Ministry of Higher Education and Scientific Research

University of Kasdi Merbah Ouargla

Faculty of New Technologies of Information and Communication

Department of Computer Science and Information Technologies

Memory ACADEMIC MASTER

Theme :

Co-occurence based query expansion

Publicly supported ***The:09/2020***

In front of the jury:

Mr **MEZATTI Messaoud**	***President***	***UKM Ouargla***
Mr **KHALDI Amine**	***Examiner***	***UKM Ouargla***
Mr **BEKKARI Fouad**	***Supervising***	***UKM Ouargla***

Presented by :

- **DIAR Fatima zohra**
- **KHAMRA Souhila**

Year Universitv: ***2019/2020***

Abstract

Abstract

The Co-occurrence of words has been widely used in addressing the problem of query expansion, and the challenge remains is how to obtain the best way to express the interconnectedness of words through their Co-occurrence. In this work we propose a method based on the idea of weighting words in documents and we test it to obtain a best evaluation of the idea.

Key words

Information retrieval, Information Retrieval system, Query Expansion, Co-occurrence.

ملخص

الظهور المشترك للكلمات استعمل على نطاق واسع في معالجة مشكل تمديد الاستعلام ويبقى

لتحدي هو كيفية الحصول على الطريقة المثلى للتعبير عن ترابط الكلمات من خلال ظهورها المتبادل

في هذا العمل نقترح طريقة مبنية على فكرة تثقيل الكلمات في الوثائق و نقوم باختبارها للحصول على تقييم امثل للفكرة

الكلمات الدالة

استرجاع المعلومات، نظام استرجاع المعلومات، توسيع الاستعلام، التواجد المشترك

List of contents

List of contents

List of illustrations

General introduction:

General introduction:

The advent of the Internet has made it accessible to a variety of services, such as email, instant messaging and the World Wide Web. These changes have brought about a profound change in the means of communication, in particular by facilitating the exchange of documents between countries. Since then, the document sets have been enriched with trillions documents written and published.

Query Expansion is an approach often used to retrieve information To help the search form to better identify related documents. The success of this technique depends on the correct choice of extension terms and how to add these new terms, if new keywords are introduced in an expansion approach Unrelated to the need for information and the ability to find documents The associated decrease, which has a negative impact on recall and accuracy. For this reason Quality control of expansion conditions is an essential step in expanding Request. However, a good selection of expansion conditions is not sufficient to guarantee Expansion successful if these terms are not properly incorporated in to the query.

Universal query extension techniques have always been proposed as a solution to overcome the problem of terminological mismatch between the query and its associated documents. You need to use a method that automatically handles search problems for the best terms to develop a particular query, and second, how those terms are weighted to be used with the original query.

In this work, we have chosen to use the co-occurrence. Long-term co-occurrence data have been widely used in document search systems for identifying indexing terms similar to those that have been specified in a user request: these similar terms can then be used to augment the original query statement. That’s why we used this approach for query expansion. In these notes we use iteration to address the problem of the query extension, which has proven effective in this context. Our memory is organized into three chapters, beginning with the general introduction and ending with a general conclusion.

The three chapters are titled respectively and detailed as follows:

- The first chapter aims to present the field of IR. In the first part, we present the basic concepts of IR. In particular, we start by defining the IR, then we describe the concepts of document, query and relevance and what an IRS is, in the second part we are interested in the indexing, search and reformulation processes requests; In the third part we cite the IR models. And at the end the last part of this chapter is discussed the difficulties of IR.

- In Chapter Two, we present below the notion of query expansion and then discuss the two most important techniques of query reformulation, and we have clarified what the notion of co-occurrence.

- Finally, in the third chapter we present our experimental environment: the tools used, the stages of implementation of the system, we present the experimental results obtained, as well as the analysis and evaluation of these results.

At the conclusion, we will present the main points of this work and some views that may arise from it.

Problem

The idea of Automatic Query Expansion (or Modification) is to get other additional words through a corpus based on co-occurrence, to extend the initial query to whatever satisfies the user's need.

The problem to be solved in this job is how to use co-occurrence to do query expansion.

1 Chapter I: **State of the art**

1.1Introduction:

Today information plays an important role in the daily life of individuals and companies. However, the development of the Internet and the widespread use of computers in all fields have provided an unprecedented amount of information. In fact, the amount of information available, especially on the Internet, is measured in billions of pages.

As a result, it becomes more and more difficult to determine exactly what you are looking for in this massive amount of information. Information research (IR) is an area of expertise that seeks to meet these expectations.

Information retrieval can be defined as a target activity it is to locate and present a set of documents to the user based on need information. The challenge is to be able, among the large size of documents available, find those that best meet user expectations.

This chapter aims to present the field of IR. In the first part, we present the basic concepts of IR. In particular, we start by defining the IR, then we describe the concepts of document, query and relevance and what an IRS is, in the second part we are interested in the indexing, search and reformulation processes requests; In the third part we cite the IR models. And at the end the last part of this chapter is discussed the difficulties of IR.

1.2Basic concepts in information retrieval:

1.2.1 -Information retrieval (IR):

Several definitions of information retrieval have emerged in these years we cite in this context the three definitions following:

-Definition 1: The research is an activity the purpose of which is to locate and deliver documentary granules to a user according to his need for information.

-Definition 2: Information retrieval is a branch of computing who is interested in acquisition, organization, storage, research and selection of information.

-Definition 3: Information retrieval is a research discipline which incorporates models and techniques whose purpose is to facilitate access to information relevant for a user with a need for information [01].

1.2.2 Document and collection of documents:

In an information retrieval system, the document is an essential element, one of which is considered to be the physical medium of the information, which can be text, an image, a video sequence, a web page.

The collection of documents called documentary base or corpus is the set of documents processed by an IRS.

In the case of a text document, it can be represented according to three views [02]:

The semantic (or content) point of view: it is interested in the information presented in the document.

The logical view: it focuses on the logical structure of the document (structuring in chapters, sections).

The presentation view: it consists of the presentation on a support for two dimensions (paragraph alignment, indentation, headers and footers, etc.).

1.2.3 Relevance:

The definition of relevance in the field of IR is not simple, it can mean the correspondence between the document and the request or a measure of informativeness of the document to the request.

There are two types of relevance: system relevance and relevance user

System relevance: works with objectivity and determination, its way of doing things is to assess the adequacy of the content of the documents vis-à-vis that of the request by a score assigned by the IRS.

User relevance: works with a subjectivity (that two users who have different interests can judge the same response provided by the IRS has a request), its way of doing is judged the relevance of the user on the documents provided by the IRS in response to a request.

In addition, [02] a document deemed irrelevant at time t for a request may be considered relevant at time t + 1, because the user's knowledge of the subject has evolved, it is evolving.

Need for information and request:

The need for information of the user is expressed by a request, the treatment of this last by SRI does not always give the result that the user hopes to have.

This is due, on the one hand, to the fact that the user has a restricted vision of the documents available in the collection and ignores the internal functioning of the IRS.

On the other hand, the IRS has no knowledge of its users (centers interests, levels ...).

This bias between the request and the information need is One of the major difficulties of any IRS is the bias between the need for information and the request and for this we integrate a mechanism for reformulating requests in the IRS.

1.3 Information Retrieval System

The role of an Information Retrieval System (IRS) is to implement techniques and means for returning relevant documents from a collection in response to a need for information from a user, expressed by a query language which can be natural language, a list of keywords or a Boolean language **[02].**

1.3.1 Process of IR System

For this, we must follow a process of indexing the documents of the collection which constructs a synthetic representation of the documents, which is called the index.

Once the user writes his request a process is performed on it at the same time. The purpose of this process is to analyze the request and establish an internal representation, we continue the system but implement a link between the representation of the request and the representation of the documents (index), so that the user needs (relevant documents) will be best satisfied.

The link or proper correspondence between the document and the request is established by IR models, this diagram shows the architecture of an SRI.

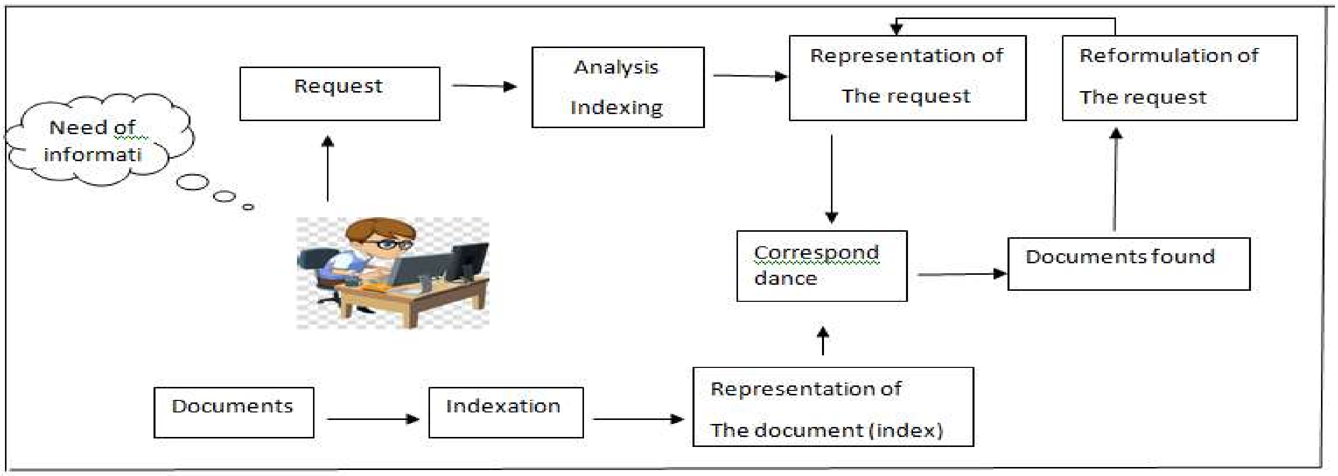


Figure 1. General architecture of the information system

1.3.2 The indexing process:

Indexing is a fundamental operation on the document of the collection to have acceptable costs in order to achieve the IR [03].

This operation uses descriptors which are a list of keywords associated with each document to best represent the semantic content of the documents.

Indexing is to give a synthetic representation of documents, by terms, extracting these latter facts in three ways:

Manual:

It is specialists in the field who analyze each document in the collection documents that respond to user requests will be returned by the IRS with better precision [04].

With this advantage offered by manual indexing, there are the disadvantages of wasted time and the large number of people (specialists). In addition, a document can be indexed by different specialists with different terms.

Semi-automatic:

It is a mixed indexing which uses analysis of a specialist in the field and the computerized program and also the latter uses a controlled indexing language [05].

The human specialist is the one who makes the final choice of descriptors.

Automatic:

This is an automatic indexing. It goes through a set of steps to automatically create the index: lexical analysis, elimination of stop words, normalization (lemmatization or radicalization), the selection of descriptors, calculating statistics on descriptors and documents (frequency appearance of a descriptor in a document and in the collection, the size of each document, etc.) and finally the creation of the index and possibly its compression.

Lexical analysis:

This step takes the document (text) and converts it to a term list (group of characters), it recognizes the separator, numbers, punctuations…

Elimination of stop words:

Stop words are words that do not deal with the subject of the document and have no meaning. To eliminate stop words we use anti-dictionary (stop-list) which is a list of stop words already established or we eliminate words with a frequency which exceeds a predefined threshold in the collection. This step reduces the size of the index for a minimal response time, but it is not one hundred percent efficient (for example, the query be or not to be).

Standardization:

Standardization consists in representing the different variants of a term by a format unique called lemma or root. This step will also reduce the size of the index. But in some cases there is the semantic drawback when comparing with the original terms (for example terms derivate / derive, activate / active).

The choice of descriptors:

The descriptor and elementary units that represent the documents, to minimize the loss of semantic information it is wrong to have a good representation of these documents (descriptor). There are many types of descriptors [06].

-Single words: single words from document text with stop words removed the lemmas or the roots of the extracted words.

- N-grams: which represents an original text by a sequence of N consecutive characters; there are also bi-grams and trigrams.

- Compound words: a compound word is a group of words or expressions, the latter are often semantically richer than the words that compose them separately. For example, the compound word "dryer" is more precise than "dryer" and "linen" taken in isolation. It is for this reason that compound words are often used in IR.

- Concepts: concepts are expressions taken from ontology's or thesauri (of a conceptual structure).

Creation of the index:

The purpose of the indexing process is to create a set of data structures for efficient access to the representation of documents; the most used data structure is the reverse file which consists in registering the identifiers of the documents which contain it for each descriptor and also to record its frequency in each of these documents.

To reduce the size of the index, data structures are compressed before being saved to disk. (There are several methods for compressing data structures, eg the Elias Gamma method).

1.3.3 Document-request matching:

To make document-query matching, we must measure the relevance value of a document against a request. In this operation, the IRS takes the document and the request and makes their representation in the same formalism, then compares the two representations.

A score (degree of similarity or resemblance of the document to the query) which determines the probability of relevance is deduced from the result of this comparison.

This matching function is denoted SVR (d, q) (Retrieval Status Value), where d represents a document from the collection and q the query. The last step is to order the documents returned to the user.

A. Information retrieval models:

An interpretation of relevance and provided by the infomation research model. There are several IR models that are based on different theoretical frameworks eg set theory, probability, algebra, etc. Overall, we will quote the three main categories of models: Boolean models, vector models and models probabilistic [07].

a. The Boolean model

The Boolean model is simple and quick to master because from the start they used it to develop the first SRIs, even in our time many search engines (SRIs) use the Boolean model.

The Boolean model is based on set theory and Boolean algebra. In this model, a document d is represented by a set of keywords (terms) or a boolean vector.

The user's query q is represented by a logical expression, composed of terms linked by logical operators: AND (^), OR (v) and EXCEPT ().

The match (SRV) between a query and a document is an exact match, otherwise says if a document logically implies the query then the document is relevant. If not, it is considered irrelevant.

The correspondence between document and request is determined as following :

1 if d belongs to the set d written by q otherwise

RSV(d,q)=(

0 otherwise

Despite the wide use of this model, it has a number of weaknesses:

- Documents returned to the user are not ordered according to their relevance.

- The binary representation of a term in a document is not very informative, because it does not informs neither on the frequency of the term in the document nor on the length of document, which can be important information for IR.

- It is difficult for users to formulate good queries. Therefore, the set of documents found is often too large, for short queries, or completely empty in the case of long requests.

- This model does not support relevance feedback.

- Tests carried out on standard RI evaluation collections have shown that the Boolean systems have lower search efficiency.

In order to remedy certain problems of this model, extensions have been proposed, among we find: the boolean model based on the theory of fuzzy sets [08] [09] [10].

b. The vector models

The basic vector model was introduced by Salton [10], materialized within the framework of SMART system. This model is based on a geometric formalization. Indeed, the documents and requests are represented in the same space, defined by a set of dimensions, each dimension represents an index term. Requests and documents are then represented by vectors, whose components represent the weight of the indexing term considered in d2 The vector models The basic vector model was introduced by Salton [10], materialized within the framework of SMART system. This model is based on a geometric formalization. Indeed, the documents and requests are represented in the same space, defined by a set of dimensions, each dimension represents an index term. Requests and documents are then represented by vectors, whose components represent the weight of the indexing term considered in the document (the query).

Formally, if we have a space **T** of index terms of dimension n

T= $\{t1\ , t2,......,tj,....tn\}$. a document di is represented by a vector $di(Wi1\ , wi2\ , \ldots..wij\ , \ldots \ldots win)$.

A query q by a vector $q(Wq1\ , wq2\ , \ldots..wqj\ , \ldots \ldots wqn)$

Or wij (resp. wqj) represents the weight of the term tj in the document di (respectively in the request q).

