection of documents by ranking each one according to its importance for the query.

DOI: 10.4018/IJIRR.315764

*Corresponding Author

This article published as an Open Access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0/>) which permits unrestricted use, distribution, and production in any medium, provided the author of the original work and original publication source are properly credited.

During the past decades, a huge amount of research was done to build a ranking model to retrieve the best relevant documents. Generally, a ranking model is either constructed with probabilistic methods or modern machine learning methods. The algorithm is based on the frequency of words, considering that a document is a set of words, often called a word bag. With these models, if a user enters a simple query, for example, “what is information retrieval” in a given IRS, hundreds of thousands, if not million results are retrieved and ranked. However, sometimes a large amount of time is spent just to get a small piece of information in those documents which are considered relevant. The amount of effort put in by the user, either satisfies or dissatisfies the user in gaining the necessary information knowledge. It was mentioned before that real users tend to give up easily when searching for information in the retrieved documents (Verma et al., 2016). Therefore, the concept of relevance no longer remains in just ensuring relevant information is available in the document but also the amount of effort needed in finding relevant information (Yilmaz, 2014).

Two widely used methods evaluate the effectiveness of information retrieval systems. The first method is called the collection-based method and it is often referred to as the Cranfield approach (Cleverdon, 1991). This approach is based on a document collection (corpus), a set of topics that contain the query, title and description to define a user’s need, and a set of relevance judgments pointing out the relevant documents in the collection to each topic, often judged by topic experts. So, to evaluate the effectiveness of IRS, the scores for the systems are generated using the retrieved ranked list of documents by the systems and the relevance judgment. The scores are calculated using evaluation indicators such as precision, recall, mean average precision, and others (Clough & Sanderson, 2013). The second evaluation method is the user-based evaluation. This approach is based on the interaction between the user and the IRS which is defined by the user’s environment such as his/her educational background, the context, subject expertise, and his/her perspective like the search goal (Park, 1994).

Comparing both the evaluation methods, the system-based and the user-based evaluation can match each other’s results (Al-Maskari, 2008). However, previous research has shown there is a broad gap between these two approaches, given that the collection-based method makes many hypotheses about what the real user looks for to satisfy his/her information needs. Additionally, there are many other assumptions to simplify the relevance evaluation (Allan et al., 2005). So, the mismatch between the two evaluation methods is due to the dissension between what the expert judges consider as relevant documents, and what the real users need to satisfy their information demand. The user’s need is specified as document utility (Turpin & Hersh, 2001). Evaluating IR relevance by documents utility in a semantic and pragmatic view was argued by Saracevic (1979) in earlier research (Saracevic, 1975) as follows: “it is fine for IR systems to provide relevant information, but the true role is to provide information that has utility-information that helps to directly resolve given problems, that directly bears on given actions, and/or that directly fits into given concerns and interests. Thus, it was argued that relevance is not a proper measure for a true evaluation of IR systems. A true measure should be utilitarian.” Following that, Yilmaz et al. stated that relevance is about how documents found by the retrieval system are useful (2014).

Hence, the IRS models must consider effort, especially that users are generally impatient and do not quickly capture evidence of relevance (Yilmaz, 2014). Ease of finding information leads to satisfied users. In this context, the effort can be defined as the amount of work the user needs to identify and consume information from a relevant document (Villa & Halvey, 2013). But, the question arises, how to determine if a given relevant document requires a high or low amount of effort? The effort could be measured through readability, findability, and understandability. Readability measures the amount of effort required to read a document, findability measures the effort needed to find the relevant information in a document, and understandability measures effort required to understand a document to satisfy the information need.

Previous studies focused on readability measures and their importance in relevance judgment (Villa & Halvey, 2013). The classification of a document through readability was based on the range of the readability indexes. For example, if a readability index takes a value between a given

interval, a score is assigned to it and each document is classified accordingly. Nevertheless, there is a lack of studies in determining the effort of findability and understandability. This paper has two objectives, the first is to utilize the findability features to determine the amount of effort needed to find information from relevant documents using an unsupervised machine learning approach, and the second is to demonstrate changes in information retrieval systems' performance when the effort is included in the evaluation.

The remaining of the paper is as follows: Section 2 covers the various natural language processing approaches used in text processing and followed by unsupervised machine learning methods in Section 3. Section 4 details the methodology, while the results and discussion are in Section 5. Finally, the conclusion is in Section 6.

TEXT PROCESSING

Natural language processing (NLP) has become a part of the information, document, and text retrieval. In the NLP, text processing, morphological analysis (stemming and lemmatization), syntactic analysis, and lexical semantics are among the common NLP tasks (Natural Language Processing, 2020). Stemming is a process to reduce different word forms of a word like its noun, adjective, verb, adverb, etc. to its base or root form. While lemmatization is the process of removing inflectional endings of words to return the base or dictionary form of a word. Both stemming and lemmatization have been studied widely in information retrieval and have been recognized to improve document retrievals (Balakrishnan & Lloyd-Yemoh, 2014). Alternatively, the vector space model is mostly used in information filtering, indexing, and relevance rankings. In a past study, the term frequency-inverse document frequency (TF-IDF) algorithm was compared with one that used stemming and lemmatization. Documents retrieved using the same query for each of the algorithms proved that stemming and lemmatization is better than TF-IDF for document retrieval (Balakrishnan & Lloyd-Yemoh, 2014). Instead of a document, a recent study focused on sentence retrieval using the term frequency-inverse sentence frequency (TF-ISF) together with the stemming and lemmatization language modeling technique (Boban et al., 2020). And their study claimed that data pre-processing using lemmatization with longer queries produces better results compared to stemming.

The Word2Vec is also an NLP technique that can be used to learn word association from a large corpus of text (2020) and uses a neural network model. Once trained, the model can detect synonymous words. In general, the Word2Vec is based on the distributional hypothesis where words that are used and occur in the same contexts tend to imply similar meanings. Hence, by looking at its neighboring words, it is possible to predict the target word. Since its emergence, the Word2Vec has been instrumental in the retrieval tasks and has shown positive outcomes when used in text classification. This algorithm can be used in both supervised and unsupervised machine learning (Lilleberg et al., 2015). A past study experimented on the effectiveness of information retrieval using CBOW (Continuous Bag of Words) and SG (Continuous Skip-Gram) models of the Word2Vec algorithm (Zuccon, 2015). The study found that the neural translation language model is statistically significantly better than the Dirichlet. Additionally, it experimented that word embedding does not have to come from the same corpus of words used for retrieval. Yet, another study had also proved that information retrieval when combined with Word2Vec achieves better retrieval accuracy compared to retrieval without a neural network model (Van Nguyen, 2017). A separate study attempted multilabel text classification using the semantic feature of Word2Vec and the results show improved performance compared to the traditional bag of words and the TF-IDF (Rahmawati & Khodra, 2016). Besides long text such as documents, word processing has a strong influence on text classification for short texts such as tweets (Paalman, 2019) and microblogs (Singh et al., 2016).

