Improving Text Retrieval Accuracy by Using a Minimal Relevance Feedback

Francesco Colace[1], Massimo De Santo[1], Luca Greco[1], and Paolo Napoletano[2,⋆]

[1] Department of Electronic Engineering and Computer Engineering
University of Salerno, 84084 Fisciano, Italy
{fcolace,desanto,lgreco}@unisa.it
[2] DISCo (Department of Informatics, Systems and Communication)
University of Milan, Bicocca Viale Sarca 336
20126 Milan, Italy
napoletano@disco.unimib.it

Abstract. In this paper we have demonstrated that the accuracy of a text retrieval system can be improved if we employ a query expansion method based on explicit relevance feedback that expands the initial query with a structured representation instead of a simple list of words. This representation, named a mixed *Graph of Terms*, is composed of a directed and an a-directed subgraph and can be automatically extracted from a set of documents using a method for *term extraction* based on the *probabilistic Topic Model*. The evaluation of the method has been conducted on a web repository collected by crawling a huge number of web pages from the website ThomasNet.com. We have considered several topics and performed a comparison with a baseline and a less complex structure that is a simple list of words.

Keywords: Text Retrieval, Query Expansion, Probabilistic Topic Model.

1 Introduction

The widespread use of digital technologies in all aspects of daily life has improved knowledge about the behavior of the individual entities involved in a complex system. This has increased both conscious and unconscious collaborative modes of information/knowledge sharing/exchange: consider information systems like Amazon, e-bay, Twitter, Facebook, Wikis, e-marketplaces, Myspace, blogs and so on.

As a consequence, Intelligent Systems have been introduced to assist and augment this natural social process and so help people sift through available books, articles, web pages, movies, music, restaurants, jokes, grocery products, and so forth to find the most interesting and valuable information for them. All the existing intelligent systems are based on data mining methods which comprise also collaborative filtering and text mining techniques. These methods are either memory-based, model-based, content-based or hybrids. While the memory and model-based methods make use of the records contained in structured data (User X is quite interested inx product Y) to make predictions, the content-based methods analyze the content of textual information to match and find patterns. Leaving aside the memory and model-based methods, we focus only on the

⋆ Corresponding author.

A. Fred et al. (Eds.): IC3K 2011, CCIS 348, pp. 126–140, 2013.
© Springer-Verlag Berlin Heidelberg 2013

content-based ones that, thanks to the massive use of the reviewing of the items activity by people, are becoming of great interest.

The content analysis is possible thanks to the findings obtained in the fields of text mining, text classification, text categorization as well as of sentiment analysis and detection, thus exploiting all the text retrieval theories. In the field of text retrieval the main problem is: "How can a computer tell which documents are relevant to the query, and, more importantly, which results are more relevant than others?"

There is of course no definitive answer, and all the existing approaches to solve this problem consider a different Information Retrieval model to represent a document in the document collection. We can divide all the existing methods into several categories: set-theoretic (including boolean) models, algebraic models and probabilistic models [9][1]. Although each method has its own properties, there is a common denominator: the *bag of words* approach to document representation.

The "bag of words" assumption claims that a document can be considered as a feature vector where each element in the vector indicates the presence (or absence) of a word, so that the information on the position of that word within the document is completely lost [9].

The elements of the vector can be weights and computed in different ways, for instance *BM25*, *tf-idf*, etc., thus a document can be considered as a list of weighted features (that are words). A query is considered as a document and so it is represented as a vector of weighted words.

The *term frequency-inverse document (tf-idf)* model is a weighting model used to give weights to the terms in a document collection by measuring how often a term is found within a document (*term frequency*), offset by how often the term is found within the entire collection (*inverse document frequency*).

In this paper we argue that a vector of weighted words, due to the inherent ambiguity of language (polysemy etc.), is not capable of discriminating between documents in the case of ad-hoc text retrieval tasks. Here the aim is to find the documents that best match the performed query (that is a topic). The ambiguity, in fact, can be reduced if we give more importance to words that convey concepts and that contribute to specify a topic, and if we assign less importance to those words that contribute to specify concepts and that, due to the fact that they can be more plausibly shared between concepts, can increase the ambiguity. This leads to a hierarchical structure that we call a mixed *Graph of Terms* and that can be automatically extracted from a set of documents using a global method for term extraction based on the Latent Dirichlet Allocation model implemented as the Probabilistic Topic Model.