The vector model offers means for taking into account the term weight in the document. In the literature, several weighting schemes have been proposed. The majority of these schemes take into account the local weighting and the global weighting [11].

Local weighting measures the importance of the term in the document. She takes take into account the local information of the term which depends only on the document. She generally corresponds to a function of the frequency of occurrence of the term in the document (denoted tf for term frequency), expressed as:

$$tfi\,j = 1 + \log(f(f(ti, dj))$$

Or $f(t_i, d_j)$est la fréquence du terme t_i dans le document d_j.

As for the overall weighting, it takes into account the information concerning the term in the collection.

A greater weight should be assigned to the terms that appear less frequently in the collection. Because the terms that appear in many documents in the collection do not distinguish the relevant documents from irrelevant documents (i.e. not very useful for discrimination).

A weighting factor global is then the advent. This factor called Idf (reverse document frequency), depends on inversely to the document frequency of the term and described as follows :

$$Idf = log(\frac{N}{ni})$$

Où ni is the document frequency of the term considered, and N is the total number of documents in the collection.

The weighting functions combining local and global weighting are referenced under the name of the measure tf x idf.

This measure gives a good approximation of the importance of the term in collections of documents of uniform size.

However, a factor important is ignored, the size of the document. Indeed, the measure (tf x idf) thus defined favors long documents because they tend to repeat the same term, which increases their frequency, therefore increase the similarity of these documents to the query.

To remedy this problem, work has proposed to integrate the size of the document in weighting formulas, as a normalization factor [12] [13].

The document-query pairing in the vector model consists of finding the vectors documents that most closely match the query.

This pairing is obtained by the evaluation of the distance between the two vectors. Several similarity measures have been defined [14].

The vector model has the advantage of considering the weight of terms in Documents, allows you to find documents that partially answer the request.

Additionally, this template provides an easy way to categorize search results, namely Based on possible similarities between documents and query.

The main drawback The vector model is that it is based on the assumption that the indexing terms are independent, However, these terms found in documents are often meaningfully related. To remedy this limitation, several variants of the vector model have been proposed.

That is, consider the dependency between indexing terms. Among them we are He finds, Generalized Vector Model [15], LSI (Latent Semantic Indexing) model [16] [17] and the contact form [02] .

c. Probabilistic Models.

Basic Probabilistic Model

The probabilistic model is based on probability theory [02].It sorts documents according to the likelihood of it fitting into a query Sort function for this form is expressed as follows:

$$Rsv(q,d)=\frac{p(Per/q,di)}{p(NPer/q,di)}$$

The basic idea of this job is to identify which documents have both strong The likelihood that it is relevant and the least likely that it is not relevant to the query. Where P (Per / q, di) and P (NPer / q, di): the probability that the document di is relevant (for each) with respect to the query q (respectively irrelevant (NPer)). By applying Bayes' formula to the two probabilities, we obtain:

$$P\ (Per/q,di)=\frac{p(Per/q).p(di/Per,q,)}{p(di)}$$

$$P\ (NPer/q,di)=\frac{p(NPer/q).p(di/NPer,q,)}{p(di)}$$

or: P (di) is the probability of choosing the document di, and we consider it to be constant; P (Per / q) denotes the probability that di is one of the documents related to the query q;

P (NPer / q) denotes the probability that di is one of the documents unrelated to the query

q;P (Per / q) and p (NPer / q) respectively denote the potential for fitness and inadequacy. From any document (with (per / q) + p (NPer / q = 1) that has been fixed. After substituting in the sort function, we will have the following formula:

$$Rsv(q,d)=\frac{p(di|Per,q)}{p(di|NPer,q)}$$

Assuming the indexing terms are independent, we can estimate both the possibilities are as follows:

$$P(d_i|\ Per,q)=\prod_{tj\in di} P(t_j|\ Per,q) \times \prod_{tj\notin di} 1-P(t_j|\ Per,q)$$

$$P(d_i|\ NPer,q)=\prod_{tj\in di} P(t_j|\ NPer,q) \times \prod_{tj\notin di} 1-P(t_j|\ NPer,q)$$

Where P (di | Per, q) denotes the probability of occurrence of the term tj knowing that the document It belongs to the group of related documents and P (tj | NPer, q) denotes the probability the term tj appears knowing that not all documents belong to it relevant.

By setting Pi = P (di | Per, q), qi = P (tj | NPer, q) and Pi = qi for the terms which do not appear in the query, and after simplification, the calculation of the score of correspondence between a document and a query can be expressed as follows

$$Rsv(di,q)=\Sigma_{ti} \in q \; log[\frac{pi(1-qi)}{qi(1-pi)}$$

To classify documents in this format, we must estimate the values of the two probabilities Pi and qi.

In the absence of the learning set (documentation); We can set the constant value of Pi.[02]

Language model

Statistical language models are used with great success in various Domains:

Speech Recognition [02][18], Machine Translation [19], Searching for information [20] [21], etc.

The use of language models in IR dates back to 1998 [22]. The principle of this form It consists in building a language model for each document either Md, then an account The possibility of creating a query with the language form of the document, or P (q| Md) [23].

Often the language model used is the uni-gram, Then the probability P (q| Md) is expressed as:

$$P\ (q|\ M_d) = \Pi_t \in_Q P\ (t|\ M_d)$$

P (t| Md) can be estimated based on the maximum likelihood estimate (maximum likelihood estimate). It is given by:

$$P\ (t|\ M_d) = \frac{tf(t,d)}{|d|}$$

Where tf(t,d) is the repetition of the term ti in document d to address the issue that missing request words in the document raise

The effect of having zero probability P (q| Md) Smoothing techniques are used, including Laplace smoothing (one addition) smoothing fine Turing, smoothing retraction, settlement by completeness, etc. [24]

1.4 Difficulties of IR

_Difficulties of access, coverage, processing time.

–Difficulties in defining relevance.

–Difficulty in defining the user's need: Because the information need formulated by a request is so vague and imprecise then the object of the information search is generally unknown which leads us to a loss of information (by comparing what the user wants and the expression of the need for information).

_Difficulty of natural language (implicit, redundant, ambiguous).

1.5Conclusion

The search for information has taken a big leap over the past forty years and has made it faster and easier to access information. But there is still a long way to go.

This first chapter is the main entry to get to the concept of query expansion and the different methods used for, which we will focus on in the second chapter on query expansion using co-occurrence.

2 Chapter II: Query Expansion

2.1 introduction :

Usually, the user does not formulate his information need in an exact way (short requests and / or the user does not provide good terms). As a result, the performance of information retrieval systems is relatively degraded. To take this difficulty into account, query reformulation techniques are used, in order to obtain potentially better queries. Modification of the query can be: adding new terms and/or re-estimating the importance of the terms of the query.

In order to select the expansion terms, several methods and techniques were used: the co-occurrence relation, the co-occurrence relation and the "Information Flow" inference mechanism, the association rules, the measure of mutual information "Mutual Information ”, query classification and EM algorithm, term classification, maximum estimate, Markov chains, relevance model, mixed model and functions based on term distribution analysis in pseudo-relevant documents, such as: Kullback-Leibler distance (KLD) and Robertson Selection Value (RSV)….

We present below the notion of query expansion and then discuss the two most important techniques of query reformulation.

And we have clarified what the notion of co-occurrence between words consists of, the general phenomenon by which they are likely to be used in the same context. We also previously gave an overview on the different measures of association, the particular measures of the χ^2 test, the likelihood ratio and the mutual information were presented.

2.2 Query Expansion

Query expansion is a process of transforming a user's request in order to provide more meaningful answers. If the search system considers that the answers given are not satisfactory, the initial query (user query) is modified to have new results. Sometimes the number of responses obtained is zero or almost (too low), in this case I should proceed to enrich the initial request; And if the search system is too verbose (a lot of rethinking) it requires a response filtering mechanism. The query reformulation techniques are proposed according to the type of data processed by the search system and also by the different contexts in which they have been used, in this context we will discuss some of the main query reformulation techniques mentioned above [26]:

2.1.1 Thematic expansion

Reformulation and thematic expansion of requests have been around for a long time and are based on different techniques.

Thematic expansion is based on the vocabulary used in the research process. Initially, many systems used statistical methods such as the calculation of statistical co-occurrence of terms.

Subsequently, various techniques were invented such as the use of morphological families of terms, user profiles or even thesauri. Statistical methods of query expansion often use co-occurrence between terms, that is, the fact that two terms often appear together in the same document.

For example, the terms port and boat are often used in the same context but it is not possible to establish a linguistic relationship between them (such as synonymy, antinomy, ..).

The analysis of co-occurrence between terms, on a corpus of documents for example, makes it possible to record pairs of words which are strongly related.

The expansion process will then be based on these co-occurrence links.

This statistical co-occurrence method was used for example by a research team whose objective was to build a documentary ontology of Law .

An automatic analysis of a corpus of legal texts was used to calculate co-occurrence links between the terms present and these links were used as the skeleton of the ontology.

A search engine then uses this ontology to suggest related words to the user in order to expand their query.

Another advantage of methods using the co-occurrence between terms is the disambiguation in the event of polysemy of a term. Indeed, if a term can have several different meanings, it is the context and the co-occurrence of this term with others which will make it possible to determine the desired meaning.

A technique related to data mining also makes it possible to find rules of association between terms, that is to say terms that are often found together. The French Cismef project used this method to improve the search engine of its medical portal.

From their corpus of referenced documents, a knowledge extraction process was set up in order to deduce association rules between keywords. These association rules are then used by the search engine to perform query expansion [26].

However, statistical expansion methods do not always achieve a significant improvement in results, statistical methods are easier to prepare within the research system but they are not as successful as methods based on linguistic relationships.

Therefore, it is absolutely important to broaden the search performed on a keyword to include all adjective family terms. For example, if the user gives the keyword sea, an expansion can be made to the following words: sea, sailor, marine, marine...

These methods have been used with success by relying on various dictionaries to create word families.

There are other interesting linguistic relationships to expand the query, such as semantic relationships between terms.

The main semantic links used in search systems are synonym relationships, hyper-nouns (generalization), hyponymy (specialization)...

The most used relationship that seems to give good results is the thesaurus relationship.

The expansion process often uses several types of semantic relationships and not just one. Voorhees for example tested different combinations of using semantic relationships to extend inquiries.

His experiments were based on semantic links provided in the system WordNet2, which is a very general lexical base [26].

It turns out that improvements in the results were possible when the terms added to the query were very close meaningfully to the initial terms. Baziz et al also used WordNet database semantic relationships in their search system to extend queries.

They have obtained good results according to the requests, but they indicated that Word Net is a very general rule and therefore not very effective for specialized research in a specific field.

Other experiments used less common semantic bonds, such as the study by Claveau and Sébillot in which the links used are the verb noun kywalia links. Qualia correlates a noun and a verb that are associated significantly.

According to these experiments, the expansion based on quality links improves greatly Results.

However, the disadvantage of this method is that it involves an elementary process of building a term base with these very specific qualitative relationships. No lexical base like WordNet actually contains such links.

These various techniques that use semantic relationships between terms necessarily need access to lexical resources in order to know these relationships.

Therefore, interest in the thesaurus or ontology in the search system appears here. Some studies have made use of existing lexical resources such as WordNet, but there is not always a lexical base suitable for the field of study. Douyere et al for example, had to build a French lexical base applicable to the medical field from the English-speaking MeSH thesaurus. When the search system relates to documents that all belong to the same field, it is very effective to reformulate the queries with a detailed lexical resource for that field rather than a general one.

There are also other ways to expand that rely on completely different technologies. For example, Bottraud et al try using user profiles. Their idea is to build by learning the user profile on which the expansion will be built [26].

The user profile is constantly updated with regard to the documents being referenced, so that it becomes more and more user friendly and allows for better results the more it is enriched.

Finally, since methods of query extension depend on the type of data involved in the search system, techniques most appropriate to geographic data have been studied in order to allow for spatial expansion.

2.1.2 Spatial expansion

Most search systems focus on the objective side of queries by allowing keywords to be used to perform a search. However, geographic information is an important part of the data available on the web, but search engines do not adapt to the specifics of this data. As Egenhofer points out, if you were to search for lakes in Maine using a traditional search engine, you would get a lot of irrelevant answers. The results will certainly contain the terms Lakes and Maine but will not necessarily respect the topological relationship "in" **[26].**

Therefore, it proposes an evolution towards a semantic geographic network in which the spatial location and topological relationships between geographical organisms will be taken into account. According to him, the semantic geographic network should be based on adding semantics to web documents, just as suggested Berners-Lee, with the addition of a description and spatial constraints.

Spatial queries can then be properly processed by the search engine that will consult the associated geographic ontology [26].

Another example of spatial expansion has been provided by Fu et al. in 2005.

Their proposal focuses on ambiguous spatial relationships (near, toward, around, north, ...).

The idea is that every spatial object has a geographical footprint and that depending on the request made for that object, its fingerprint can be modified.

For example, if a user searches for castles near Edinburgh, the search system will first consult the geographic ontology to find out the geographic footprint (area of influence) associated with Edinburgh.

If he gets a satisfactory number of responses with this fingerprint, he will present it to the user, but if he gets too many responses or doesn't get enough responses, then the size of the geographic footprint will change according to the topological relationship (here the footprint will be) enlarged or shrunk across Edinburgh) The application was re-submitted to Drive Research.

The results obtained in this way are very encouraging when you effectively use Ontology in an environment where the data is geographically located [26].

2.3 Co-occurrence

In this part of it must be concerned with the links between the words of the lexicon. What are these links?

Links that represent relatively strong associations between words such as the relation of synonymy, anonymity, hyperonymy, meronymy, etc.

Co-occurrence and another kind of association, weaker in nature, but still very relevant

Although in this dissertation this knowledge about words is used for researching information, it could potentially be exploited in other applications such as automatic text categorization as well as automatic text translation.

2.3.1 Definition

The notion of co-occurrence refers to the general phenomenon by which words are likely to be used in the same context [27].

In another way, when the presence of a word in a text indicates the presence of another word it is a co-occurrence.

Let's think of a simple example of two words to better understand this definition, such as "*Student*" and "*Class*", two words which in all likelihood are used most of the time in a common context, that of "Education" often. , if we find one of these words in a text, we can predict that the second will also be present.

These two words are neither synonyms (they do not have the same meaning) nor antonyms (one is not the opposite of the other). There is no hyperonymy (The meaning of one is 'is not included across the board) and there is no meronymy (one is not a part of the other).

Yet it is obvious that these two words do share something. We therefore choose to say that they are co-occurring.

There are several examples of very easy to find word pairs: "*Plane*" and "*Airport*", "*doctor*" and "*nurse*", "*student*" and "*teacher*", and so on.

To better understand the notion of co-occurrence, we must make a comparison between another kind of association between words which is collocation; co-occurrence and collocation share some points in common, but they are different all the same.

Rather, a collocation is defined as being: "*a sequence of two or more consecutive words which are characterized by syntactic and semantic unity and whose meaning or connotation cannot be derived from the meaning of each of its individual components* "[28].

Typically, it's a group of two or more words that correspond to some conventional way of saying things.

For example nominal groups (eg: "*weapons of mass destruction*"), verbal groups (eg.

"Take into account") or other common expressions (eg: "rich and famous") are considered to be collocations [29].

It is a combination of one or more words that is difficult to understand or explain but often to use, for example as "*artificial intelligence*" and not "*artificial intelligence*").

Among differences between co-occurrence and collocations is their limited compositionality (An expression in natural language is compositional when the overall meaning of the expression can be predicted by the meaning of each of its parts (eg, "*red house*")).