UNSUPERVISED MODELING

There are common clustering machine learning algorithms such as hierarchical clustering, k-means, mixture models, DBSCAN, and OPTICS algorithm (Unsupervised Learning, 2020). Of these, the k-means or DBSCAN algorithms seem appropriate for this study as it attempts to cluster documents with findability features. The main purpose of unsupervised learning is to identify patterns that were not known previously. There had been numerous studies on measuring the differences between documents, which involve different clustering techniques such as k-means, HAC, and KNN (Gopala Rao & Bhanu Prasad, 2013; Kalaivendhan & Sumathi, 2014; Kanavos et al., 2018; Lin et al., 2013; Sindhiya & Tajunisha, 2014). Meanwhile, SMTP, Cosine, Euclidean, Jaccard, and others have been utilized in the past to measure similarities.

The density-based spatial clustering of applications with noise (DBSCAN) is considered the main density-based clustering technique (Braune et al., 2015; Latifi-Pakdehi & Daneshpour, 2021) that clusters a set of data based on their distribution. The advantage of DBSCAN is that the model itself determines the number of clusters unlike the k-means model, and it can detect noise in the dataset that corresponds to outliers. In addition, the DBSCAN can find any shape of clusters and is not restricted to a circular shape. However, the main drawback of this algorithm is determining the two parameters, Epsilon, and minPts (Braune et al., 2015; Latifi-Pakdehi & Daneshpour, 2021).

Separately, the k-means clustering is based on dividing data into groups that have similar properties. The target number of clusters, K can be determined formally by reducing the in-cluster sum of squares. While k-means is very good at identifying clusters with a spherical shape, it is sensitive to the initial centroid (Zhang et al., 2008). Nevertheless, this algorithm is better for large document datasets, has low computational requirements (Gopala Rao & Bhanu Prasad, 2013; Zhang et al., 2018), preferred for its simplicity and still commonly used (Ezugwu, 2020; Ezugwu et al., 2022; Wu, 2019). In comparing both the k-means and DBSCAN, it's been stated that they are both equally good based on the SSE value (Karthika & Janet, 2020).

In data mining of text, a document is represented as a vector whereby each component specifies the value of its corresponding feature in the document (Han et al., 2011). In a clustering algorithm, the difference in distance is measured using Euclidean distance, Cosine similarity, Jaccard correlation, and Similarity measure for text processing. A previous study in data mining showed that the Similarity measure for text processing is better than Euclidean distance, Cosine similarity and Jaccard correlation for text classification of a real-world data set (Maher & Joshi, 2016). Studies exploring text processing and word embeddings have shown the euclidean distance is capable of effective descriptors of documents (Clinchant & Perronnin, 2013) and usage of Euclidean distance with k-means gives better results than using different distance metrics (Singh et al., 2013).

METHODOLOGY

The first purpose of our research is to discover the effort needed to find the relevant information in a given document. To do this, findability features are extracted by exploring and analyzing a set of collections. Then scores are calculated to quantify the semantical closeness between the document and the query. Once the scores are calculated, an unsupervised learning model is established to label the documents in terms of the effort needed to identify relevant information. The methodology consists of data collection, data processing and feature engineering, and modeling. Each of these phases is detailed in the next subsections.

Data Collection

For this research, the TREC-9 Web track was used due to the suitability and availability of the document corpus. The TREC test collection consists of three parts; a set of documents, a set of topics, and relevance judgment indicating the relevance of documents to a specific topic. The Web

track data set was used because this study intended to measure the effort of documents retrieved in the context of search engines.

In the TREC-9 Web track, the relevance judgment used ternary relevance; irrelevant (0), relevant (1), and highly relevant (2). However, for simplicity of calculation, the ‘highly relevant’ and ‘relevant’ documents were both considered relevant for this study. This test collection contains 105 runs and 50 topics. For each topic, a maximum of 1000 retrieved documents can be listed based on their similarity scores. However, less than 1000 retrieved documents can also be listed for any topic.

Data processing and feature engineering

After the data collection, the data undergoes processing and is represented in a suitable form for fitting a model. Figure 1 shows the data processing steps involved. In the first step, the files from each retrieval system, per topic, from TREC is in the form of HTML. These HTML files were then scraped to extract the content and saved as .txt files. These text files will be used consequently to build a summary and word corpus for each document as part of the text processing and feature engineering.

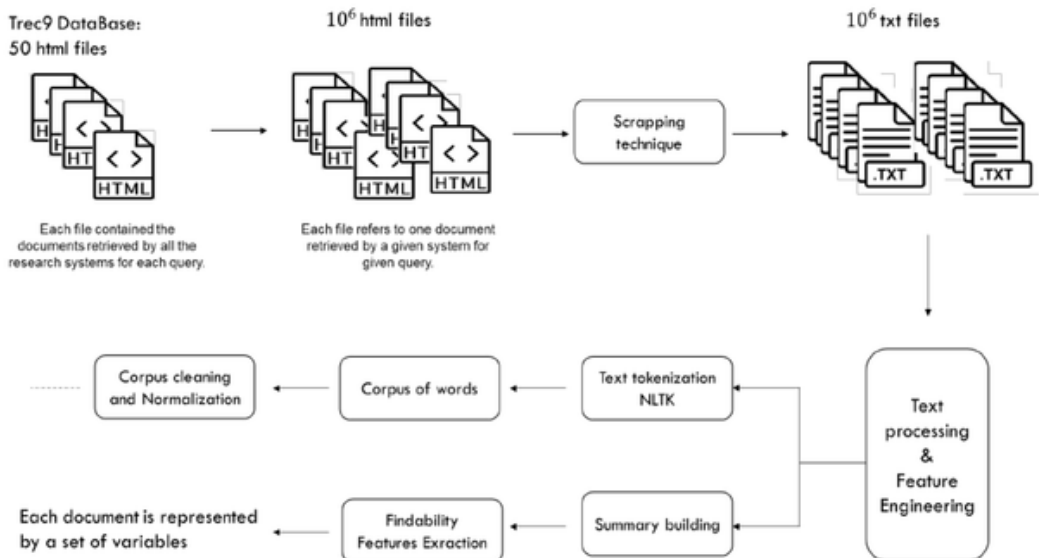
Text tokenization

The other part of text processing and feature engineering is the creation of word corpus. To build the word corpus for each document, the .txt files are normalized by tokenizing text into words using the Natural Language Toolkit (NLTK). So, each document is split into sentences, then by words. The document is represented as below after tokenization.

$$D_i = [[W_{11}, W_{12}, W_{13}, \dots, W_{1N_1}], [W_{21}, W_{22}, W_{23}, \dots, W_{2N_2}], \dots, [W_{n1}, W_{n2}, W_{n3}, \dots, W_{nN_n}]]$$

Where: D_i refers to the document/ N_j refers to the number of words in the j^{th} sentence in a given document; $j = 1, 2, \dots, n$ $W_k N_j$ refers to the word k in the j^{th} sentence in a given sentence.