We have employed the mixed *Graph of Terms* in a query expansion method based on explicit relevance feedback that expands the initial query with this new structured query representation. The evaluation of the method has been conducted on a web repository collected by crawling a huge number of web pages from the website ThomasNet.com. We have considered several topics and performed a comparison with a less complex structure that is a simple list of words. The results obtained, independently of the context, show that a more complex representation is capable of retrieving a greater number of relevant documents achieving a mean average precision of about 50%.

2 Query Expansion Techniques

It is well documented that the query length in typical information retrieval systems is rather short (usually two or three words [16], [15]) which may not be long enough to avoid the inherent ambiguity of language (polysemy etc.), and which makes text retrieval systems, that rely on a term-frequency based index, suffer generally from low precision, or low quality of document retrieval.

In turn, the idea of taking advantage of additional knowledge, by expanding the original query with other topic-related terms, to retrieve relevant documents has been largely discussed in the literature, where manual, interactive and automatic techniques have been proposed [12][9][1]. The idea behind these techniques is that, in order to avoid ambiguity, it may be sufficient to better specify "the meaning" of what the user has in mind when performing a search, or in other words "the main concept" (or a set of concepts) of the preferred topic in which the user is interested. A better specialization of the query can be obtained with additional knowledge, that can be extracted from *exogenous* (e.g. ontology, WordNet, data mining) or *endogenous* knowledge (i.e. extracted only from the documents contained in the repository) [2,22,9].

In this paper we focus on those techniques which make use of the "Relevance Feedback" (in the case of endogenous knowledge) which takes into account the results that are initially returned from a given query and so uses the information about the relevance of each result to perform a new expanded query. In the literature we can distinguish between three types of procedures for the assignment of the relevance: explicit feedback, implicit feedback, and pseudo feedback [1]. The feedback is obtained from assessors (or other users of a system) indicating the relevance of a document retrieved for a query. If the assessors know that the feedback provided is interpreted as relevance judgments then the feedback is considered as explicit; otherwise it is implicit. On the contrary, the pseudo relevance feedback automates the manual part of the relevance labeling by assuming that the top "n" ranked documents after the initial query are relevant and so finally performing relevance feedback as before under this assumption.

Most existing methods, due to the fact that the human labeling task is enormously annoying and time consuming [17,25], make use of pseudo relevance feedback. Nevertheless, fully automatic methods suffer from obvious errors when the initial query is intrinsically ambiguous. As a consequence, in recent years, some hybrid techniques have been developed which take into account a minimal explicit human feedback [21,11] and use it to automatically identify other topic related documents. The performance achieved by these methods is usually medium with a mean average precision of about 30% [21].

However, whatever the technique that selects the set of documents representing the feedback, the expanded terms are usually computed by making use of well known approaches for term selection such as Rocchio, Robertson, CHI-Square, Kullback-Lieber etc [23][7]. In this case the reformulated query consists in a simple (sometimes weighted) list of words.

Although such term selection methods have proven their effectiveness in terms of accuracy and computational cost, several more complex alternative methods have been proposed. In this case, they usually consider the extraction of a structured set of words so that the related expanded query is no longer a list of words, but a weighted set of clauses combined with suitable operators [5], [10], [18].

3 The Proposed Approach

The *vector of features* needed to expand the query is obtained as a result of an interactive process between the user and system. The user initially performs a retrieval on the dataset $\mathcal{D}$ by inputting a query to the system and later identifies a small set Ω_r of relevant documents from the hit list of documents returned by the system, that is considered as the training set $\Omega_r = \{\mathbf{d}_1, \ldots, \mathbf{d}_{|\Omega|}\} \subset \mathcal{D}$ (the relevance feedback).

Existing query expansion techniques mostly use the relevance feedback of both relevant and irrelevant documents. Usually they obtain the term selection through the scoring function proposed in [24], [7] which assigns a weight to each term depending on its occurrence in both relevant and irrelevant documents. Differently, in this paper we do not consider irrelevant documents.

Precisely, the *vector of features*, that we call the mixed *Graph of Terms*, can be automatically extracted from a set of documents Ω_r using a method for *term extraction* based on the *Latent Dirichlet Allocation* model [4] implemented as the *Probabilistic Topic Model* [13].

The general idea of this paper is supported by previous works [20] that have confirmed the potential of supervised clustering methods for term extraction, also in the case of query expansion [6,19].