Think of the expressions "*to be dressed to the nines*" or "*to fall in the apples*", which are not intended to refer to "*pins*" or "*apples*", on the contrary; It was mentioned above that the collocations presented a limited compositionality, in the sense that the modification of the semantics, resulting from the combination of words, can be more subtle than in the preceding examples (eg: "*white wine*"). [29]

Adding to the non-compositionality, the presence of a collocation can be indicated by other indices, when a word cannot be changed by another which is almost a synonym (eg: "*yellow wine*" instead of "*wine. white* ") or that the expression cannot be modified by adding words (eg:" *falling into small apples* ") or by applying grammatical transformations.

And if you can't translate an expression word for word, it is probably because it is a collocation (eg: "*take a decision*" and "*make a decision*"). [29]

So, with all that has been explained previously regarding collocation, we can distinguish the

co-occurrence; the co-occurrence between words may not appear in a common grammatical unit and yet the words may not be in a particular order.

Two words considered to be co-occurring are strongly associated with each other, but the common context in which they appear is wider than in the case of collocations: it can be a paragraph, a text or a collection of texts according to the application, according to the intended use for this information. [29]

2.3.2 Association measures

Several ways of assigning an association score to a pair of words have been developed, as part of the statistical approach to natural language processing.

Among these measures, some are based on solid theoretical foundations, while others are more in the domain of heuristics. [29]

Some are drawn directly from the discipline of statistics while others have arisen in the field of information research.

Three of these metrics were chosen (based on their popularity and good performance in certain applications) to be further investigated and discussed in detail below.

In general, association scores obtained from different measures cannot be compared directly, in many cases also the numerical value as such cannot be interpreted [29].

Typically, their use is more for the purpose of ordering pairs of candidate words according to their degree of association. All subsequent processing and comparisons are based on lists of n best words or on thresholds, and the numerical value as such is no longer considered [W01].

Generally, association measures can be applied to both collocations and co-occurrences, with little or nothing.

For each of the measures that will be presented here (and for almost all the others), the calculation is based on the data in Table 01, which contains the frequencies of occurrence of the pairs of words. It is from the four values present in this table that we will calculate the degree of association between two words: (a) the number of times the two words appear together, (b) the number of times the first word is present without the second being present, (c) the number of times the second word is present without the first being present, and (d) the number of times neither of the two words occurs.

	Word 2 present	Word 2 missing
Word 1 present	A	B
Word 2 missing	C	D

Table 01 - Contingency table to measure the degree of association between two words

It is the interpretation given to the values of this table that specifies the phenomenon being measured.

We must define the common context in which two words must appear in order for them to appear together. If we want to measure collocations, we will analyze the presence of words within the same bigram or the same trigram, and the order of appearance will be of some importance. If, on the other hand, we want to measure co-occurrences, we will analyze the presence of words inside the same sentence, the same paragraph, the same text or the same collection of texts, depending on the case. [29]

In our work in, the co-occurrence between two words is seen as being the tendency of two words to appear in the same document. The contingency table at the base of our calculations therefore lists the number of documents containing or not the two words whose co-occurrence is being studied.

A. χ^2 test

A first way to assign a numerical value to the degree of co-occurrence between two words is the χ^2 test [27].

The fact that two words have a high frequency of occurrence together does not necessarily reflect an association between them, it may be a random phenomenon, it is a problem that must be solved.

The question then is whether the co-occurrence occurs more often than by chance.

The classic way of doing this begins with formulating a null hypothesis H0 that there is no association between the words at all.

We calculate the probability that a co-occurrence will occur if H0 were true and we reject this hypothesis if the probability turns out to be too low. Otherwise, we retain the hypothesis of independence. The general idea of measuring χ^2 is to compare the frequencies observed in a corpus with the frequencies expected if there were independence. If the difference between the two is great, we can reject the independence hypothesis.

Based on Table 01, the equation that reflects this notion is as follows:

$$\chi^2 = \Sigma_{ij} \frac{\left(O_{ij} - E_{ij}\right)^2}{E_{ij}}$$

Or

- i covers the rows of table 01
- j covers the columns of table 01
- Oij is the observed value for cell (i, j)
- Eij is the expected value for cell (i, j)

The statistic therefore sums the differences squared between the observed and expected values for each cell in the table, each difference being normalized by the expected value.

The observed values are quite simply calculated from the texts of the corpus.

As for the expected values, they are calculated using the marginal probabilities, that is, from the totals for the rows and for the columns of the table, converted into proportions.

For example, the expected probability E_11, which is the probability that word 1 and word 2 are both present in a text, is calculated as follows:

We multiply the marginal probability that word 1 appears with the marginal probability that word 2 occurs, as well as the number of documents.

Knowing that N represents the total number of documents, we can do the following calculations:

- Probability that word 1 appears : $P_1 = (a + b) / N$
- Probability that word 2 appears : $P_2 = (a + c) / N$
- Probability that word 1 and word 2 both appear: $E_{11} = P_1 \cdot P_2 \cdot N$

After some algebraic manipulations and simplifications, we arrive at the following expression which allows us to obtain the value of χ^2:

$$\chi^2 = \frac{N(ad-bc)^2}{(a+b)(a+c)(b+d)(c+d)}$$

When the sample size is large enough, the χ^2 test follows a χ^2 distribution.

A table of this distribution is consulted to know the degree of confidence associated with the value obtained by the calculation. For example, at the top of the critical value $\chi^2 = 3.841$.

B. Likelihood ratio

Another way to perform hypothesis testing is the likelihood ratio. [27]

By using the likelihood ratio test the numerical value that is calculated can be more easily interpreted than with the χ^2 test.

This test uses two hypotheses:

Hypothesis 1, possibility of complete independence between the two words: the conditional probabilities that word 2 is present knowing respectively the presence or absence of word 1 are the same.

Hypothesis 2, possibility that the two words are dependent implies that these conditional probabilities are different.

We consider the two hypotheses presented below and the ratio indicates which is the most probable.

Hypothesis 1: P (word 2 | word 1 present) = p = P (word 2 | word 1 absent) (independence)

Hypothesis 2: P (word 2 | word 1 present) = p1 ≠ p2 = P (word 2 | word 1 absent) (dependence)

Based on these assumptions, we want to evaluate the likelihood of each of them, given the frequencies observed in the corpus concerning the appearance of words, in order to then report these likelihood values. Everything is done by assuming a binomial distribution.

We first estimate the maximum likelihood for the following probabilities, using the data from our initial contingency table:

- $p = (a + c) / N$
- $p1 = a / (a + b)$
- $p2 = c / (c + d)$

Then, we consider the expression L (H) to be the likelihood of a hypothesis H.

The logarithm of the likelihood ratio of the two hypotheses in question is therefore calculated as follows [29]:

$$\log \lambda = \log \frac{L(H_1)}{L(H_2)} = \log \frac{B(a,a+b,p).B(c,c+d,p)}{B(a,a+b,p_1).B(c,c+d,p_2)}$$

où $B(k, n, x) = \binom{n}{k} x^k (1 - x)^{n-k}$ (binomial distribution)

Indeed, the likelihood of H1, for example, corresponds to the probability of having a times word 2 present when word 1 is present (a + b times) multiplied by the probability of having c times word 2 present when word 1 is absent (c + d times).

Similar reasoning can be done for the second hypothesis.

A simplification leads to the following expression:

$\log \lambda = \log L(a, a + b, p) + \log L(c, c + d, p) - \log L(a, a + b, p1) - \log L(c, c + d, p2)$

où $L(k, n, x) = x^k(1 - x)^{n-k}$

C. Mutual information

The third possible association measure is pointwise mutual information (PMI).

It measures the amount of information that the presence of one word gives about the presence of another word.

The following equation is used to calculate this measure and involves, in the numerator, the proportion of documents containing the two words and then, in the denominator, the respective proportions of documents in which only one of the two words appears.**[29]**

$$\mathrm{I\ (mot\ 1,\ mot\ 2)} = log_2 \frac{P(mot1\ \&\ mot2)}{P(mot1)P(mot2)}$$

In conclusion, the more two words tend to appear together, the higher the score will be. Indeed, if the two words were completely independent, the probability ratio would be 1, thus giving a final value of 0.

Conversely, if the two words have a strong tendency to appear together, then the probability in the numerator will overtake the other, increasing the mutual information score. Still from the contingency table and after certain manipulations, we arrive at the simplified way to calculate the PMI (pointwise mutual information) **[29]:**

$$I(mot1, mot2) = log_2 \frac{Na}{(a+c)(a+b)}$$

The problem in our work is how to use co-occurrence for query expansion.

2.4Our approach

The main idea in our approach is to consider the matrix of the Co-occurrence as a new corpus or data set where etch word is considered as a document, with this point of view we can calculate weights for etch word with a chosen model.

Using this method allows us, according to our assumption, to represent the strength of the relationship between two words. For example, with the use of TF-IDF, the weight of words with a lot of reciprocal appearance will be strong by the TF factor with the exception of common words, whose weight will decrease by the IDF factor

After we use query terms to rink and select candidate words for the expansion by taking the query words and inferring by them on the vector that represent them in the resulting index, then calculating the averages of the weights, then arranging the resulting vector according to the values of the weights and in the end choosing a list of candidate words

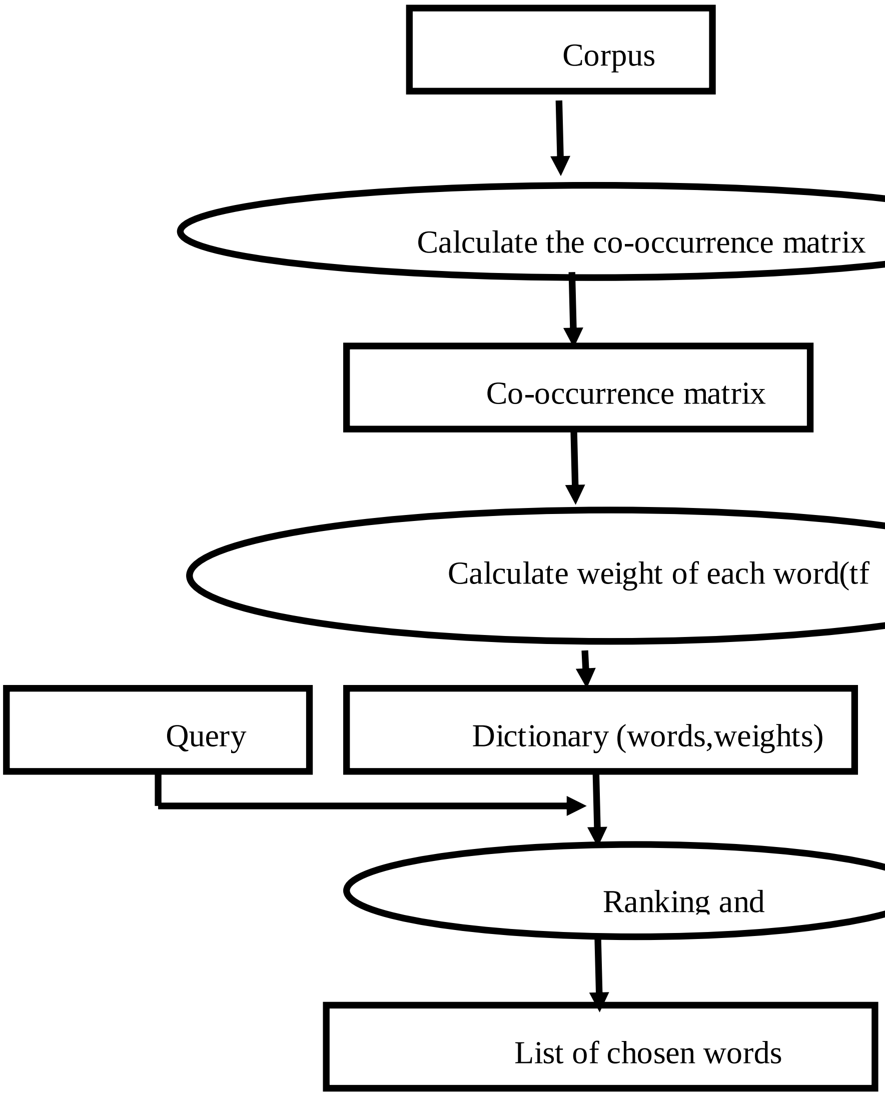

Corpus
Calculate the co-occurrence matrix
Co-occurrence matrix
Calculate weight of each word(tf
Query
Dictionary (words,weights)
Ranking and
List of chosen words

Figure 2 the process of our approach

2.5Conclusion

In this chapter we want to give an overview on query expansion while focusing on co-occurrence to properly determine the concepts and start the third chapter which and the implementation of the system (co-occurrence based query expansion).

3 Chapter III: Experimental and Result

3.1Introduction:

In this chapter we present our co-occurrence implemented approach and experimental environment: used tools, the implementation of the system, We present the experimental results obtained, as well as the analysis and evaluation of these results.

First, we gave an overview on the used tools, the used database, programming language and chosen libraries for this work.

Then the system implementation steps goes through two phases:

The first phase consists of indexing the documents using victor modal, the latter allows us to have vector of words, each word and presented by the document which continues this word and its number of occurrence (inverted frequency) then we calculate the co-occurrence matrix, at the end of the indexing comes the search step where we enter the query and perform a dictionary search comparing the words between the query and the words in the dictionary and obtain the result (co-occurrence).

The second and most important part is to consider that every word is a document and to present a list of documents and their weight and to put them in preparation for the fireworks algorithm.

Then we moved on to discussing the results, and in conclusion we mentioned what we will provide for the development of this work in the future.

3.2 Working tools

Starting with the programming language we have chosen Python along with some packages to achieve our goal. And also used the Dataset library

3.2.1Programming Language

Python is an interpreter, multi-paradigm, cross-platform programming language. It promotes structured, functional and object-oriented imperative programming. It has strong dynamic typing, automatic memory management by garbage collection and an exception management system; it is similar to Perl, Ruby, Scheme, Smalltalk, and Tcl.

The Python language is placed under a free license close to the BSD license and works on most computer platforms, from smartphones to mainframe computers6, from Windows to Unix with in particular GNU / Linux via macOS, or even Android, iOS , and can also be translated into Java or .NET. It is designed to optimize the productivity of programmers by offering high-level tools and an easy-to-use syntax.

It is also appreciated by some pedagogues who find in it a language where the syntax, clearly distinct from low-level mechanisms, allows an easy initiation to the basic concepts of programming [W02].

It was created by Guido van Rossum and first released in 1991.

Python's large standard library, commonly cited as one of its greatest strengths, provides tools suitable for many tasks. The official repository for third-party Python software contains over 130,000 packages with a wide range of features [W03].

We used TKinter, math, numpy, nltik collection packages.

In addition, we use lists, tuples:

Lists: A list is a data structure in Python that is an ordered and modifiable sequence. Each item or value that is inside a list is called an item. Lists are defined by values in square brackets [W04].

Tuples: A tuple is a sequence of immutable Python objects. Tuples are sequences, just like lists. The differences between tuples and lists are that tuples cannot be changed against lists and tuples use parentheses, while lists use square brackets. [W05]

Dictionary: array of elements indexed by immutable types of elements can be added or removed. [W06]; Set: array of unindexed unique items. [W06] [30].

3.2.2 Corpus:

In our work we have designed a corpus which is a set of 100 documents, each document contains about five to ten words, and this for the purpose of testing our software while saving time.

3.3The system implementation steps

As we said before, we have designed a small corpus to facilitate indexing and to save time during testing.