Figure 1. Data preparation and processing



Word Corpus

After tokenization, the word corpus becomes available (Refer Figure 2). However, the current state of the word corpus is not suitable since it contains some noise that should be eliminated such as punctuation, stop words, prepositions, adverbs, pronouns, etc. Following this, non-ASCII words, stop words, and punctuations were removed. Besides that, all words were set to lowercase, and numbers were replaced with words.

Word Corpus Cleaning and Normalization

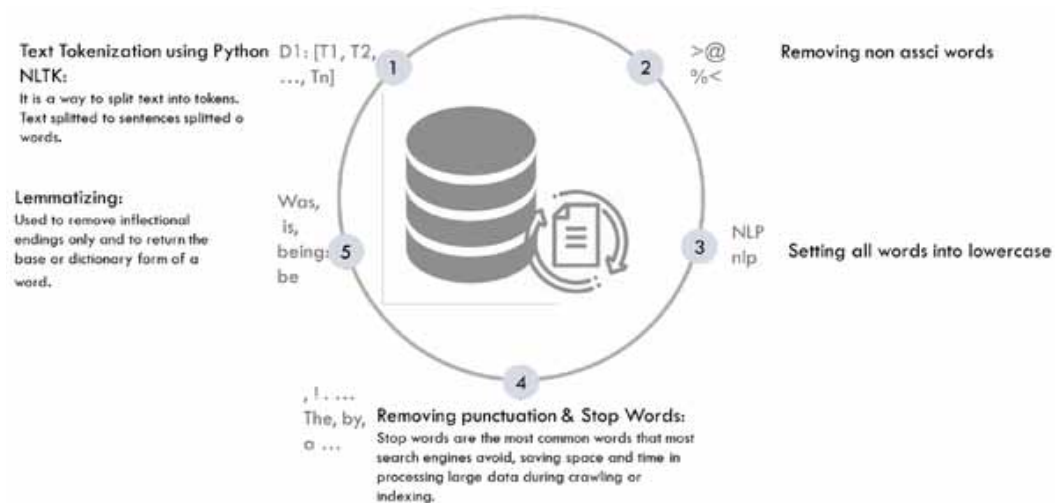
Generally, when training a model, if the corpus of words contains ‘car’, ‘cars’, ‘car’s’ and ‘cars’, the model will consider these words as different inputs. Hence, the model will build the same representation 4 times. As a result, the model builds redundant information, and the complexity of the training algorithm increases without further learning. Therefore, stemming and lemmatization are used to reduce inflectional forms and derivationally related forms of a word to a common base form.

Stemming is based on a crude heuristic process that chops off grammatically the modifications of words like tense, case, aspect, person, number and gender, and keep their roots by removing derivational affixes. Meanwhile, lemmatization normalizes the text with the use of a vocabulary and morphological analysis of words, aiming to remove inflectional modifications only and to return the base or dictionary form of a word, which is known as the lemma. Lemmatizing is more powerful than stemming in terms of the irregular forms of words. For example, the “er” at the end of the noun “water” is considered as a modification for stemming that turns it to “wat” as a root.

Summary Building

The summary is constructed from the sentences that contain at least one word of the query and the sentences before and after. For example, there is a document containing N sentences and a query with 3 words. Firstly, find the i^{th} query term in the j^{th} sentence of the document. Then the summary will contain the $(j-1)^{\text{th}}$, j^{th} and $(j+1)^{\text{th}}$ sentences for each query term. Similarly, a summary is built for each document. With this approach, the summary will contain information that the user would skim at first contact with the article.

Figure 2. Text tokenization and word corpus



Findability Feature Extraction

After building the summary for each document, the features are extracted for each document. However, the feature extraction is done on the summary generated in the previous step rather than the entire document. Through this step, each document is represented by a vector of length p where the components are the findability features listed in Table 1 along with the mathematical representation of the features.

$$QueryFrequency = \sum_{s=1}^N q_s, \text{ where } q \text{ is number of query words, } s \text{ is sentences in the summary}$$

$$SumSent = \sum_{s=1}^N s, \text{ where } s \text{ is number of sentences in the summary}$$

$$SumWord = \sum_{s=1}^N w_s, \text{ where } w \text{ is number of words in each } s \text{ sentence in the summary}$$

$$SumSentQt = \sum_{s=1}^N \sum_{i=1}^t q_{i,s}, \text{ where } q \text{ is the } i \text{th query term, in the } s \text{th sentence in the summary}$$

Modeling

In the text processing and feature engineering phase, document summary and word corpus were derived. Now, each document is represented by a set of features and its contents in a tokenized text format. However, words as a string of characters could neither be understood by a machine nor be a suitable input for a machine learning algorithm. Therefore, the NLP is used to turn text into a matrix and make it comprehensible to the machine. Using the Word2Vec model, each word is represented as a vector. The Word2Vec model considers the context and the semantic meaning of the words. So if two words are close in meaning they will be located close in the space (Mikolov, 2013). With the transformation of words to vectors, the machine-learning algorithms can perform algebra operations on them.

Vector representation of words

To build a vector representation for each word of the corpus, the Word2Vec model using the Gensim library in Python is implemented. The input to the Word2Vec model is the tokenized texts from TREC-9 documents (Refer Figure 3). The Word2Vec model is trained on the .txt files separately and for each document, a vector representation for each word is obtained. There are possibilities

Table 1. Findability features

Findability feature	Description
QueryFrequency	The frequency of query words in the summary
SumSent	The number of sentences in the summary
SumWord	The number of words in the summary
SumSentQt	The number of sentences that contain at least one query word in the summary
MinWQSum	Minimum position of query word in the summary
MaxWQSum	Maximum position of query word in the summary

that a word in two different documents will have two different representations. It is assumed that a word will appear in different contexts and to build a general representation for a given word, the model has to be presented with a huge data. As such, the similarity between query terms and each document's words is calculated using the vector representation generated after training the model on the document in question.