3.1 Data Preparation

Texts can not be directly interpreted by a search engine and for this reason, an indexing procedure that maps a text into a compact representation of its content must be uniformly applied to the entire corpus and to the training set. Let us consider the case of Ω_r.

Each document can be represented, following the *Vector Space Model* [9], as a vector of term *weights*

$$\mathbf{d}_m = \{w_{1m}, \ldots, w_{|\mathcal{T}|m}\},$$

where $\mathcal{T}$ is the set of *terms* (also called *features*) that occur at least once in at least one document of Ω_r, and $0 \leq w_{nm} \leq 1$ represents how much a term t_n contributes to a semantics of document $\mathbf{d}_m$.

If we choose to identify terms with words, we have the *bag of words* assumption, that is $t_n = v_n$, where v_n is one of the words of a vocabulary. The *bag of words* assumption claims that each w_{nm} indicates the presence (or absence) of a word, so that the information on the position of that word within the document is completely lost [9].

To determine the weight w_{nm} of term t_n in a document $\mathbf{d}_m$, the standard tf-idf (*term frequency-inverse document frequency*) function can be used [26], defined as:

$$\text{tf-idf}(t_n, \mathbf{d}_m) = N(t_n, \mathbf{d}_m) \cdot \log \frac{|\Omega_r|}{N_{\Omega_r(t_n)}} \tag{1}$$

where $N(t_n, \mathbf{d}_m)$ denotes the number of times t_n occurs in $\mathbf{d}_m$, and $N_{\Omega_r(t_n)}$ denotes the document frequency of term t_n, i.e. the number of documents in Ω_r in which t_n occurs.

In order for the weights to fall within $[0, 1]$ interval and for the documents to be represented by vectors of equal length, the weights resulting from tf-idf are usually normalized by cosine normalization, given by:

$$w_{nm} = \frac{\text{tf-idf}(t_n, \mathbf{d}_m)}{\sqrt{\sum_{n=1}^{|\mathcal{T}|}(\text{tf-idf}(t_n, \mathbf{d}_m))^2}} \tag{2}$$

In this paper, before indexing, we have performed the removal of function words (i.e. topic-neutral words such as articles, prepositions, conjunctions, etc.) and we have performed the stemming procedure[1] (i.e. grouping words that share the same morphological root).

Once the indexing procedure has been performed, we have a matrix $|\mathcal{T}| \times |\Omega_r|$ of real values instead of the training set Ω_r. The same procedure is applied to the entire corpus $\mathcal{D}$.

3.2 A Mixed Graph of Terms

In this paper we have used a *global* method for *feature transformation* that considers pairs of words instead of single words as basic features thus obtaining a new space $\mathcal{T}_p$ of features. The dimensionality of such a new space is very high, much higher than $|\mathcal{T}|$, in fact: $|\mathcal{T}_p| \propto |\mathcal{T}|^2$. For this reason we need to reduce the transformed space in order to obtain a new space $\mathcal{T}_{sp}$ such that $|\mathcal{T}_{sp}| \ll |\mathcal{T}_p|$.

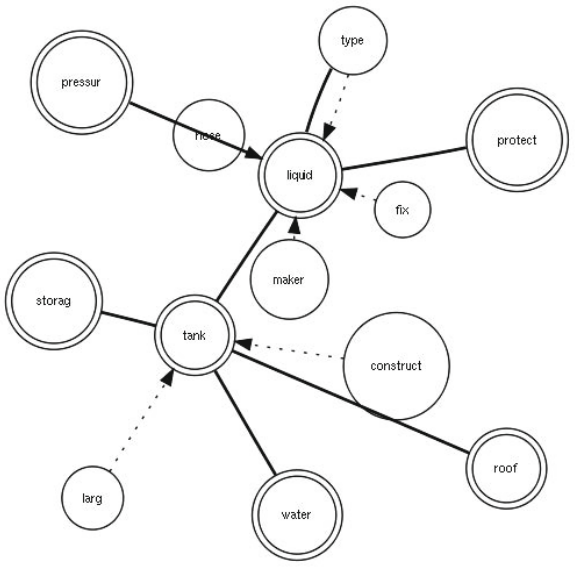
The method used to select the most representative pairs of words is based on the *Latent Dirichlet Allocation* model [4] implemented as the *Probabilistic Topic Model* [13] and this is the core of a new representation, named the mixed *Graph of Terms*, that consists of related pairs of words. The graph contains two kinds of relations between words, directed and undirected, and for this reason it is called *mixed*.