The full-text search therefore consists of two stages:

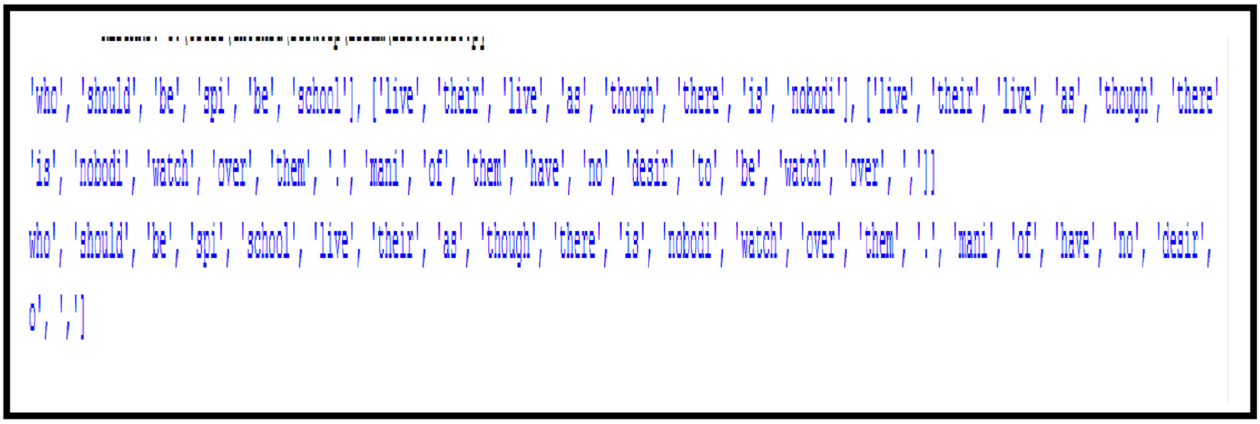
1. Indexing stage: in this stage , we applied normal indexing in addition to the calculation of the co-occurrence matrix and the computation of the resulting index from it
2. Searching stage: After entering the query, we activate the sort path and the resulting list as an input to the fireworks algorithm used for searching.

3.3.2 Indexing step:

Indexing goes through several stages or processes

1. Normalization: An important part of indexing is normalization. This is a word processor, which puts the source text in standard canonical form. This means that stop words and articles are deleted, diacritics (as in the words "pâté", "naive", "złoty") are deleted or replaced by standard alphabetic signs. In addition, only one case is chosen (only upper or lower).
2. Another important part of standardization is struggle. It is a process of reducing a word to a root or base form.
3. Generating the inverted index (index word, document)

In our work, we created the co-occurrence matrix which contains words, The number of co-occurrence of each word with the rest of the words in the corpus (fig 03).



```
'who', 'should', 'be', 'spi', 'be', 'school'], ['live', 'their', 'live', 'as', 'though', 'there', 'is', 'nobodi'], ['live', 'their', 'live', 'as', 'though', 'there'
'is', 'nobodi', 'watch', 'over', 'them', '.', 'mani', 'of', 'them', 'have', 'no', 'desir', 'to', 'be', 'watch', 'over', '.']]
who', 'should', 'be', 'spi', 'school', 'live', 'their', 'as', 'though', 'there', 'is', 'nobodi', 'watch', 'over', 'them', '.', 'mani', 'of', 'have', 'no', 'desir',
o', '.']
```

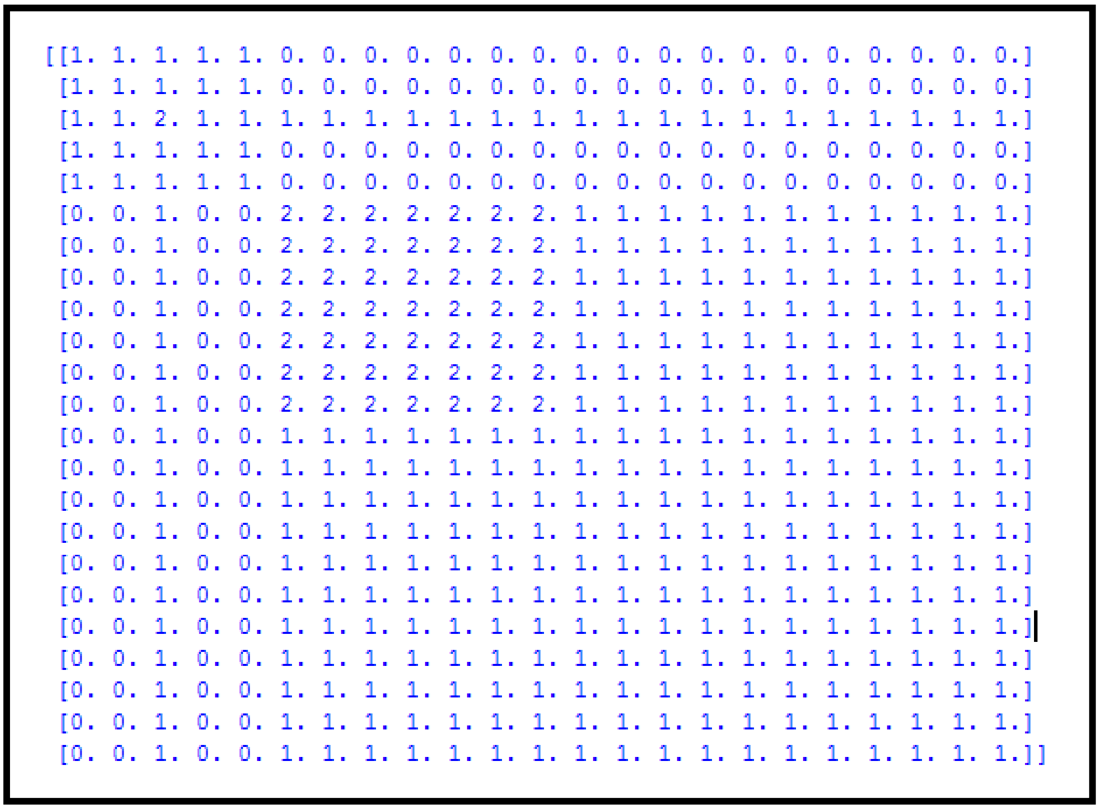
Figure 3 Dictionary of words of the corpus

Then, in a use the tf-idf measurement methods, i.e. the terms are given a weight based on how often a term appears in a particular document and how often it appears in the set from the collection of documents.

The first part of the tf-idf scheme is called the term frequency, the number of occurrences of the term in document D. The second part is called the inverse document frequency and is calculated as follows:

$$\mathit{Idfi} = \mathit{log}\ \mathbf{n}/\mathbf{dfi}$$

where ***n*** is the total number of documents in the collection and ***dfi*** the number of documents in which term appears at least once.



```
[[1. 1. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 1. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 1. 2. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 1. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 2. 2. 2. 2. 2. 2. 2. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 2. 2. 2. 2. 2. 2. 2. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 2. 2. 2. 2. 2. 2. 2. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 2. 2. 2. 2. 2. 2. 2. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 2. 2. 2. 2. 2. 2. 2. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 2. 2. 2. 2. 2. 2. 2. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 2. 2. 2. 2. 2. 2. 2. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]]
```

Figure 4 the co-occurrence matrix

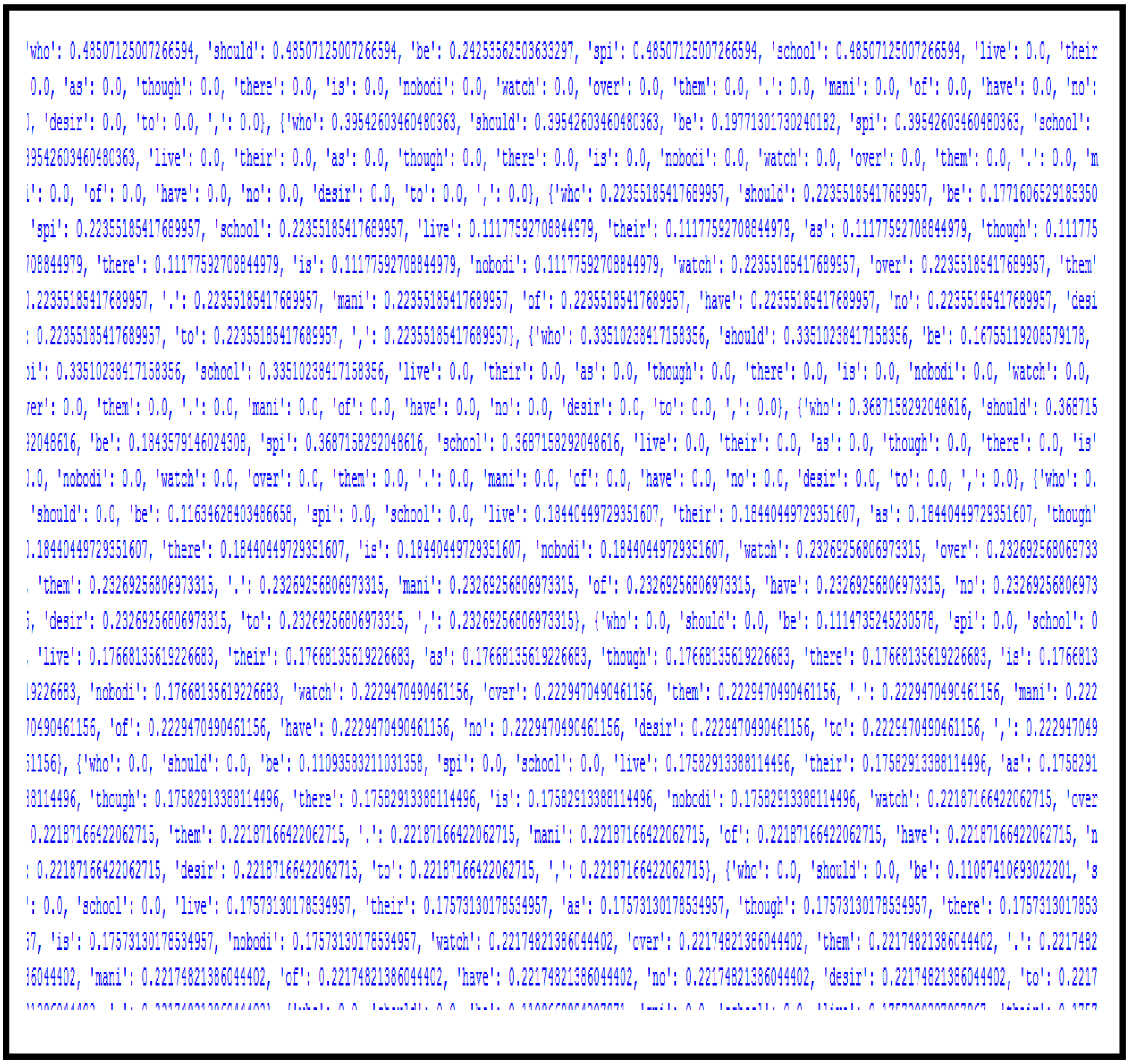
The weighting factor **Wi** of document ***i*** is determined by the product of the term frequency and the inverse document frequency:

$$\mathit{Wi} = \mathit{tfi} - \mathit{idfi}$$

The assumptions behind tf-idf are based on two characteristics of text documents. First, the more times a term appears in a document, the more relevant it is to the topic of the document.

Second, the more times a term occurs in all documents in the collection, the more poorly it discriminates between documents. [30]

At the end of this faze we have as a result a dictionary which contains all the words and their weight in the corpus.



```
'who': 0.48507125007266594, 'should': 0.48507125007266594, 'be': 0.24253562503633297, 'spi': 0.48507125007266594, 'school': 0.48507125007266594, 'live': 0.0, 'their
 0.0, 'as': 0.0, 'though': 0.0, 'there': 0.0, 'is': 0.0, 'nobodi': 0.0, 'watch': 0.0, 'over': 0.0, 'them': 0.0, '.': 0.0, 'mani': 0.0, 'of': 0.0, 'have': 0.0, 'no':
), 'desir': 0.0, 'to': 0.0, ',': 0.0}, {'who': 0.39542603460480363, 'should': 0.39542603460480363, 'be': 0.19771301730240182, 'spi': 0.39542603460480363, 'school':
39542603460480363, 'live': 0.0, 'their': 0.0, 'as': 0.0, 'though': 0.0, 'there': 0.0, 'is': 0.0, 'nobodi': 0.0, 'watch': 0.0, 'over': 0.0, 'them': 0.0, '.': 0.0, 'm
i': 0.0, 'of': 0.0, 'have': 0.0, 'no': 0.0, 'desir': 0.0, 'to': 0.0, ',': 0.0}, {'who': 0.22355185417689957, 'should': 0.22355185417689957, 'be': 0.177160652918535O
 'spi': 0.22355185417689957, 'school': 0.22355185417689957, 'live': 0.11177592708844979, 'their': 0.11177592708844979, 'as': 0.11177592708844979, 'though': 0.111775
708844979, 'there': 0.11177592708844979, 'is': 0.11177592708844979, 'nobodi': 0.11177592708844979, 'watch': 0.22355185417689957, 'over': 0.22355185417689957, 'them'
).22355185417689957, '.': 0.22355185417689957, 'mani': 0.22355185417689957, 'of': 0.22355185417689957, 'have': 0.22355185417689957, 'no': 0.22355185417689957, 'desi
: 0.22355185417689957, 'to': 0.22355185417689957, ',': 0.22355185417689957}, {'who': 0.33510238417158356, 'should': 0.33510238417158356, 'be': 0.16755119208579178,
i': 0.33510238417158356, 'school': 0.33510238417158356, 'live': 0.0, 'their': 0.0, 'as': 0.0, 'though': 0.0, 'there': 0.0, 'is': 0.0, 'nobodi': 0.0, 'watch': 0.0,
ver': 0.0, 'them': 0.0, '.': 0.0, 'mani': 0.0, 'of': 0.0, 'have': 0.0, 'no': 0.0, 'desir': 0.0, 'to': 0.0, ',': 0.0}, {'who': 0.3687158292048616, 'should': 0.368715
82048616, 'be': 0.1843579146024308, 'spi': 0.3687158292048616, 'school': 0.3687158292048616, 'live': 0.0, 'their': 0.0, 'as': 0.0, 'though': 0.0, 'there': 0.0, 'is'
).0, 'nobodi': 0.0, 'watch': 0.0, 'over': 0.0, 'them': 0.0, '.': 0.0, 'mani': 0.0, 'of': 0.0, 'have': 0.0, 'no': 0.0, 'desir': 0.0, 'to': 0.0, ',': 0.0}, {'who': 0.
 'should': 0.0, 'be': 0.11634628403486658, 'spi': 0.0, 'school': 0.0, 'live': 0.18440449729351607, 'their': 0.18440449729351607, 'as': 0.18440449729351607, 'though'
).18440449729351607, 'there': 0.18440449729351607, 'is': 0.18440449729351607, 'nobodi': 0.18440449729351607, 'watch': 0.23269256806973315, 'over': 0.232692568069733
 'them': 0.23269256806973315, '.': 0.23269256806973315, 'mani': 0.23269256806973315, 'of': 0.23269256806973315, 'have': 0.23269256806973315, 'no': 0.23269256806973
5, 'desir': 0.23269256806973315, 'to': 0.23269256806973315, ',': 0.23269256806973315}, {'who': 0.0, 'should': 0.0, 'be': 0.1114735245230578, 'spi': 0.0, 'school': 0
 'live': 0.17668135619226683, 'their': 0.17668135619226683, 'as': 0.17668135619226683, 'though': 0.17668135619226683, 'there': 0.17668135619226683, 'is': 0.1766813
19226683, 'nobodi': 0.17668135619226683, 'watch': 0.2229470490461156, 'over': 0.2229470490461156, 'them': 0.2229470490461156, '.': 0.2229470490461156, 'mani': 0.222
70490461156, 'of': 0.2229470490461156, 'have': 0.2229470490461156, 'no': 0.2229470490461156, 'desir': 0.2229470490461156, 'to': 0.2229470490461156, ',': 0.222947049
61156}, {'who': 0.0, 'should': 0.0, 'be': 0.11093583211031358, 'spi': 0.0, 'school': 0.0, 'live': 0.17582913388114496, 'their': 0.17582913388114496, 'as': 0.1758291
38114496, 'though': 0.17582913388114496, 'there': 0.17582913388114496, 'is': 0.17582913388114496, 'nobodi': 0.17582913388114496, 'watch': 0.22187166422062715, 'over
 0.22187166422062715, 'them': 0.22187166422062715, '.': 0.22187166422062715, 'mani': 0.22187166422062715, 'of': 0.22187166422062715, 'have': 0.22187166422062715, 'n
: 0.22187166422062715, 'desir': 0.22187166422062715, 'to': 0.22187166422062715, ',': 0.22187166422062715}, {'who': 0.0, 'should': 0.0, 'be': 0.11087410693022201, 's
': 0.0, 'school': 0.0, 'live': 0.17573130178534957, 'their': 0.17573130178534957, 'as': 0.17573130178534957, 'though': 0.17573130178534957, 'there': 0.1757313017853
57, 'is': 0.17573130178534957, 'nobodi': 0.17573130178534957, 'watch': 0.22174821386044402, 'over': 0.22174821386044402, 'them': 0.22174821386044402, '.': 0.2217482
36044402, 'mani': 0.22174821386044402, 'of': 0.22174821386044402, 'have': 0.22174821386044402, 'no': 0.22174821386044402, 'desir': 0.22174821386044402, 'to': 0.2217
```

Figure 5 Dictionary (words, weights)

3.3.3 Searching stage:

After the index is created, uses the search algorithm to crawl the index (instead of the original document set) and exposes the results. Indexing takes a lot of time and effort, but offers a much faster information search

In the field of information retrieval, this step is based on models to achieve the goal, where models work on sets of large and fixed documents (corpus), through which we can find useful information that matches the better to the search query.