Finally, the output of the Word2Vec implementation is a matrix where each row is a vector representation of a word. If two words appear in the same context or they have a similar meaning, they will have almost the same representation. After training the model on the corpus for each document, a matrix M of similarity between each term in the query and each word in the document's corpus given by the Word2Vec model is generated. For a given query Q that contains p words and a given document D where its corpus of words contains n tokens, we define the similarity matrix as:

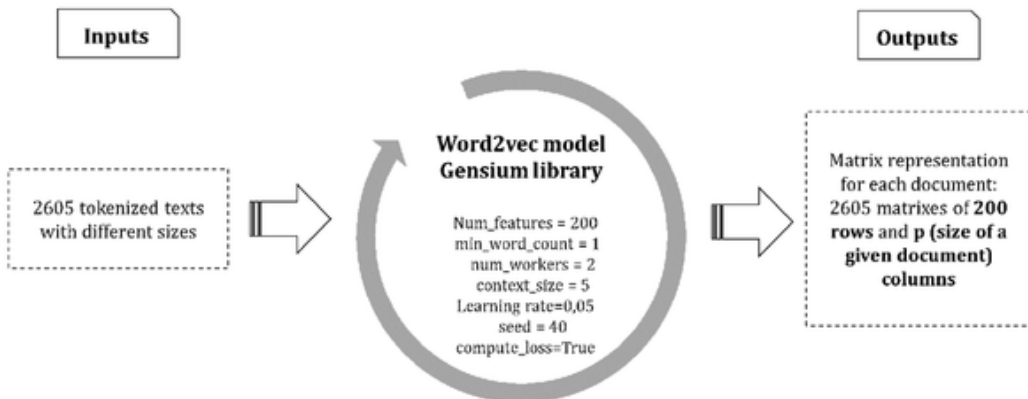
$$M_{ij} = \text{Similarity}(QT_{i1}DW_j)$$

$$\text{Score}(Q, D) = \begin{bmatrix} m_{11} & \cdots & m_{1n} \\ \vdots & \ddots & \vdots \\ m_{p1} & \cdots & m_{pn} \end{bmatrix}$$

The similarity between two words is defined as the Cosine between their vector representations. The cosine ranged between 1 and -1. So the closer they are in the meaning and their context of appearance, the cosines of their vector representation converge to 1 (Refer Figure 4). The similarity could take negative values that we can interpret as opposing words, which is different from dissimilarity presented by a cosine of 0 which corresponds to two perpendicular words. The final score that links between document's word and a given query for each document, DW_j , where j in $[1 ; n]$, m_{ij} is similarity matrix and p is the number of words.

$$\text{Score}(DW_j) = \frac{\sum_{i=1}^p |m_{ij}|}{p}$$

Figure 3. Word2Vec implementation process



It is assumed that a negative similarity closer to -1 is much better than a null similarity. Therefore, the weighted average of the absolute values of similarity scores is used. Finally, each document D_k is represented by a vector that contains scores of the document's words and query terms.

$$D_k = \left(\text{Score}(DW_j) \right)_{1 \leq j \leq \text{size}(D_k)}$$

Clustering using k-means

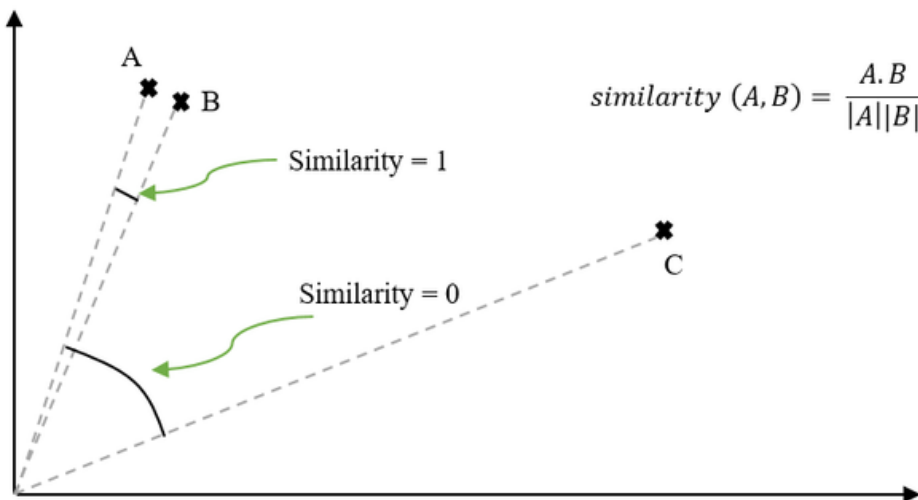
Each document is now in the form of a vector, which contains similarity scores between query words and the document's summary. This vector will be used along with findability features in the clustering model to discover document effort. However, the dimension of the vectors is not the same due to the differences in document length. In other words, depending on the length of the summary, the dimension vector will also vary. For example, longer summaries will result in increased vector dimension. To overcome inconsistencies, similarity variances and likelihood were used. The length of the document is important, given that the user almost reads just the summary or the abstract of a document and sentences containing query words. The variance as a score is meaningful compared to vector scores since some words have higher similarities than others.

The Euclidean Distance measure in the k-means algorithm to find the closest cluster of a document makes the algorithm sensitive to magnitudes. Hence, feature scaling will help to weigh all the features equally. Formally, if a feature in the data set is bigger in scale compared to others, then this feature dominates Euclidean Distance. Therefore, the features are normalized using the formula below.

$$z = \frac{x - \mu}{\sigma}$$

Where: z : The normalized observation x : Original feature values μ : The mean of all values taken by a feature σ : Standard deviation

Figure 4. Vector representation and similarity between words



The KMeans class from scikit-learn’s cluster module is used to implement the k-means clustering. For k-means, the number of clusters should be specified by the user and it’s not always obvious to find the optimal number of clusters. With the usage of the Elbow method, the optimal k value can be determined. Figure 5 shows the graphical presentation to estimate the optimal number of clusters, k for a given task. The plot looks like an arm and the ‘elbow of the arm’ is the optimal k value.

Intuitively, if k increases, the within-cluster SSE (“distortion”) will decrease. This is because the samples will be closer to the centroids they are assigned to. The elbow method applied for all queries showed that the optimal number of clusters is 3. With the optimal k value, the clustering is performed to determine the effort needed to identify relevant information in documents according to queries.

RESULTS AND DISCUSSION

The k-means clustering model produced the graphs in Figure 7 showing the clustering of documents into 3 clusters for each query. However, each cluster needs to be assigned to the right level of effort. To determine the level of effort for each of the clusters, the centroids of each cluster are compared.

To compare the clusters, the average centroid values of each cluster are shown in Figure 6. Besides the findability features, the standard deviation and likelihood values are also included in the table. Cluster 0 has a high likelihood score which means there is a high similarity between the query words and document words. This cluster also has a small standard deviation value whereby all the documents within this cluster are very close to the centroid. The low standard deviation could also mean all the documents in Cluster 0 have similar findability feature values. Looking at the average values of all the findability feature values, it can be observed that the number of sentences in the summary (SumSent) is approximately 20 sentences, while the number of words in the summary (SumWord) is about 484 words. Therefore, based on these two average findability feature values, Cluster 0 indicates a group of documents that require less effort from the user to find relevant information.