In the graph we can find several clusters of words and each cluster contains a set of words v_s that specifies, through a directed weighted edge, a special word, that we have named the *concept*, r_i, that is the centroid of such a cluster. The weight ρ_{is} can measure how far a word is related to a concept, or how much we need such a word to specify that concept, and it can be considered as a probability: $\rho_{is} = P(r_i|v_s)$. The resulting structure is a subgraph rooted on r_i (see fig. 1(a)).

Moreover, special words, namely *concepts*, can be linked together through undirected weighted edges, so forming a subgraph of pairs of centroids. The weight ψ_{ij} can be considered as the degree of semantic correlation between two concepts and it can be considered as a probability: $\psi_{ij} = P(r_i, r_j)$ (see fig. 1(a)).

Considering that each concept is a special word, we can say that the graph contains directed and undirected pairs of features that are all lexically denoted as words. Given the training set Ω_r of documents, the proposed method, through a learning procedure, selects a subset of pairs obtaining a number of pairs $|\mathcal{T}_{sp}| \ll |\mathcal{T}_p|$. In this way, the term extraction procedure is obtained by firstly computing all the semantic relatednesses

[1] Stemming has sometimes been reported to hurt effectiveness. However the recent tendency has been to adopt it, as it reduces both the dimensionality of the feature space and the stochastic dependence between terms.



(a) (b)

Fig. 1. *Vector of features* for the topic *Storage Tanks*. 1(a) A mixed *Graph of Terms*. 1(b) A *List of Terms*.

between words and concepts, that is ρ_{is} and ψ_{ij}, and secondly selecting the right subset of pairs from all the possible ones. Before explaining in detail the learning procedure of a graph, we would like to highlight some aspects of this representation.

3.3 Graph and Document Representation in the Space $\mathcal{T}_{sp}$

A mixed *Graph of Terms* $\mathbf{g}$ can be viewed, following the *Vector Space Model* [9], as a vector of features t_n:

$$\mathbf{g} = \{b_1, \ldots, b_{|\mathcal{T}_{sp}|}\},$$

where $|\mathcal{T}_{sp}|$ represents the number of pairs and each feature $t_n = (v_i, v_j)$ can be a *word/concept* or *concept/concept* pair. The weight b_n is named the *boost* factor and is equal to ψ_{ij} for both *word/concept* or *concept/concept* pairs.

Moreover, by following this approach, also each document of a corpus $\mathcal{D}$ can be represented in terms of pairs:

$$\mathbf{d}_m = (w_{1m}, \ldots, w_{|\mathcal{T}_{sp}|m}),$$

where w_{nm} is such that $0 \leq w_{nm} \leq 1$ and represents how much term $t_n = (v_i, v_j)$ contributes to a semantics of document $\mathbf{d}_m$. The weight is calculated thanks to the tf-idf model applied to the pairs represented through t_n:

$$w_{nm} = \frac{\text{tf-idf}(t_n, \mathbf{d}_m)}{\sqrt{\sum_{n=1}^{|\mathcal{T}_{sp}|} (\text{tf-idf}(t_n, \mathbf{d}_m))^2}} \tag{3}$$

4 Graph Learning

A graph $\mathbf{g}$ is well determined through the learning of the weights, the *Relations Learning* stage, and through the learning of three parameters, the *Structure Learning* stage, that are $\Lambda = (H, \tau, \mu)$ which specify the shape, namely the structure, of the graph. In fact, we have:

1. H: the number of concepts (namely the number of clusters) of the set of documents;
2. μ_i: the threshold that establishes for each concept the number of edges of the directed subgraph, and so the number of *concept/word* pairs of the corpus . An edge between the word s and the concept i can be saved if $\rho_{is} \geq \mu_i$. To simplify the formulation, we assume that $\mu_i = \mu, \forall i$;
3. τ: the threshold that establishes the number of edges of the undirected subgraph, and so the number of *concept/concept* pairs of the corpus. An edge between the concept i and concept j can be saved if $\psi_{ij} \geq \tau$.

4.1 Relations Learning

Due to the fact that each concept is lexically represented by a word of the vocabulary, then we have that $\rho_{is} = P(r_i|v_s) = P(v_i|v_s)$, and $\psi_{ij} = P(r_i, r_j) = P(v_i, v_j)$.