In our work, we used the vector space model, which is an algebraic model; it represents the text documents as a vector of words and weights [30]

But this model is used inside the implementation of the Fireworks optimization algorithm.

In our work the searching stage comports two phases:

1. Ranking and selection the list of candidate words for expansion
2. Use that list as a input for the fireworks optimization algorithm to achieve the retrieving process

1. Ranking and selection process

Considered we have a query with three words W_1 W_2 W_3 then we have:

$$W_1\{W_1We_{11}\,,W_2We_{21}\,,W_3WE_{31}\ldots\ldots\ldots\ldots\ldots\ldots W_nWe_{n1}\,\}$$
$$W_2\{W_1We_{12}\,,W_2We_{22}\,,W_3WE_{32}\ldots\ldots\ldots\ldots\ldots\ldots W_nWe_{n2}\,\}$$
$$W_3\{W_1We_{13}\,,W_2We_{23}\,,W_3WE_{33}\ldots\ldots\ldots\ldots\ldots\ldots W_nWe_{n3}\,\}$$

Where We_{ij} is the weight of the word j in the vector of the word I based on tf idf measure.

For etch word we calculate a vertical addition and we divide the result with the length of the query

After we have a vector with a set of words we sorted it and we select K first word as a candidate words.

After ranking and selection, get a list of conditional documents to put them in readiness for the fireworks algorithm to find the expected results

2. Implementation of FW:

In this step we have used the fireworks part of the memory of the past year which is as a continuation

After the random selection of words and added to the query, where the sparks appear in their site at the beginning then burst fireworks where the transaction is a search in the neighborhood around a specific site.

This explosion applies to every member of the population, resulting in the generation of sparks around each individual, with this, both are calculated:

□ **The number of sparks**

$$s_i = m \cdot \frac{y_{\max} - f(x_i) + \xi}{\sum_{i=1}^{n} (y_{\max} - f(x_i)) + \xi}$$

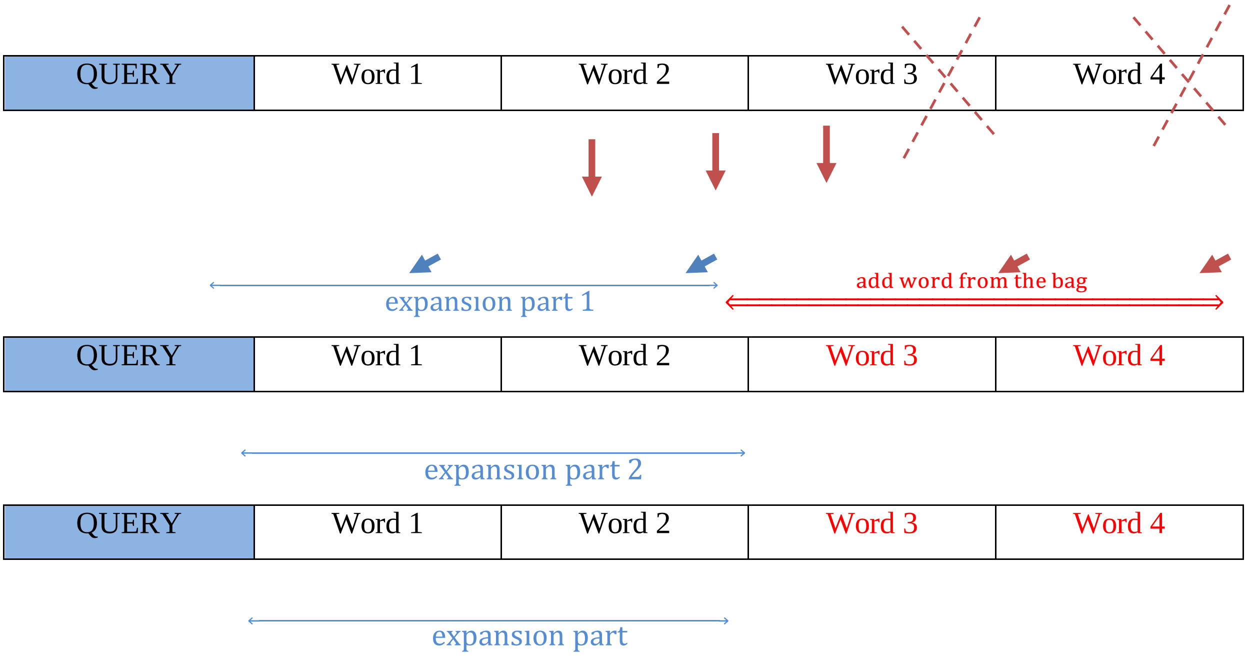
□ **the amplitude of each individua**

$$A_i = \hat{A} \cdot \frac{f(x_i) - y_{\min} + \xi}{\sum_{i=1}^{n} (f(x_i) - y_{min}) + \xi}$$

Example 1:

For example, suppose we get 4 new sparks and 2 amplitudes, which means that we have produced four queries, but each has two words different from the first extended query.

These two words are deleted randomly and keep the other two words, and we have two blank cells, filled with words taken from the first bag in which the words selected as a starting point. [30]



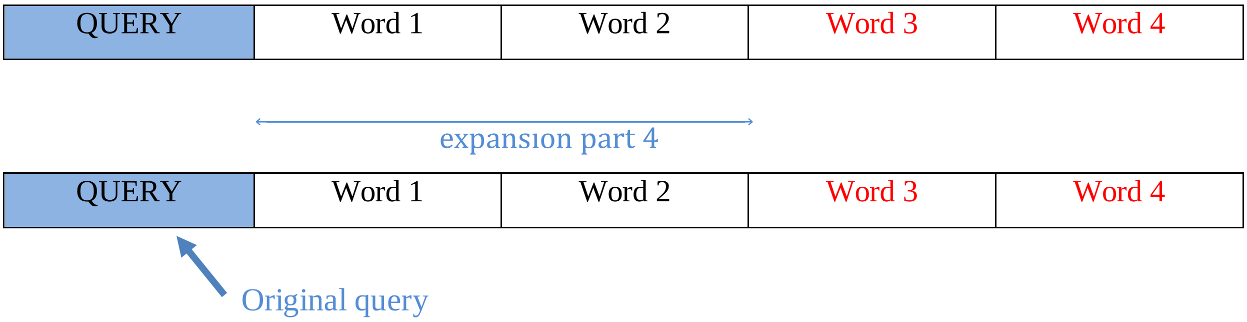


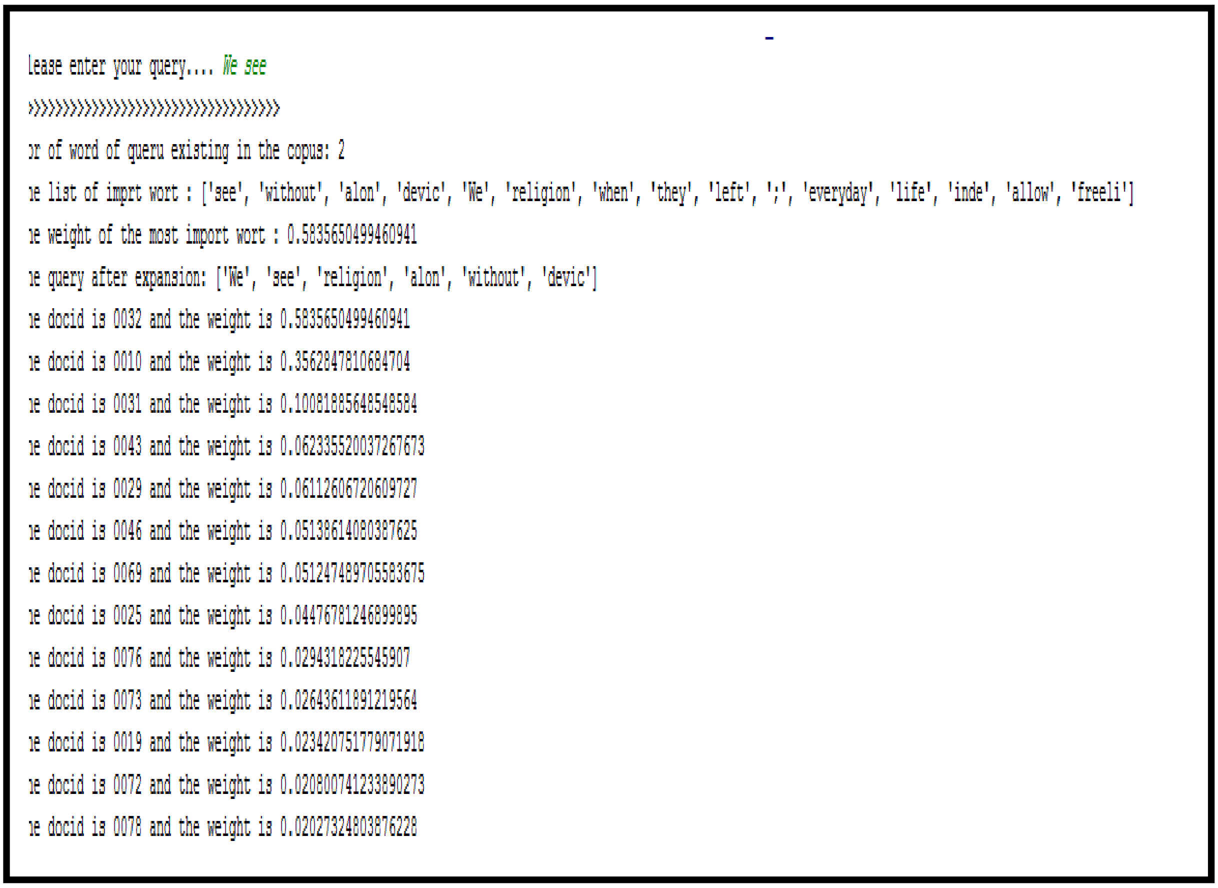
Figure 6 the expansion process

Selection:

The meaning of selection: We select the next generation members who result from the explosion of sparks, where initially the best individual is retained, while the rest, n - 1 individual is selected according to the distance between each individual and others, where the opportunity to take each one is dependent on furthest individual About other individuals[30].

In conclusion we have summarized these steps in the diagram presented if below:

And finally we will have a result like the following image (exemple)



```
lease enter your query.... We see
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
or of word of queru existing in the copus: 2
ne list of imprt wort : ['see', 'without', 'alon', 'devic', 'We', 'religion', 'when', 'they', 'left', ';', 'everyday', 'life', 'inde', 'allow', 'freeli']
ne weight of the most import wort : 0.5835650499460941
ne query after expansion: ['We', 'see', 'religion', 'alon', 'without', 'devic']
ne docid is 0032 and the weight is 0.5835650499460941
ne docid is 0010 and the weight is 0.3562847810684704
ne docid is 0031 and the weight is 0.10081885648548584
ne docid is 0043 and the weight is 0.062335520037267673
ne docid is 0029 and the weight is 0.06112606720609727
ne docid is 0046 and the weight is 0.05138614080387625
ne docid is 0069 and the weight is 0.051247489705583675
ne docid is 0025 and the weight is 0.04476781246899895
ne docid is 0076 and the weight is 0.0294318225545907
ne docid is 0073 and the weight is 0.02643611891219564
ne docid is 0019 and the weight is 0.023420751779071918
ne docid is 0072 and the weight is 0.020800741233890273
ne docid is 0078 and the weight is 0.02027324803876228
```

Figure 7 Example of the result obtained after query expansion

3.4 Test and result

For our work we use tow measure of performance Precision and Recall

Precision is the number of relevant results returned to the total number of results returned. [30]

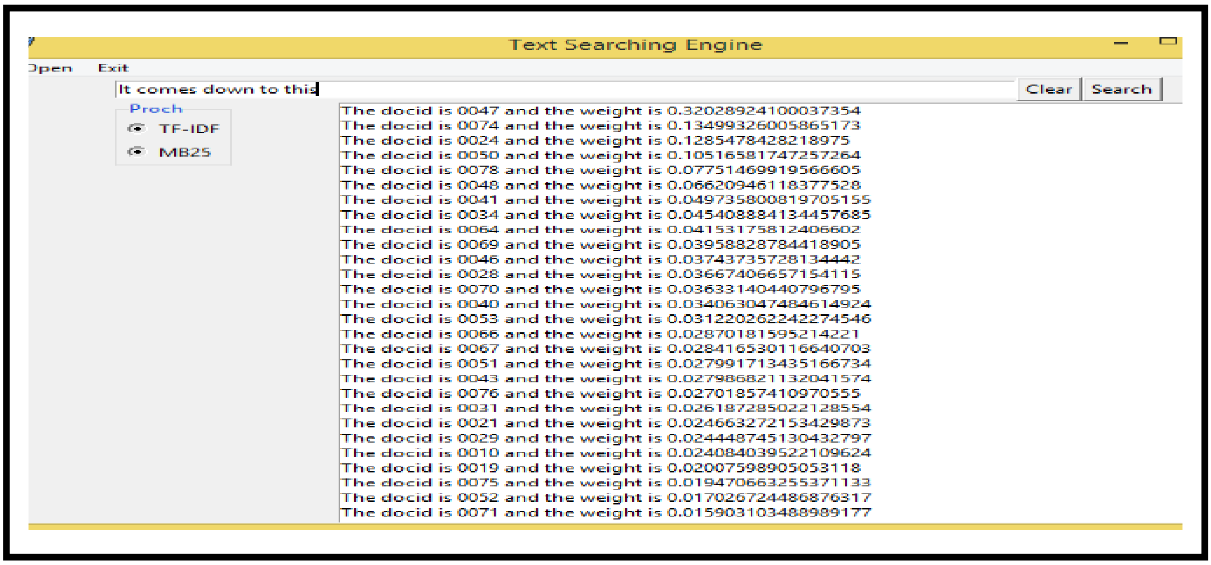
$$Precision = \frac{|\{Relevant\ Documents\} \cap \{Retreved\ Documents\}|}{\{Retreved\ Documents\}}$$

Recall measures the quantity of relevant results returned by a search, while precision is the measure of the quality of the results returned. Recall is the ratio of relevant results returned to all relevant results. [30]

$$Precision = \frac{|\{Relevant\ Documents\} \cap \{Retreved\ Documents\}|}{\{Relevant\ Documents\}}$$

In order to verify the effectiveness of the proposed approach, we randomly selected Document from the corpus, we selected query contains five words: " *It comes down to this* ", we will use this query as a standard target query.