In contrast, the likelihood score for Cluster 1 is 0.25 which indicates low similarity between the query words and document words. The standard deviation for this cluster is 0.08, somewhat moderately deviating from the centroid of Cluster 1 compared to the other two clusters. On average, the summary of documents in this cluster contains approximately 22 sentences (SumSent) and the

Figure 5. Elbow plot estimation of the optimal number of clusters

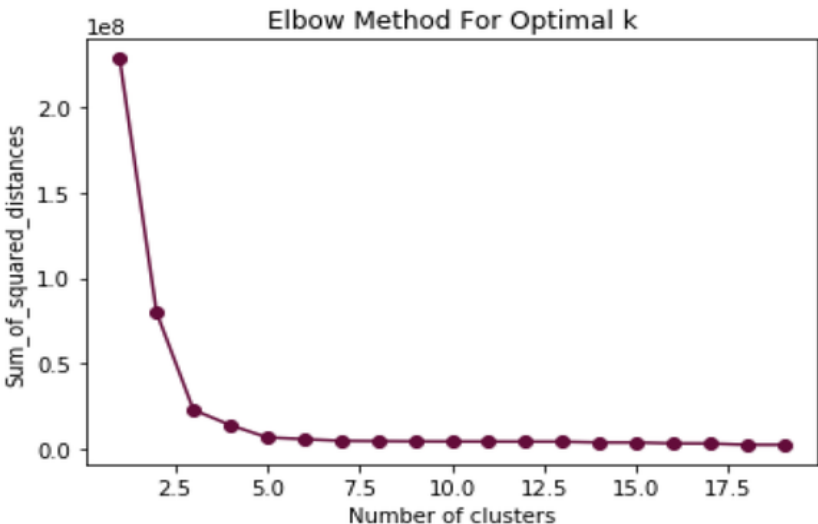


Figure 6. k-means average centroid values for 3 clusters

	QueryFrequency	SumSent	SumWord	SumSentQt	NbImage	NbTable	MinWQSum	MaxWQSum	SrtDv	Likelihood
0	0.025791	19.538371	484.319949	9.439742	3.265237	0.743535	134.571890	314.070986	0.048150	0.628394
1	0.036745	22.214577	419.741658	12.661798	4.447300	0.358645	50.864903	289.376977	0.081564	0.251889
2	0.029024	105.112868	2119.206715	52.810337	5.054466	0.281973	95.906748	1749.595223	0.105989	0.556737

number of words (SumWord) in the summary is about 420 words. In comparison to Cluster 0, Cluster 1 does appear to have similar results with regards to the number of sentences and number of words in the summary. Although the number of sentences between both these clusters is very close, the number of words in the summary does vary largely.

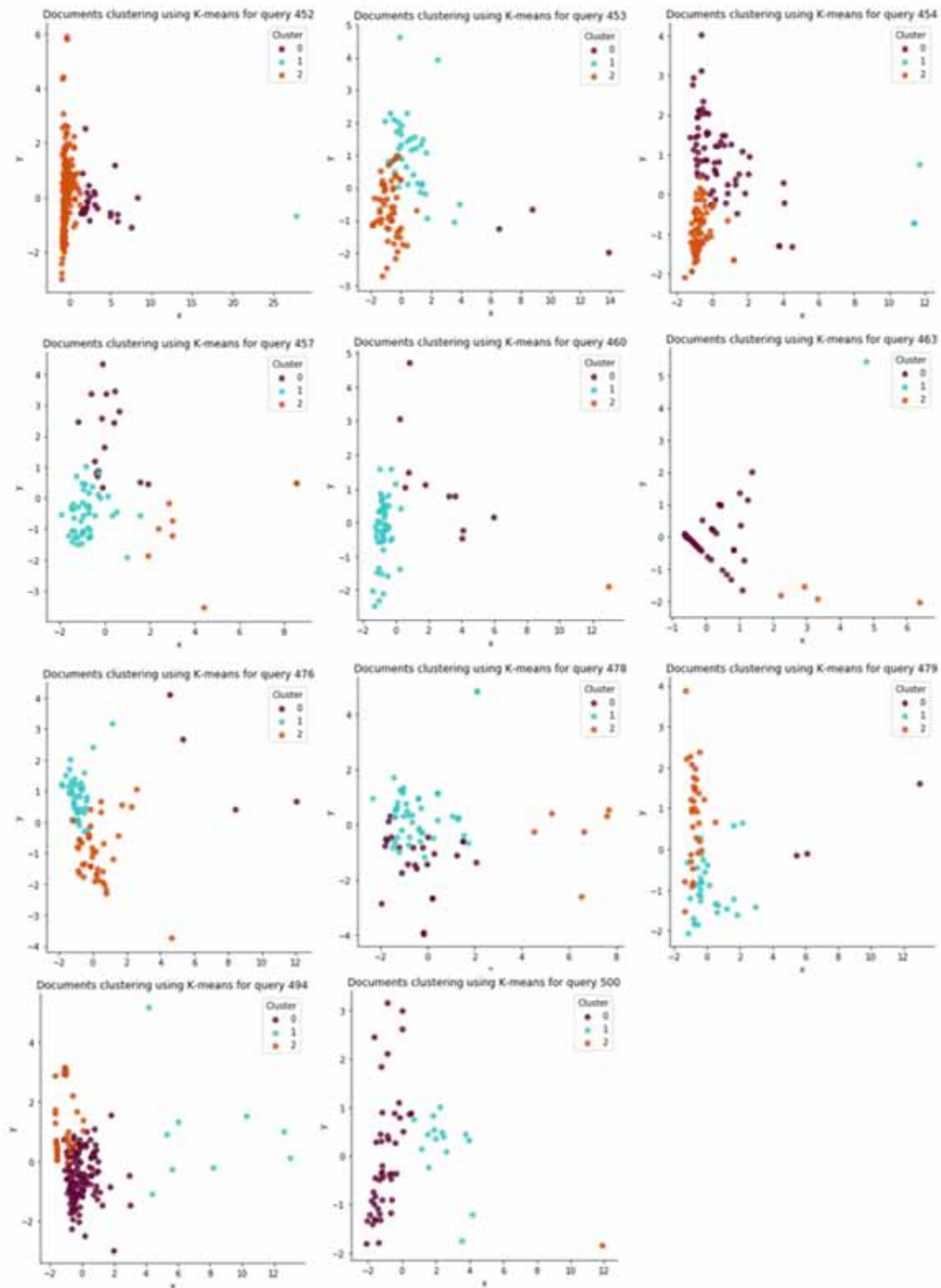
As for Cluster 2, the likelihood value is 0.57 which shows a high level of similarities between the query words and the words in the documents. This cluster also has the highest amount of standard deviation of 0.1 compared to Cluster 0 and Cluster 1. Consequently, the number of sentences in the summary (SumSent) of documents in this cluster is also high, on average 105 sentences. Similarly, the number of words (SumWord) in the summary is more than 2000 words which implies high effort is needed to read the long content of the document. Nevertheless, the number of sentences that contain at least one query word is 53 and suggests that in every 2 sentences, there will be at least one query word. Despite the need to put more effort into reading lengthy documents, the user will be able to find relevant information in the document.