Considering that $P(v_i, v_j) = P(v_i|v_j)P(v_j)$, it is necessary, to learn all the relations between words, to compute the joint, or the conditional, probability $\forall i, j \in \{1, \cdots, |\mathcal{T}|\}$ and each $P(v_j)$ $\forall j$.

We show here that the exact calculation of $P(v_j)$ and the approximation of the joint, or conditional, probability can be obtained through a smoothed version of the generative model introduced in [4] called Latent Dirichlet Allocation (LDA), which makes use of Gibbs sampling [13].

The original theory introduced in [13] mainly asserts a semantic representation in which documents are represented in terms of a set of probabilistic topics z. Formally, we consider a word u_m of the document $\mathbf{d}_m$ as a random variable on the vocabulary $\mathcal{T}$ and z as a random variable representing a topic between $\{1, \cdots, K\}$. The probability distribution of a word within a document $\mathbf{d}_m$ of the corpus can be obtained as:

$$P(u_m) = \sum_{k=1}^{K} P(u_m|z = k, \beta_k)P(z = k|\theta_m). \tag{4}$$

The generation of a document $\mathbf{d}_m$ can be obtained considering the generation of each word of the document. To obtain a word, the model considers three parameters assigned: α, η and the number of topics K. Given these parameters, the model chooses θ_m through $P(\theta|\alpha) \sim$ Dirichlet(α), the topic k through $P(z|\theta_m) \sim$ Multinomial(θ_m) and $\beta_k \sim$ Dirichlet(η). Finally, the distribution of each word given a topic is $P(u_m|z, \beta_z) \sim$ Multinomial(β_z).

As we have already discussed, we have used a smoothed version of Latent Dirichlet Allocation (LDA), which makes use of Gibbs sampling. The results obtained by performing this algorithm on a set of documents Ωr are two matrixes:

1. the *words-topics* matrix that contains $|\mathcal{T}| \times K$ elements representing the probability that a word v_i of the vocabulary is assigned to topic k: $P(u = v_i|z = k, \beta_k)$;
2. the *topics-documents* matrix that contains $K \times |\Omega r|$ elements representing the probability that a topic k is assigned to some word token within a document $\mathbf{d}_m$: $P(z = k|\theta_m)$.

In the same way, the joint probability between two words u_m and y_m of a document $\mathbf{d}_m$ of the corpus can be obtained by assuming that each pair of words is represented in terms of a set of topics z and then:

$$P(u_m, y_m) = \sum_{k=1}^{K} P(u_m, y_m|z = k, \beta_k)P(z = k|\theta_m) \tag{5}$$

Note that the exact calculation of Eq. 5 depends on the exact calculation of $P(u_m, y_m|z = k, \beta_k)$ that can not be directly obtained through LDA. For this reason, we have introduced an approximation that considers words in a document as conditionally independent given a topic. In this way Eq. 5 can be written as:

$$P(u_m, y_m) \simeq \sum_{k=1}^{K} P(u_m|z = k, \beta_k)P(y_m|z = k, \beta_k)P(z = k|\theta_m). \tag{6}$$

Note that Eq. 4 gives the probability distribution of a word u_m within a document $\mathbf{d}_m$ of the corpus. To obtain the probability distribution of a word u independently of the document we need to sum over the entire corpus:

$$P(u) = \sum_{m=1}^{M} P(u_m)\delta_m \tag{7}$$

where δ_m is the prior probability for each document ($\sum_{m=1}^{|\Omega r|} \delta_m = 1$).

In the same way, if we consider the joint probability distribution of two words u and y, we obtain:

$$P(u, y) = \sum_{m=1}^{M} P(u_m, y_v)\delta_m \tag{8}$$

Concluding, once we have $P(u)$ and $P(u, y)$ we can compute $P(v_i) = P(u = v_i)$ and $P(v_i, v_j) = P(u = v_i, y = v_j)$, $\forall i, j \in \{1, \cdots, |\mathcal{T}|\}$ and so the relations learning can be totally accomplished.

4.2 Structure Learning

Given a set of documents, once each ψ_{ij} and ρ_{is} is known $\forall i, j, s$, letting the parameters $\Lambda_t = (H, \tau, \mu)_t$ assume a different set of values, we can observe a different structure of the graph $\mathbf{g}_t$ (here t is representative of different parameter values).

A way to learn the structure of the graph is to use an optimization based algorithm that searches for the best set of parameters Λ_t. In this case we need a scoring function and a searching strategy [3].