We have the result shown in the following Figure using standard search:



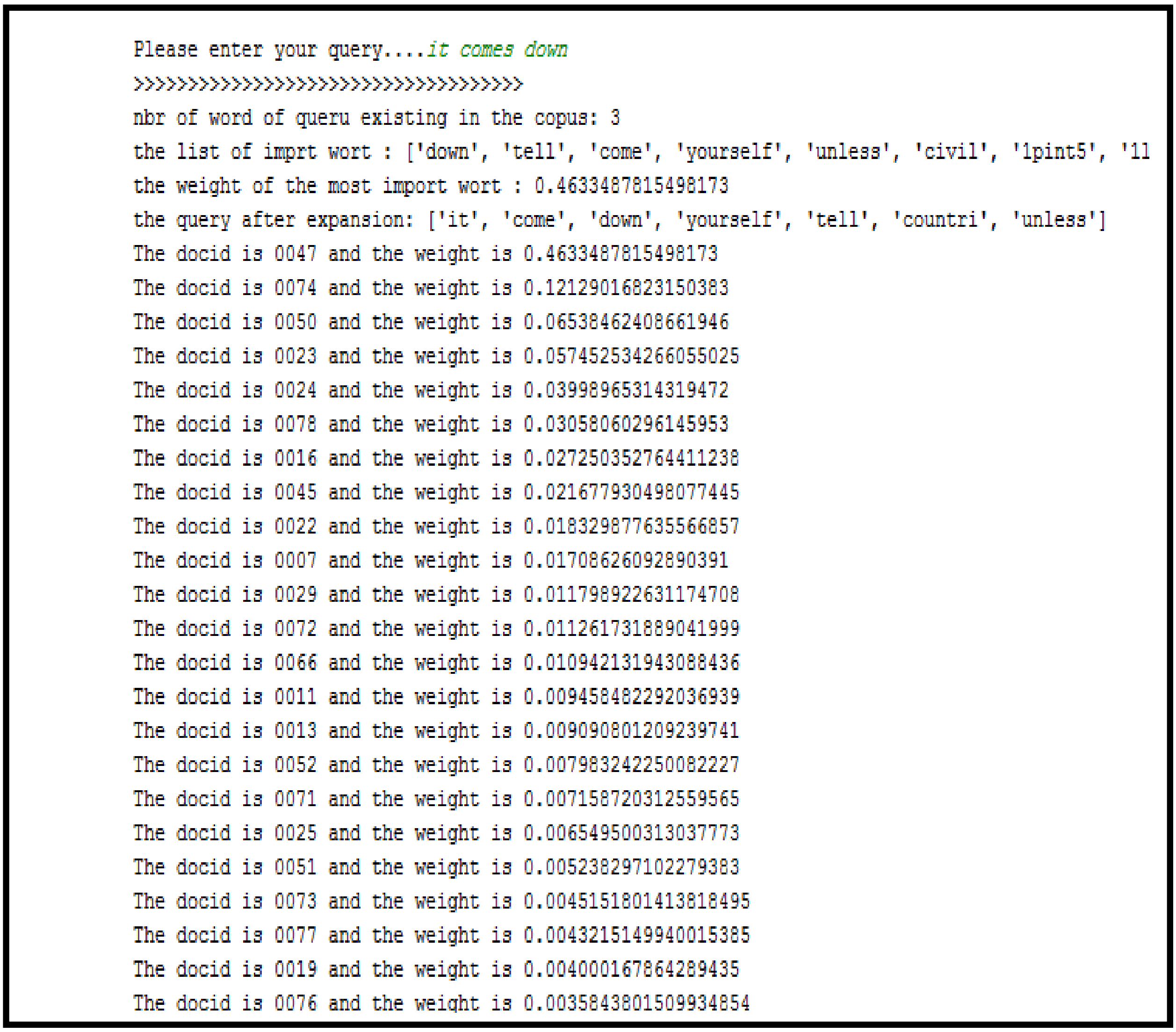
It comes down to this

```
The docid is 0047 and the weight is 0.32028924100037354
The docid is 0074 and the weight is 0.1349932600586517З
The docid is 0024 and the weight is 0.1285478428218975
The docid is 0050 and the weight is 0.10516581747257264
The docid is 0078 and the weight is 0.07751469919566605
The docid is 0048 and the weight is 0.06620946118377528
The docid is 0041 and the weight is 0.049735800819705155
The docid is 0034 and the weight is 0.045408884134457685
The docid is 0064 and the weight is 0.04153175812406602
The docid is 0069 and the weight is 0.03958828784418905
The docid is 0046 and the weight is 0.03743735728134442
The docid is 0028 and the weight is 0.03667406657154115
The docid is 0070 and the weight is 0.03633140440796795
The docid is 0040 and the weight is 0.034063047484614924
The docid is 0053 and the weight is 0.031220262242274546
The docid is 0066 and the weight is 0.02870181595214221
The docid is 0067 and the weight is 0.028416530116640703
The docid is 0051 and the weight is 0.027991713435166734
The docid is 0043 and the weight is 0.027986821132041574
The docid is 0076 and the weight is 0.02701857410970555
The docid is 0031 and the weight is 0.026187285022128554
The docid is 0021 and the weight is 0.024663272153429873
The docid is 0029 and the weight is 0.024448745130432797
The docid is 0010 and the weight is 0.024084039522109624
The docid is 0019 and the weight is 0.02007598905053118
The docid is 0075 and the weight is 0.019470663255371133
The docid is 0052 and the weight is 0.017026724486876317
The docid is 0071 and the weight is 0.015903103488989177
```

Figure 8 Retrieved document for the standard target query1

We will use these results as reference results for calculation the precision and the recall.

As a second step we searched the query: “ *it comes down* “, but we use now the extended query by our application, and we have the results shown in following figure.



```
Please enter your query....it comes down
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
nbr of word of queru existing in the copus: 3
the list of imprt wort : ['down', 'tell', 'come', 'yourself', 'unless', 'civil', '1pint5', '11
the weight of the most import wort : 0.4633487815498173
the query after expansion: ['it', 'come', 'down', 'yourself', 'tell', 'countri', 'unless']
The docid is 0047 and the weight is 0.4633487815498173
The docid is 0074 and the weight is 0.12129016823150383
The docid is 0050 and the weight is 0.06538462408661946
The docid is 0023 and the weight is 0.057452534266055025
The docid is 0024 and the weight is 0.03998965314319472
The docid is 0078 and the weight is 0.03058060296145953
The docid is 0016 and the weight is 0.027250352764411238
The docid is 0045 and the weight is 0.021677930498077445
The docid is 0022 and the weight is 0.018329877635566857
The docid is 0007 and the weight is 0.01708626092890391
The docid is 0029 and the weight is 0.011798922631174708
The docid is 0072 and the weight is 0.011261731889041999
The docid is 0066 and the weight is 0.010942131943088436
The docid is 0011 and the weight is 0.009458482292036939
The docid is 0013 and the weight is 0.009090801209239741
The docid is 0052 and the weight is 0.007983242250082227
The docid is 0071 and the weight is 0.007158720312559565
The docid is 0025 and the weight is 0.006549500313037773
The docid is 0051 and the weight is 0.005238297102279383
The docid is 0073 and the weight is 0.0045151801413818495
The docid is 0077 and the weight is 0.0043215149940015385
The docid is 0019 and the weight is 0.004000167864289435
The docid is 0076 and the weight is 0.0035843801509934854
```

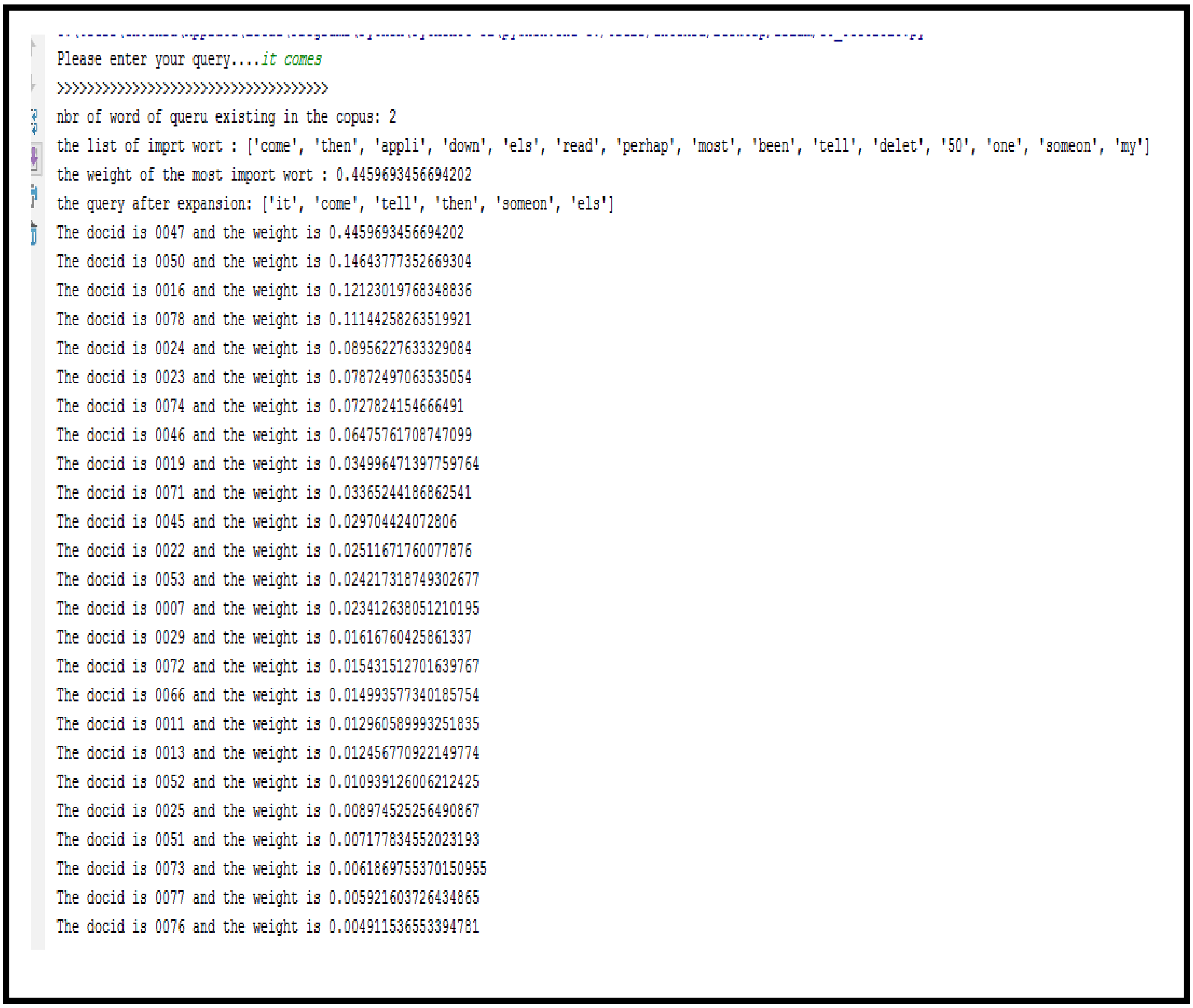
Figure 9 Retrieved document for three words query1

In this case: when we calculate both of precision & recall we get:

Precision = 08/20 = 0,4

Recall =08/23 = 0,34

We will do another calculation, this time using the query “ *it comes* “



```
Please enter your query....it comes
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
nbr of word of queru existing in the copus: 2
the list of imprt wort : ['come', 'then', 'appli', 'down', 'els', 'read', 'perhap', 'most', 'been', 'tell', 'delet', '50', 'one', 'someon', 'my']
the weight of the most import wort : 0.4459693456694202
the query after expansion: ['it', 'come', 'tell', 'then', 'someon', 'els']
The docid is 0047 and the weight is 0.4459693456694202
The docid is 0050 and the weight is 0.14643777352669304
The docid is 0016 and the weight is 0.12123019768348836
The docid is 0078 and the weight is 0.11144258263519921
The docid is 0024 and the weight is 0.08956227633329084
The docid is 0023 and the weight is 0.07872497063535054
The docid is 0074 and the weight is 0.0727824154666491
The docid is 0046 and the weight is 0.06475761708747099
The docid is 0019 and the weight is 0.034996471397759764
The docid is 0071 and the weight is 0.03365244186862541
The docid is 0045 and the weight is 0.029704424072806
The docid is 0022 and the weight is 0.02511671760077876
The docid is 0053 and the weight is 0.024217318749302677
The docid is 0007 and the weight is 0.023412638051210195
The docid is 0029 and the weight is 0.01616760425861337
The docid is 0072 and the weight is 0.015431512701639767
The docid is 0066 and the weight is 0.014993577340185754
The docid is 0011 and the weight is 0.012960589993251835
The docid is 0013 and the weight is 0.012456770922149774
The docid is 0052 and the weight is 0.010939126006212425
The docid is 0025 and the weight is 0.008974525256490867
The docid is 0051 and the weight is 0.007177834552023193
The docid is 0073 and the weight is 0.0061869755370150955
The docid is 0077 and the weight is 0.005921603726434865
The docid is 0076 and the weight is 0.004911536553394781
```

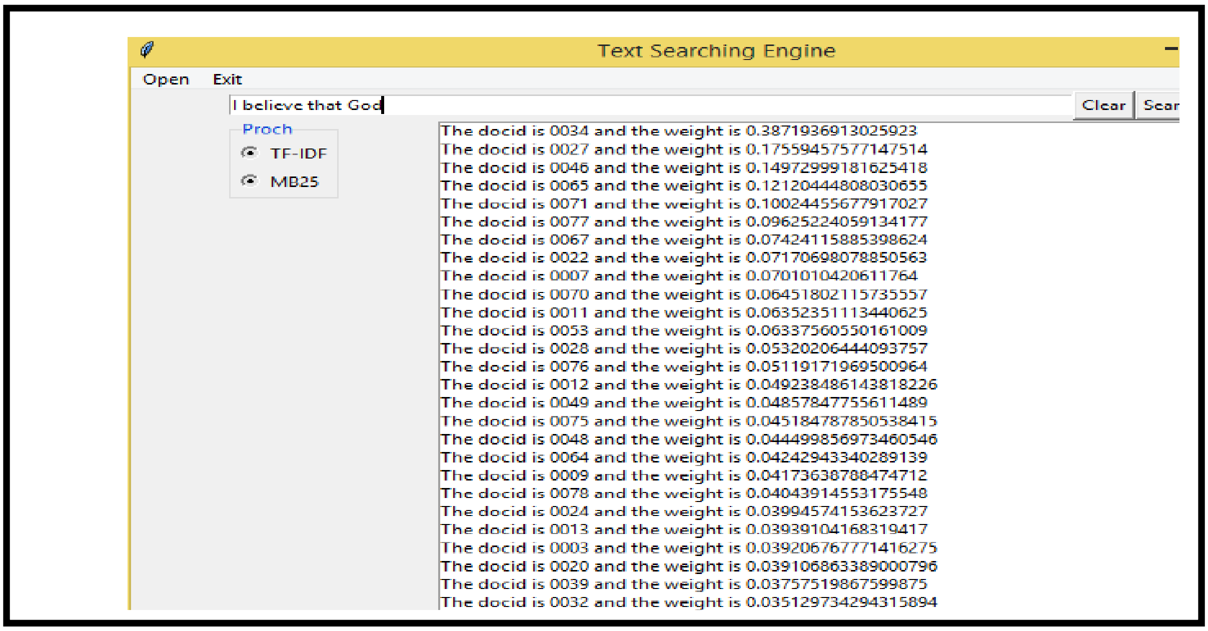
Figure 10 Retrieved document for two words query1