For Cluster 0, the number of sentences that contain at least one query term (SumSentQt) is at least 9 sentences, while for Cluster 1 it is 13 sentences. Hence, for Cluster 0 it is possible for users to find a query term in every 2 sentences and for Cluster 1 in about every 1.5 sentences there is at least 1 query term. It appears that there are no apparent differences in the number of query terms that are found in these documents, regardless of them being in different clusters and having different levels of findability effort. The QueryFrequency feature which measures the frequency of query words in the summary reinforces the observation of seeing query terms per total number of sentences in the summary.

The other two features, MinWQSum and MaxWQSum measure the minimum position of query words in the summary and maximum position of query words in the summary respectively. The MinWQSum and MaxWQSum for Cluster 0 are 135 and 314, for Cluster 1 are 51 and 289, and for Cluster 2 are 96 and 1750 respectively. From this observation, a user will be able to find relevant information earlier in the documents from Cluster 1 but much later from the documents in Cluster 0 because the first query word in the summary is found at position 135. Comparing Cluster 1 and 2, it appears that query term appears earlier for documents in Cluster 2 compared to Cluster 1. Although the summary for documents in Cluster 2 is very long, the first occurrence of the query term happens early. Earlier occurrences of query terms could benefit users as it enables them to find relevant information faster but the length of the summary could inflict a drawback for some users who do not prefer long text.

Nevertheless, with these interpretations, Cluster 1 seems to have documents that require the lowest effort in finding relevant information due to the small number of summary sentences and words while having high occurrences of query terms in the summary sentences. Besides that, the first query term also appears earlier in the text and the last occurrence of the query term is within the next 238 words. On the other hand, Cluster 2 has higher summary sentences and words despite having the query terms occurring quite frequently in the summary. Even though the first query term appears quite early in the summary text, the user would only be able to consume all relevant information concerning the query terms if almost all the content of the summary is read. Such numbers of the findability features indicate the high effort needed by the end-user. Finally, the documents in Cluster 0 have low sentence numbers but a moderately high number of words. In addition, the first occurrence of query term appears later

Figure 7. k-means clustering (k=3) of the relevant documents retrieved for 11 different queries



but the final query term emerges in the next 180 words. Despite the late occurrence of a query term, the summary of these documents is rather short with frequent appearances of query terms. Hence, the documents' in Cluster 0 could require a moderate amount of effort in finding relevant information.

In this result, the k-means algorithm was used with findability features to uncover hidden efforts for finding relevant information in relevant documents. Using the centroid values of each cluster, the documents could be grouped in terms of findability effort and thus achieving the first objective of this study.

EVALUATION

The original input files from TREC contain up to 1000 documents per query for each system run. With the relevance judgment, the relevancies of these documents can be determined. During that process, some documents would appear relevant or irrelevant to the respective query. However, when it comes to measuring the effort needed by end-users to find relevant information, it is only right to ensure effort is considered for relevant documents (Yilmaz, 2014). It is pointless for effort measurement in irrelevant documents as a user will not be able to gain any knowledge from them.

The findability effort is now obtained through clustering as discussed in the previous section. Using these effort details, the documents can be further classified as relevant and not relevant to evaluate the retrieval systems. A low effort document is highly preferred by end-users compared to high effort documents. Consequently, documents from Cluster 1 are translated as relevant documents, and those from Cluster 2 are interpreted as irrelevant documents. Based on the analysis of results, the documents within Cluster 0 require moderate effort. Hence, similar to the translation done on the original ternary relevance, where 'relevant' and 'highly relevant' documents were considered as relevant, the documents from Cluster 0 will also be classified as relevant.

With this information, a new relevance judgment can be created containing the relevancies of documents based on the amount of findability effort. Using this new relevance judgment, the systems' performance can be evaluated using the average precision and mean average precision. Instead of using the original input files, only the initially marked relevant documents were retained in the new input files. Such an approach was taken to check if the original relevant documents were a high or low effort. In a scenario where all the original relevant documents are low effort, the average precision score would be 1. If there is a high number of documents with high effort, the average precision score will be closer to 0 and possibly 0 if all the documents were a high effort.

The average precision (AP) and mean average precision (MAP) scores were calculated for all the systems and topics in TREC-9 with the newly generated input files and relevance judgment. Some of the average precision scores are shown in Figure 8. The first row of the table represents the query number (the original queries are numbers 451 – 500) and the first column represents the systems.

From Figure 8, it can be observed that for some queries, more systems were able to retrieve low effort documents as the scores are 1. Specifically, most systems have high AP scores for Query5 and somewhat similar for Query4. The relevant documents for these queries were highly likely to have short summaries with query terms appearing early in the text. Besides that, there could be possibilities that these are simple topics such that it is easy to find relevant documents. Other than that, Query1 shows low AP scores which means these relevant documents appear to require high effort in finding relevant information. Hence, more documents were now categorized as not relevant due to the high amount of effort needed to find relevant information. The mean average precision was also calculated for each system. On average, the values deviate from the baseline by 23%. This is likely because, with the clustering approach in identifying findability effort, some documents have become non-relevant. When effort is considered for the evaluation retrieval systems, the performance of the retrieval systems has been impacted. With that, the second objective of this paper, to demonstrate changes in information retrieval systems' performance when the effort is included in evaluation is met.