As we have previously seen, a $\mathbf{g}_t$ is a vector of features $\mathbf{g}_t = \{b_{1t}, \ldots, b_{|\mathcal{T}_{sp}|t}\}$ in the space $\mathcal{T}_{sp}$ and each document of the training set Ω_r, as well as the documents of the corpus $\mathcal{D}$, can be represented as a vector $\mathbf{d}_m = (w_{1m}, \ldots, w_{|\mathcal{T}_{sp}|m})$ in the space $\mathcal{T}_{sp}$. A possible scoring function is the cosine similarity between these two vectors:

$$\mathcal{S}(\mathbf{g}_t, \mathbf{d}_m) = \frac{\sum_{n=1}^{|\mathcal{T}_{sp}|} b_{nt} \cdot w_{nm}}{\sqrt{\sum_{n=1}^{|\mathcal{T}_{sp}|} b_{nt}^2} \cdot \sqrt{\sum_{n=1}^{|\mathcal{T}_{sp}|} w_{nm}^2}} \tag{9}$$

and thus the optimization procedure would consist in searching for the best set of parameters Λ_t such that the cosine similarity is maximized $\forall \mathbf{d}_m$.

By following this approach, the best $\mathbf{g}_t$ for the set of documents Ω_r is the one that produces the maximum score attainable for each of the documents when the same graph is used as a vector of features to measure the similarity of a set containing just those documents which have fed the graph builder. As a consequence, we obtain a score for each document $\mathbf{d}_m$ and then we have

$$\mathbf{S}_t = \{\mathcal{S}(\mathbf{g}_t, \mathbf{d}_1), \cdots, \mathcal{S}(\mathbf{g}_t, \mathbf{d}_{|\Omega_r|})\},$$

where each score depends on the specific set $\Lambda_t = (H, \tau, \mu)_t$.

To compute the best value of Λ we can maximize the score value for each document, which means that we are looking for the graph which best describes each document of the repository from which it has been learned. It should be noted that such an optimization maximizes at the same time all $|\Omega_r|$ elements of $\mathbf{S}_t$.

Alternatively, in order to reduce the number of the objectives being optimized, we can at the same time maximize the mean value of the scores and minimize their standard deviation, which turns a multi-objective problem into a two-objective one. Additionally, we can reformulate the latter problem by means of a linear combination of its objectives, thus obtaining a single objective function, i.e., *Fitness* ($\mathcal{F}$), which depends on Λ_t,

$$\mathcal{F}(\Lambda_t) = E\left[\mathbf{S}_t\right] - \sigma\left[\mathbf{S}_t\right],$$

where E is the mean value of all the elements of $\mathbf{S}_t$ and σ_m is the standard deviation. Summing up, the parameters learning procedure is represented as follows,

$$\Lambda^* = \underset{t}{\operatorname{argmax}}\{\mathcal{F}(\Lambda_t)\}.$$

We will see next how we have performed the searching strategy phase.

Since the space of possible solutions could grow exponentially, we have considered[2] $|\mathcal{T}_{sp}| \leq 100$. Furthermore, we have reduced the remaining space of possible solutions by applying a clustering method, that is the *K-means* algorithm, to all ψ_{ij} and ρ_{is} values, so that the optimum solution can be exactly obtained after the exploration of the entire space.

This reduction allows us to compute a graph from a repository composed of a few documents in a reasonable time (e.g. for 3 documents it takes about 3 seconds with a Mac OS X based computer, 2.66 GHz Intel Core i7 CPU and a 8GB RAM). Otherwise,

[2] This number is usually employed in the case of Support Vector Machines.

we would need an algorithm based on a random search procedure in big solution spaces. For instance, Evolutionary Algorithms would be suitable for this purpose, but would provide a slow performance. In fig. 1(a) we can see an example of a graph learned from a set of documents labeled as topic *Storage tanks*.

4.3 Extracting a Simpler Representation from the Graph

From the mixed *Graph of Terms* we can select different subsets of features so obtaining a simpler representation (see fig. 1(b)). Before discussing this in detail, we would recall that $\psi_{ij} = P(v_i, v_j)$ and $\rho_{is} = P(v_i|v_s)$ are computed through the topic model which also computes the probability for each word $\eta_s = P(v_s)$.

We can obtain the simplest representation by selecting from the graph all distinct terms and associating to each of them its weight $\eta_s = P(v_s)$. We name this representation the *List of Terms* ($\mathbf{w}$), see fig. 1(b).