We calculate both of precision & recall we get:

Precision = 09/20 = 0,45

Recall =09/25 = 0,36

Now we will do the same steps but taking another example of query “ *I believe that God* “

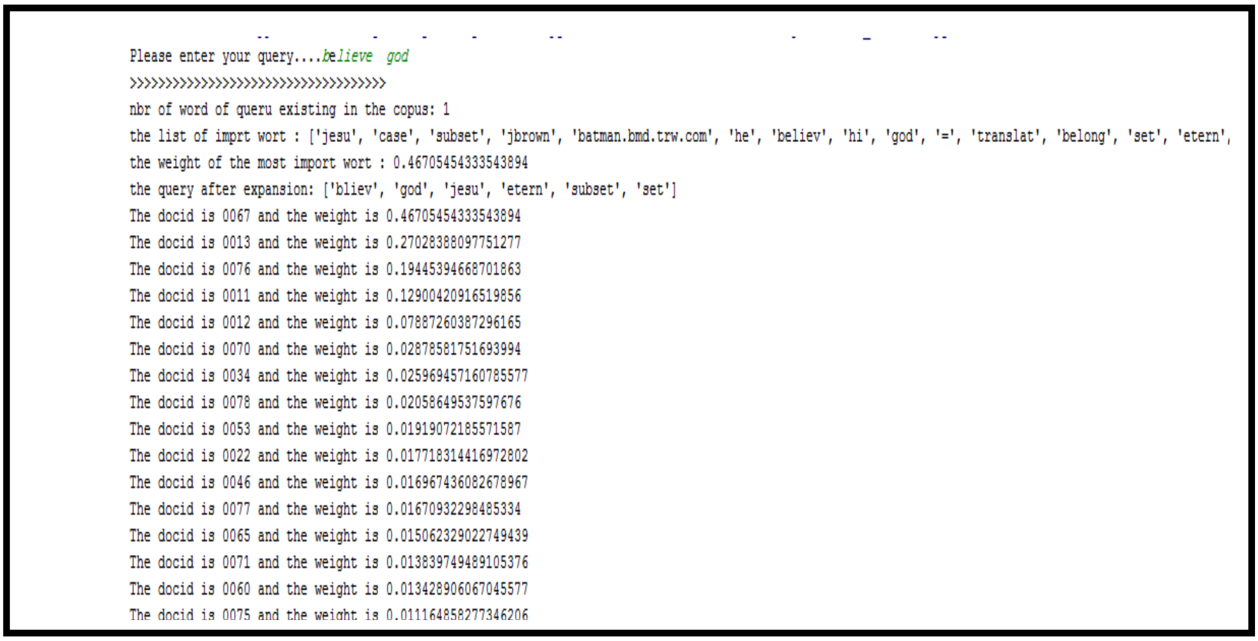


I believe that God

```
The docid is 0034 and the weight is 0.3871936913025923
The docid is 0027 and the weight is 0.1755945757714751 4
The docid is 0046 and the weight is 0.1497299918162541 8
The docid is 0065 and the weight is 0.1212044480803065 5
The docid is 0071 and the weight is 0.1002445567791702 7
The docid is 0077 and the weight is 0.0962522405913417 7
The docid is 0067 and the weight is 0.0742411588539862 4
The docid is 0022 and the weight is 0.0717069807885056 3
The docid is 0007 and the weight is 0.070101042061176 4
The docid is 0070 and the weight is 0.0645180211573555 7
The docid is 0011 and the weight is 0.0635235111344062 5
The docid is 0053 and the weight is 0.0633756055016100 9
The docid is 0028 and the weight is 0.0532020644409375 7
The docid is 0076 and the weight is 0.0511917196950096 4
The docid is 0012 and the weight is 0.04923848614381822 6
The docid is 0049 and the weight is 0.0485784775561148 9
The docid is 0075 and the weight is 0.04518478785053841 5
The docid is 0048 and the weight is 0.04449985697346054 6
The docid is 0064 and the weight is 0.0424294334028913 9
The docid is 0009 and the weight is 0.0417363878847471 2
The docid is 0078 and the weight is 0.0404391455317554 8
The docid is 0024 and the weight is 0.0399457415362372 7
The docid is 0013 and the weight is 0.0393910416831941 7
The docid is 0003 and the weight is 0.03920676777141627 5
The docid is 0020 and the weight is 0.03910686338900079 6
The docid is 0039 and the weight is 0.0375751986759987 5
The docid is 0032 and the weight is 0.03512973429431589 4
```

Figure 11 Retrieved document for the standard target query2

We searched the query: “*believe God* “, but we use now the extended query by our application, and we have the results shown in following figure.



```
Please enter your query....believe  god
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
nbr of word of queru existing in the copus: 1
the list of imprt wort : ['jesu', 'case', 'subset', 'jbrown', 'batman.bmd.trw.com', 'he', 'believ', 'hi', 'god', '=', 'translat', 'belong', 'set', 'etern',
the weight of the most import wort : 0.46705454333543894
the query after expansion: ['bliev', 'god', 'jesu', 'etern', 'subset', 'set']
The docid is 0067 and the weight is 0.46705454333543894
The docid is 0013 and the weight is 0.27028388097751277
The docid is 0076 and the weight is 0.19445394668701863
The docid is 0011 and the weight is 0.12900420916519856
The docid is 0012 and the weight is 0.07887260387296165
The docid is 0070 and the weight is 0.02878581751693994
The docid is 0034 and the weight is 0.025969457160785577
The docid is 0078 and the weight is 0.02058649537597676
The docid is 0053 and the weight is 0.01919072185571587
The docid is 0022 and the weight is 0.017718314416972802
The docid is 0046 and the weight is 0.016967436082678967
The docid is 0077 and the weight is 0.01670932298485334
The docid is 0065 and the weight is 0.015062329022749439
The docid is 0071 and the weight is 0.013839749489105376
The docid is 0060 and the weight is 0.013428906067045577
The docid is 0075 and the weight is 0.011164858277346206
```

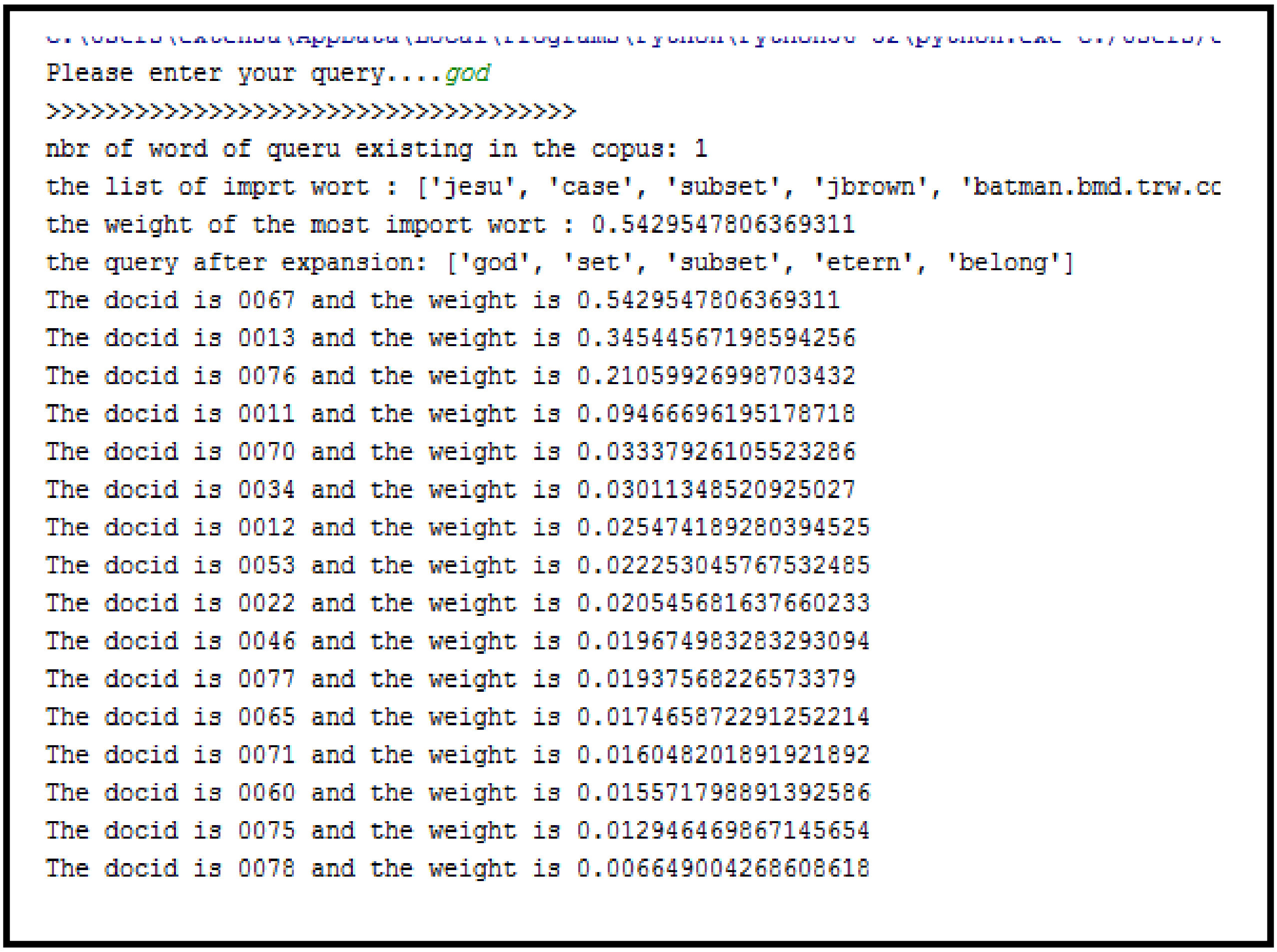
Figure 12 Retrieved document for two words query2

We calculate both of precision & recall we get:

Precision = 11/20 = 0,55

Recall =11/16 = 0,68

We will do another calculation, this time using the query “*God* “



```
C:\Users\extensa\AppData\Local\Programs\Python\Python36-32\python.exe C:/Users/e
Please enter your query....god
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
nbr of word of queru existing in the copus: 1
the list of imprt wort : ['jesu', 'case', 'subset', 'jbrown', 'batman.bmd.trw.cc
the weight of the most import wort : 0.5429547806369311
the query after expansion: ['god', 'set', 'subset', 'etern', 'belong']
The docid is 0067 and the weight is 0.5429547806369311
The docid is 0013 and the weight is 0.34544567198594256
The docid is 0076 and the weight is 0.21059926998703432
The docid is 0011 and the weight is 0.09466696195178718
The docid is 0070 and the weight is 0.03337926105523286
The docid is 0034 and the weight is 0.03011348520925027
The docid is 0012 and the weight is 0.025474189280394525
The docid is 0053 and the weight is 0.022253045767532485
The docid is 0022 and the weight is 0.020545681637660233
The docid is 0046 and the weight is 0.019674983283293094
The docid is 0077 and the weight is 0.01937568226573379
The docid is 0065 and the weight is 0.017465872291252214
The docid is 0071 and the weight is 0.016048201891921892
The docid is 0060 and the weight is 0.015571798891392586
The docid is 0075 and the weight is 0.012946469867145654
The docid is 0078 and the weight is 0.006649004268608618
```

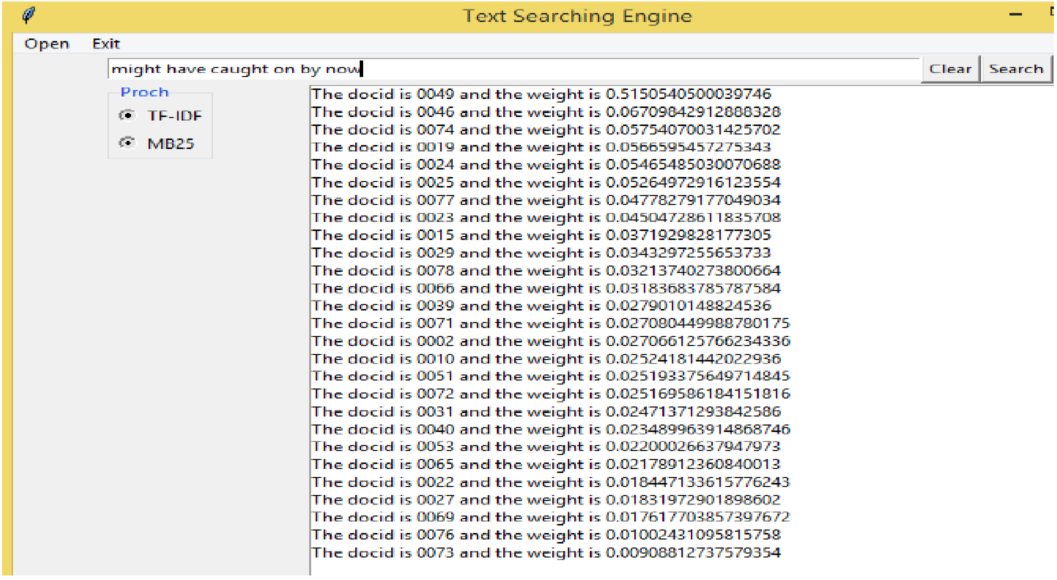
Figure 13 Retrieved document for one words query2

We calculate both of precision & recall we get:

Precision = 15/20 = 0,75

Recall =15/16 = 0,93

Now we will do the same steps but taking another example of query “*Might have caught on bay now* “