Figure 8. Average precision for each query using findability effort based relevance judgment

Systems	Queries											
		0	1	2	3	4	5	6	7	8	9	10
	0	0.988070	0.426814	0.488400	0.835850	0.833423	0.883078	0.587118	0.914457	0.596401	0.792613	0.569167
	1	0.992954	0.369210	0.433208	0.807103	0.771225	1.000000	0.432644	0.649503	0.677049	0.777503	0.524068
	2	0.953171	0.539034	0.489815	0.806508	0.779541	0.796717	0.595359	0.808453	0.679598	0.816247	0.631673
	3	0.993035	0.380427	0.450323	0.850327	0.835566	1.000000	0.443832	0.635714	0.639288	0.791046	0.523474
	4	0.985968	0.544486	0.525362	0.861934	0.821357	0.832340	0.563560	0.798619	0.642885	0.774613	0.375673
	5	0.947666	0.539941	0.510091	0.846379	0.813741	0.825263	0.576792	0.798718	0.654757	0.797447	0.591086
	6	1.000000	0.391911	0.458631	0.923101	1.000000	1.000000	0.582757	0.945036	0.629071	0.819813	0.475397
	7	1.000000	0.424843	0.522949	0.883111	1.000000	1.000000	0.581240	0.741345	0.721410	0.851084	0.436863
	8	0.998182	0.247260	0.480234	0.903171	1.000000	1.000000	0.584585	0.805190	0.632374	0.681967	0.625000
	9	1.000000	0.401503	0.484384	0.920013	1.000000	1.000000	0.582757	0.938937	0.629071	0.824279	0.588508
	10	1.000000	0.374994	0.475088	0.913085	1.000000	1.000000	0.574459	0.933485	0.632374	0.770239	0.791667
	11	0.998992	0.374994	0.467543	0.913412	1.000000	1.000000	0.584074	0.936020	0.632077	0.778314	0.791667
	12	0.986070	0.426814	0.488400	0.835850	0.833423	0.883078	0.587118	0.914457	0.596401	0.792613	0.569167
	13	0.982444	0.468356	0.593972	0.987495	0.892762	1.000000	0.676611	0.793432	0.674624	0.865513	0.691667
	14	0.878421	0.777782	0.591342	0.945733	0.781872	1.000000	0.788398	0.837614	0.654362	0.839390	0.916667
15	0.780516	0.765288	0.686867	0.793023	0.869172	1.000000	0.841728	0.782967	0.836584	0.847532	0.835000	
16	0.780516	0.765288	0.686867	0.793023	0.869172	1.000000	0.841728	0.782967	0.836584	0.847532	0.835000	

A past study (Rajagopal & Ravana, 2019) used the same dataset to measure the correlation coefficient between the system rankings of the original and effort based relevance judgments but with reduced topic sizes (ie: 40 instead of 50 topics). The AP@1000 and MAP were calculated and system rankings were compared (Rajagopal & Ravana, 2019). The Kendall's Tau was varying by 17% on average from the original when using the effort based relevance judgment. The difference from the past study (Rajagopal & Ravana, 2019) is lower than what is achieved from this research which showed higher deviation from baseline. Thus, indicating better performance in retrieving or identifying documents with low effort. In another study (Wang, 2021) which also utilized k-means for feedback embeddings in measuring the effectiveness of information retrieval show that MAP can be improved by up to 26% on the TREC 2019 and 10% on the TREC 2020 datasets. In extending the use of k-means clustering to non-English language, the study attempted to cluster documents to improve user experience in finding relevant documents (Aliwy et al., 2022). The study also claimed that the k-means algorithm is as good as Ward's clustering and better than average agglomerative clustering (Aliwy et al., 2022). Though not equally comparable with the past studies, the changes in average precision calculation due to inclusion of effort shows variation in information retrieval evaluation that shouldn't be disregarded.

CONCLUSION

The concept of relevance sits at the core of information retrieval but the utility of a document to an actual user is equally important. In this paper, the k-means clustering technique was implemented to group documents based on the various findability features. Subsequently, classifying these documents as relevant or irrelevant based on the amount of effort needed to find relevant information. The clusters were distinguishable in this study and accordingly the effort was derived from the centroid values. It was also apparent that the performance of the retrieval systems changed when findability effort was incorporated with relevance during system-based evaluation. Similarly, studies tackling different

aspects of information retrieval have highlighted the ability of the k-means clustering to improve the performance of the information retrieval systems (Aliwy et al., 2022; Wang, 2021).

Although an older data set was used for this study, it does indicate positive directions for clustering documents without human intervention in discovering findability effort. Seeking effort grading from human assessors, besides the existing human relevance judgments will be an added cost and time. As such, an alternative approach presented in this work could reduce the cost and time generating effort-based relevance judgment. This study could be further expanded to determine if easy topics tend to have low effort documents, which could impact the evaluation of information retrieval systems. Besides that, this study can be expanded to analyze graded relevancies with effort and the incorporation of k-means algorithm variations.

REFERENCES

- Al-Maskari, A. (2008). The good and the bad system: does the test collection predict users' effectiveness? *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. doi:10.1145/1390334.1390347
- Aliwy, Aljanabi, & Alameen. (2022). Arabic text clustering technique to improve information retrieval. In *AIP Conference Proceedings*. AIP Publishing LLC.
- Allan, J., Carterette, B., & Lewis, J. (2005). When will information retrieval be "good enough"? *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. doi:10.1145/1076034.1076109
- Balakrishnan & Lloyd-Yemoh. (2014). *Stemming and lemmatization: a comparison of retrieval performances*. Academic Press.
- Boban, I., Doko, A., & Gotovac, S. (2020). Sentence Retrieval using Stemming and Lemmatization with Different Length of the Queries. *Adv. Sci. Technol. Eng. Syst. J.*, 5(3), 349–354. doi:10.25046/aj050345
- Braune, C., Besecke, S., & Kruse, R. (2015). Density based clustering: alternatives to DBSCAN. In *Partitionial Clustering Algorithms* (pp. 193–213). Springer.
- Cleverdon, C. W. (1991). The significance of the Cranfield tests on index languages. *Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval*. doi:10.1145/122860.122861
- Clinchant, S., & Perronin, F. (2013). Aggregating continuous word embeddings for information retrieval. *Proceedings of the workshop on continuous vector space models and their compositionality*.
- Clough & Sanderson. (2013). *Evaluating the performance of information retrieval systems using test collections*. Academic Press.
- Ezugwu, A. E. (2020). Nature-inspired metaheuristic techniques for automatic clustering: A survey and performance study. *SN Applied Sciences*, 2(2), 273. doi:10.1007/s42452-020-2073-0
- Ezugwu, A. E., Ikotun, A. M., Oyelade, O. O., Abualigah, L., Agushaka, J. O., Eke, C. I., & Akinyelu, A. A. (2022). A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects. *Engineering Applications of Artificial Intelligence*, 110, 104743. doi:10.1016/j.engappai.2022.104743
- Gopala Rao, V., & Bhanu Prasad, S. (2013). Space and cosine similarity measures for text document clustering. *International Journal of Engineering Research & Technology (Ahmedabad)*, 2(2), 1–5.
- Han, J., Pei, J., & Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.
- Kalaivendhan & Sumathi. (2014). An efficient clustering method to find similarity between the documents. *Int. J. Innov. Res. Comput. Commun. Eng.*, 1.
- Kanavos, A., Iakovou, S., Sioutas, S., & Tampakas, V. (2018). Large Scale Product Recommendation of Supermarket Ware Based on Customer Behaviour Analysis. *Big Data and Cognitive Computing*, 2(2), 11. doi:10.3390/bdcc2020011
- Karthika, N., & Janet, B. (2020). *Clustering Performance Analysis*. Springer Singapore. doi:10.1007/978-981-15-1081-6_3
- Latifi-Pakdehi, A., & Daneshpour, N. (2021). DBHC: A DBSCAN-based hierarchical clustering algorithm. *Data & Knowledge Engineering*, 135, 101922. doi:10.1016/j.datak.2021.101922
- Lilleberg, J., Zhu, Y., & Zhang, Y. (2015). Support vector machines and word2vec for text classification with semantic features. In *2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC)*. IEEE. doi:10.1109/ICCI-CC.2015.7259377
- Lin, Y.-S., Jiang, J.-Y., & Lee, S.-J. (2013). A similarity measure for text classification and clustering. *IEEE Transactions on Knowledge and Data Engineering*, 26(7), 1575–1590. doi:10.1109/TKDE.2013.19

- Maher, K., & Joshi, M. S. (2016). Effectiveness of different similarity measures for text classification and clustering. *IJCSIT*, 7, 1715–1720.
- Mikolov, T. (2013). *Efficient estimation of word representations in vector space*. arXiv preprint arXiv:1301.3781.
- Natural Language Processing. (2020). https://en.wikipedia.org/wiki/Natural_language_processing
- Paalman, J. (2019). Term based semantic clusters for very short text classification. *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*. doi:10.26615/978-954-452-056-4_102
- Park, T. K. (1994). Toward a theory of user-based relevance: A call for a new paradigm of inquiry. *Journal of the American Society for Information Science*, 45(3), 135–141. doi:10.1002/(SICI)1097-4571(199404)45:3<135::AID-ASIS3>3.0.CO;2-1
- Rahmawati, D., & Khodra, M. L. (2016). Word2vec semantic representation in multilabel classification for Indonesian news article. In *2016 International Conference On Advanced Informatics: Concepts, Theory And Application (ICAICTA)*. IEEE. doi:10.1109/ICAICTA.2016.7803115
- Rajagopal, P., & Ravana, S. D. (2019). Effort-based information retrieval evaluation with varied evaluation depth and topic sizes. *Proceedings of the 3rd International Conference on Business and Information Management*. doi:10.1145/3361785.3361794
- Saracevic, T. (1975). Relevance: A review of and a framework for the thinking on the notion in information science. *Journal of the American Society for Information Science*, 26(6), 321–343. doi:10.1002/asi.4630260604
- Sindhiya & Tajunisha. (2014). Concept and term based similarity measure for text classification and clustering. *Int. J. Eng. Res. Sci. Tech.*
- Singh, A., Yadav, A., & Rana, A. (2013). K-means with Three different Distance Metrics. *International Journal of Computers and Applications*, 67(10).
- Singh, M., Bansal, D., & Sofat, S. (2016). Behavioral analysis and classification of spammers distributing pornographic content in social media. *Social Network Analysis and Mining*, 6(1), 41. doi:10.1007/s13278-016-0350-0
- Tarczynski, T. (2011). Document Clustering - Concepts, Metrics and Algorithms. *International Journal of Electronics and Telecommunications*, 57(3), 271–277. doi:10.2478/v10177-011-0036-5
- Turpin, A. H., & Hersh, W. (2001). Why batch and user evaluations do not give the same results. *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. doi:10.1145/383952.383992
- Unsupervised Learning. (2020). https://en.wikipedia.org/wiki/Unsupervised_learning
- Van Nguyen, T. (2017). Combining word2vec with revised vector space model for better code retrieval. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*. IEEE. doi:10.1109/ICSE-C.2017.90
- Verma, M., Yilmaz, E., & Craswell, N. (2016). On obtaining effort based judgements for information retrieval. *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. doi:10.1145/2835776.2835840
- Villa, R., & Halvey, M. (2013). Is relevance hard work? Evaluating the effort of making relevant assessments. *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*.
- Wang, X. (2021). Pseudo-relevance feedback for multiple representation dense retrieval. *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*. doi:10.1145/3471158.3472250
- Word2Vec. (2020). <https://en.wikipedia.org/wiki/Word2vec>
- Wu, S. (2019). Sentiment Analysis Method Based on Kmeans and Online Transfer Learning. *Computers, Materials, & Continua*, 60(3), 1207–1222.

Yilmaz, E. (2014). Relevance and effort: An analysis of document utility. *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. doi:10.1145/2661829.2661953

Zhang, Y., Lu, J., Liu, F., Liu, Q., Porter, A., Chen, H., & Zhang, G. (2018). Does deep learning help topic extraction? A kernel k-means clustering method with word embedding. *Journal of Informetrics*, 12(4), 1099–1117. doi:10.1016/j.joi.2018.09.004

Zhang, Z., Zhang, J., & Xue, H. (2008). *Improved K-means clustering algorithm*. In 2008 Congress on Image and Signal Processing. IEEE.

Zuccon, G. (2015). Integrating and evaluating neural word embeddings in information retrieval. *Proceedings of the 20th Australasian document computing symposium*. doi:10.1145/2838931.2838936

Prabha Rajagopal is a lecturer at the School of Information Technology (SoIT), Monash University. She received her BEng (Hons) in Electronics from Multimedia University, Malaysia. She obtained her Master of Computer Science in 2014 and Doctor of Philosophy (PhD) in Computer Science in 2018, both from the University of Malaya under the scholarships of MyBrain, Malaysia. Prabha has worked in multinational companies in the telecommunication industry from the year 2005 to 2012. Then, she was associated with the University of Malaya as a researcher. Her current research interest is in information retrieval, machine learning and sustainable communities.

Taoufik Aghris completed his Master's degree in Data Science from École Polytechnique, and graduated from EMINES - School of Industrial Management / Mohammed VI Polytechnic University, Morocco. Currently, he is a Data Scientist Intern @ BNP Paribas CIB, France.

Fatima Ezzahra is an Industrial Management Engineer graduated from EMINES - School of Industrial Management. Currently, she is working as a Modeling and Analytics consultant.

Sri Devi Ravana received the bachelor's degree in information technology from the National University of Malaysia, in 2000, the master's degree in software engineering from the University of Malaya, Malaysia, in 2001, and the Ph.D. degree from the Department of Computer Science and Software Engineering, The University of Melbourne, Australia, in 2012. She is currently an Associate Professor at the Department of Information Systems, University of Malaya. Her research interests include information retrieval heuristics, text indexing and data analytics. She received a couple of best paper awards in international conferences within the area of information retrieval. She is also the recipient of the competitive AUA Scholars Award 2019-2020 - Host Institution is The National University of Singapore. Dr. Sri Devi Ravana actively collaborated with researchers from many countries including the University of Auckland, University of Melbourne, University of Leeds, The University of Hong Kong and National University of Singapore. She also serves as Editorial Board Member in a few journals. She is also an active organizing committee member and reviewer for many conferences.