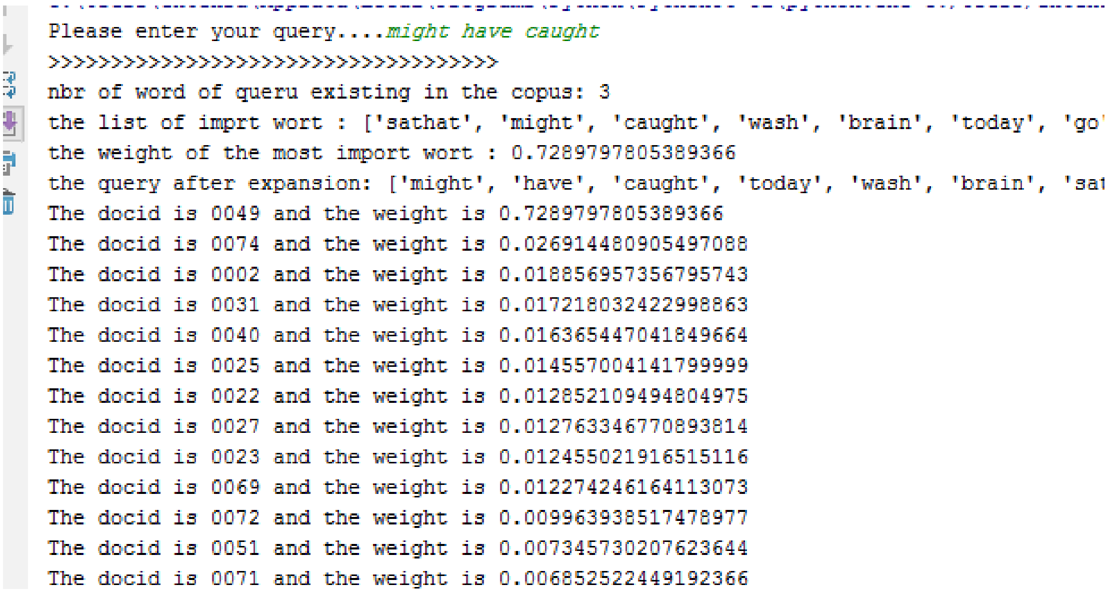


might have caught on by now

```
The docid is 0049 and the weight is 0.5150540500039746
The docid is 0046 and the weight is 0.06709842912888328
The docid is 0074 and the weight is 0.05754070031425702
The docid is 0019 and the weight is 0.0566595457275343
The docid is 0024 and the weight is 0.05465485030070688
The docid is 0025 and the weight is 0.05264972916123554
The docid is 0077 and the weight is 0.04778279177049034
The docid is 0023 and the weight is 0.04504728611835708
The docid is 0015 and the weight is 0.0371929828177305
The docid is 0029 and the weight is 0.0343297255653733
The docid is 0078 and the weight is 0.03213740273800664
The docid is 0066 and the weight is 0.03183683785787584
The docid is 0039 and the weight is 0.0279010148824536
The docid is 0071 and the weight is 0.027080449988780175
The docid is 0002 and the weight is 0.027066125766234336
The docid is 0010 and the weight is 0.02524181442022936
The docid is 0051 and the weight is 0.025193375649714845
The docid is 0072 and the weight is 0.025169586184151816
The docid is 0031 and the weight is 0.02471371293842586
The docid is 0040 and the weight is 0.023489963914868746
The docid is 0053 and the weight is 0.02200026637947973
The docid is 0065 and the weight is 0.02178912360840013
The docid is 0022 and the weight is 0.018447133615776243
The docid is 0027 and the weight is 0.01831972901898602
The docid is 0069 and the weight is 0.017617703857397672
The docid is 0076 and the weight is 0.01002431095815758
The docid is 0073 and the weight is 0.00908812737579354
```

Figure 14 Retrieved document for the standard target query3

We searched the query: “*Might have caught* “, but we use now the extended query by our application, and we have the results shown in following figure.



```
Please enter your query....might have caught
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
nbr of word of queru existing in the copus: 3
the list of imprt wort : ['sathat', 'might', 'caught', 'wash', 'brain', 'today', 'go'
the weight of the most import wort : 0.7289797805389366
the query after expansion: ['might', 'have', 'caught', 'today', 'wash', 'brain', 'sat
The docid is 0049 and the weight is 0.7289797805389366
The docid is 0074 and the weight is 0.026914480905497088
The docid is 0002 and the weight is 0.018856957356795743
The docid is 0031 and the weight is 0.017218032422998863
The docid is 0040 and the weight is 0.016365447041849664
The docid is 0025 and the weight is 0.014557004141799999
The docid is 0022 and the weight is 0.012852109494804975
The docid is 0027 and the weight is 0.012763346770893814
The docid is 0023 and the weight is 0.012455021916515116
The docid is 0069 and the weight is 0.012274246164113073
The docid is 0072 and the weight is 0.009963938517478977
The docid is 0051 and the weight is 0.007345730207623644
The docid is 0071 and the weight is 0.006852522449192366
```

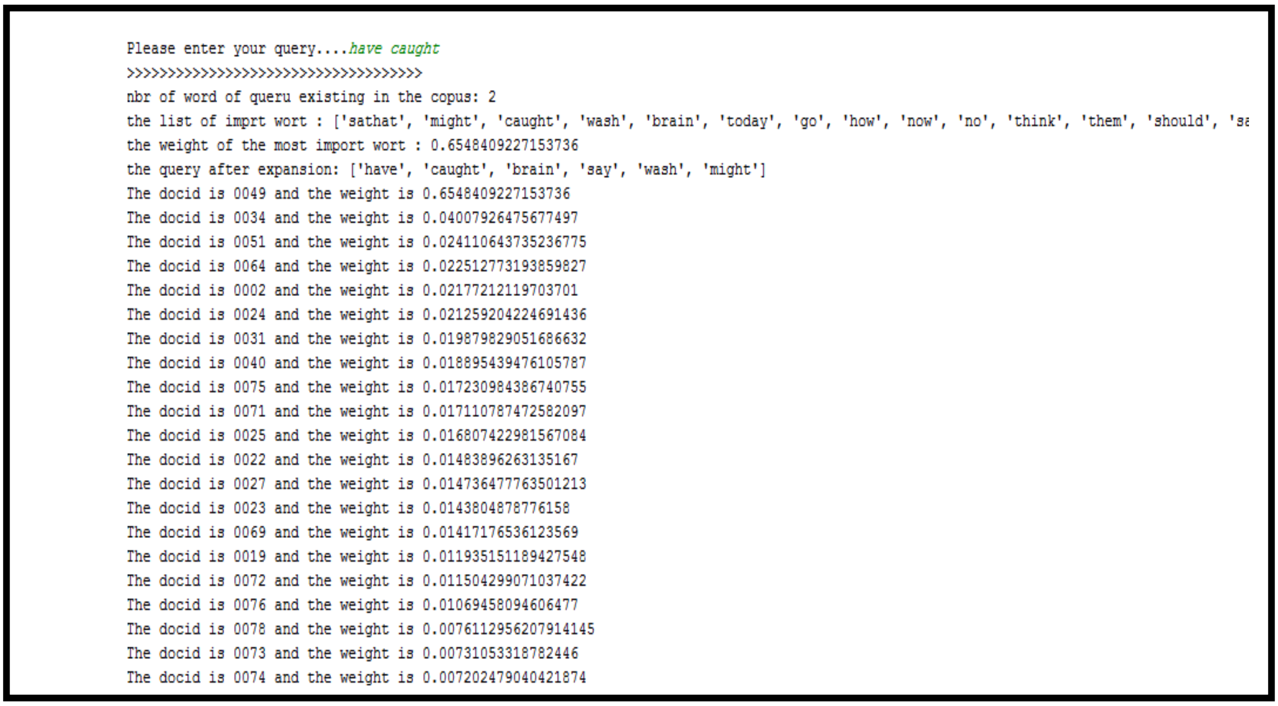
Figure 15 retrieved document for three words query3

We calculate both of precision & recall we get:

Precision = 14/20 = 0,7

Recall =14/16 = 0,87

We will do another calculation, this time using the query “*have caught* “



```
Please enter your query....have caught
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
nbr of word of queru existing in the copus: 2
the list of imprt wort : ['sathat', 'might', 'caught', 'wash', 'brain', 'today', 'go', 'how', 'now', 'no', 'think', 'them', 'should', 'sa
the weight of the most import wort : 0.6548409227153736
the query after expansion: ['have', 'caught', 'brain', 'say', 'wash', 'might']
The docid is 0049 and the weight is 0.6548409227153736
The docid is 0034 and the weight is 0.04007926475677497
The docid is 0051 and the weight is 0.024110643735236775
The docid is 0064 and the weight is 0.022512773193859827
The docid is 0002 and the weight is 0.02177212119703701
The docid is 0024 and the weight is 0.021259204224691436
The docid is 0031 and the weight is 0.019879829051686632
The docid is 0040 and the weight is 0.018895439476105787
The docid is 0075 and the weight is 0.017230984386740755
The docid is 0071 and the weight is 0.017110787472582097
The docid is 0025 and the weight is 0.016807422981567084
The docid is 0022 and the weight is 0.01483896263135167
The docid is 0027 and the weight is 0.014736477763501213
The docid is 0023 and the weight is 0.01438048787761 58
The docid is 0069 and the weight is 0.01417176536123569
The docid is 0019 and the weight is 0.011935151189427548
The docid is 0072 and the weight is 0.011504299071037422
The docid is 0076 and the weight is 0.01069458094606477
The docid is 0078 and the weight is 0.007611295620791 4145
The docid is 0073 and the weight is 0.00731053318782446
The docid is 0074 and the weight is 0.007202479040421874
```

Figure 16 retrieved document for two words query3

We calculate both of precision & recall we get:

Precision = 13/20 = 0,65

Recall =13/16 = 0,81

En consultons les résultats de ces exemples, on peut dire que la méthode est efficace dans une certaine mesure car le degré de corrélation entre la requête étendue et les résultats obtenus est augmenté .

3.5 Conclusion

In this chapter, we have approached the implementation of which we explained the query expention method based on word co-occurrence, and also used the Fireworks algorithm.
The results obtained are very satisfactory and encouraging, and we compare this method and that of our college of the past year we find that this year has had better results, let's hope that it will have other evolution in the future.

4 General conclusion.

Term co-occurrence data has been widely used in document search systems for identification

indexing terms similar to those that have been specified in a user request: these similar terms may then be used to augment the original query statement.

Our thesis is divided into three chapters, the first chapter introduces a general framework where we have presented the crucial points of the field of information retrieval by starting with basic concepts and going through the process of information retrieval and models of information retrieval and finish with a discussion of information retrieval.

In the second chapter we described query expansion and the main query reformulation techniques and then gave an in-depth explanation of co-occurrence and association metrics in this context.

In the third chapter 3 we proposed an implementation of the system that handles query expansion using co-occurrence, explained that it programming language used some and the corpus that used some in testing, and the steps that we followed for the implementation and solved the problem of using co-occurrence for expansion of a query, and then discussed the results.

From the results obtained we can say that: this method is quite efficient because in all cases, it increases the degree of correlation between the extended query and the results obtained.

5 Bibliographie

[01] Abassi Maftah, Un modèle de reformulation des requêtes pour la recherche d'information sur le Web, master ,univ-Ouargla, dec2013.pp 4.

[02] Hammache Arezki , Recherche d'Information : un modèle delangue combinant mots simples et mots composés, doctorat, Univ-Tizi Ouzou,. Pp 8-19

[03] Baeza-Yates, R., Ribeiro-Neto, B. A. Modern Information Retrieval. Pearson Education Ltd., Harlow, UK, 2nd edn, 2011

[04] Ren, F., Fan, L., Nie, J-Y. SAAK Approach: How to Acquire Knowledge in an Actual Application System. International Conference on Artificial Intelligence and Soft Computing, Honolulu , 1999. pp.136-140.

[05] Jacquemin, C., Daille, B., Royanté, J., and Polanco, X. In vitro evaluation of a program for machine-aided indexing. Inf. Process. Manage. 2002. pp. 765-792

[06] Baziz M., Conceptual Indexing Guided By Ontology For Information Research. Doctoral thesis in Computer Science from the Paul Sabatier University of Toulouse, 2005. pp 113-136.

[07] Dominich, S. Mathematical Foundations of Information Retrieval. Kluwer Academic Publishers, Dordrecht,Boston, London, 2001.

[08] Kraft, D. H. and BueIl, D. A. Fuzzy sets and generalized Boolean retrieval systems. International Journal on Man-Machine Studies,1983. pp. 49-56.

[09] Radecki, T. Fuzzy set theoretical approach to document retrieval. Information Processing and Management, the extended boolean model, 1979. pp. 247-259.

[10]Salton, G. The smart Retrieval System: Experiments in Automatic Document Processing. Prentice-Hall, 1971.

[11] Lv, Y., Zhai. C. Positional Relevance Model for Pseudo-Relevance Feedback. Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval, 2010. pp. 247-259.

[12] Robertson, S.E. , S. Walker. On relevance weights with little relevance information. Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval, 1997. pp. 16–24.

[13] Singhal, A., Salton, G., Mitra, M., Buckley, C. Document length normalization. Information Processing and Management, 1996. pp. 619–633.

[14] Van Rijsbergen, C. J. Information retrieval. London: Butterworth, 1979.

[15] Wong, S., Ziarko, W., Wong, P. Generalized vector space model in information retrieval. Proceedings of the 8th ACM SIGIR Conference on Research and Development in information retrieval, New-York, USA, 1985. pp. 18–25.

[16] Berry, M.W., Dumais, S.T., O'Brien, G. W. 1995. Using linear algebra for intelligent information retrieval. SIAM Rev, 1995. pp. 573-595.

[17] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas and Richard A.Harshman"Indexing by Latent Semantic Analysis". In Journal of the American Society of Information Science, 1990. pp. 391-407 .

[18] Jelinek, F. Statistical Methods for Speech Recognition. MIT Press, Cambridge, MA, 1998

[19] Manning, D., Schütze, H. Foundations of Statistical Natural Language Processing. MIT Press, 2000.

[20] Lafferty, J., Zhai, C. Document language models, query models, and risk minimization for informationretrieval. In W.B. Croft, D.J. Harper, D.H. Kraft, & J. Zobel (Eds.).Proceedings of the 24th annual international ACM-SIGIR conference on research and development in information retrieval ,2001. pp.111-119.

[21] Lavrenko, V., & Croft, W. B. Relevance-based language models. In W.B. Croft, D.J. Harper, D.H. Kraft, & J. Zobel (Eds.). Proceedings of the 24th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, New Orleans, Louisiana, 2001. pp.120-127.

[22] Ponte, J.M., Croft, W. B. A language modeling approach to information retrieval. Proceedings of the 21stannual international ACM SIGIR conference on Research and development in information retrieval, 1998. pp. 275-281.

[23] Petrovic S, Snajder J, Dalbelo-Basic B, Kolar M. Comparison of collocation extraction measures for document indexing. Jornal of Computing and Information Technolgie , 2006.pp 321–327.

[24] Zhai, C., Lafferty, J. A study of smoothing methods for language models applied to ad hoc information retrieval. In W.B. Croft, D.J. Harper, D.H. Kraft, & J. Zobel (Eds.), Proceedings of the 24th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, 2001. pp. 334-342.

[25] Bouabdellah ,L & Benmansour ,A ,Expansion de requête pour un système de recherche d'information par croisement de langue, master,université,mais 2012.Pp 28.

[26] Stéphane Clerc, Expansion de requêtes spatio-thématiques dans un service de catalogage, Mémoire de Stage de Master, Juillet 2006 .Pp 26_29.

[27] C.D. Manning & H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[28] Y. Choueka. Looking for Needles in a Haystack or Locating Interesting Collocational Expressions in Large Textual Databases. *Proceedings of the RIAO '88 Conference on User Oriented Content-Based Text and Image Handling*, Cambridge, 1988. , pp. 1-15.

[29] Simon Réhel,Catégorisation automatique de textes de et cooccurence de mots provenant de document non étiquetés, janv 2005;pp 57-67.

[30] Koraichi Alima.Query Expansion Using Fireworks Algorithm. Master Univer Ouargla.2019, pp 28-35.

6Webographie

[W01] S. Evert. Association measures. http://www.collocations.de/AM/

[W02] https://fr.wikipedia.org/wiki/Python_(langage)

[W03] Open Classroom. https://openclassrooms.com/courses/apprenez-a-programmer enpython/qu-est-ce-que-python

[W04] https://www.digitalocean.com/community/tutorials/understanding-lists-in-python-3

[W05] https://www.tutorialspoint.com/python/python_tuples

[W06] http://www.xavierdupre.fr/app/ensae_teaching_cs/helpsphinx/notebooks/code_liste_tu ple.html

[W07] http://qwone.com/~jason/20Newsgroups/