4.4 Consideration on the Method

It is important to make clear that the mixed Graph of Terms can not be considered as a co- occurrence matrix. In fact, the core of the graph is the probability $P(v_i, v_j)$, which we regard as a word association problem, that in the topic model is considered as a problem of prediction: given that a cue is presented, which new words might occur next in that context? It means that the model does not take into account the fact that two words occur in the same document, but that they occur in the same document when a specific topic (and so a context) is assigned to that document [13].

Furthermore, in the field of statistical learning, a similar structure has been introduced, named the Hierarchical Mixture of Experts [14]. Such a structure is employed as a method for supervised learning and it is considered as a variant of the well known tree-based methods. The similarity between such a structure and the proposed graph can be obtained by considering the "experts" as "concepts".

Notwithstanding this, the mixed Graph of terms is not a tree structure, and more importantly is not rigid but is dynamically built depending on the optimization stage. Moreover, the Hierarchical Mixture of Experts does not consider relations between experts which is, on the other hand, largely employed in the mixed Graph of Terms. Nevertheless, we will explore further connections between the two methods in future works.

5 Experiments

We have compared 2 different query expansion methodologies based on different *vector of features* with the baseline ($\mathbf{b}$): the mixed *Graph of Terms* ($\mathbf{g}$) and the *List of Terms* ($\mathbf{w}$). The baseline ($\mathbf{b}$) is the the *tf-idf* model without expansion of the query. We have embedded all the techniques in an open source text-based search engine, Lucene from the Apache project. Here the score function $\mathbf{S}(\mathbf{q}, \mathbf{d})$ is based on the standard vector cosine similarity[3] , used in a Vector Space Model combined with the Boolean Model [9]

[3] We have used the Lucene version 2.4 and you can find details on the similarity measure at `http://lucene.apache.org`

Table 1. An example of a $\mathbf{g}$ for the topic *Storage Tank*

Conceptual Level		
Concept i	**Concept** j	**boost factor** (b)
tank	roof	1.0
tank	water	0.37
tank	liquid	0.14
...	...	...
liquid	type	0.44
liquid	pressur	0.21
...	...	...
Word Level		
Concept i	**Word** s	**boost factor** (b)
tank	larg	0.15
tank	construct	0.14
...	...	...
liquid	type	0.21
liquid	maker	0.12
liquid	hose	0.06
liquid	fix	0.01
...	...	...

which takes into account the boost factor b_k whose default value is 1, which is assigned to the words that compose the original query. Such a function permits the assignment of a rank to documents $\mathbf{w}$ that match a query $\mathbf{q}$ and permits the transforming of each *vector of features*, that is the $\mathbf{g}$ into a set of Boolean clauses. For instance, in the case of the $\mathbf{g}$, since it is represented as pairs of related words, see Table 1, where the relationship strength is described by a real value (namely ψ_{ij} and ρ_{is}, the *Relation factors*), the expanded query is:

$$((tank \text{ AND } roof)^{1.0}) \text{ OR } ((tank \text{ AND } larg)^{0.15})...$$

As a consequence we search the pair of words *tank* AND *roof* with a boost factor of 1.0 OR the pair of words *tank* AND *larg* with a boost factor of 0.15 and so on.

5.1 Data Preparation

The evaluation of the method has been conducted on a web repository collected at the University of Salerno by crawling 154,243 web pages for a total of about 3.0 GB by using the website ThomasNet (http://www.thomasnet.com) as an index of URLs, the reference language being English[4]. ThomasNet, known as the "big green books" and "Thomas Registry", is a multi-volume directory of industrial product information covering 650,000 distributors, manufacturers and service companies within 67,000-plus industrial categories. We have downloaded webpages from the company websites related to 150 categories of products (considered as topics), randomly chosen from the ThomasNet directory.

[4] The repository will be public on our website to allow further investigations from other researchers.

Table 2. Number of words and pairs for each **g**

	♯ of words	♯ of pairs
Average Size	**55**	**72**

Table 3. Average values of performance

run	eMAP	eRprec	eP5	eP10	eP20	eP30	eP100
b	0.213	0.432	0.345	0.298	0.201	0.198	0.186
w	0.399	0.457	0.806	0.691	0.661	0.556	0.384
g	0.569	0.601	0.917	0.840	0.784	0.686	0.495

Note that even if the presence or absence of categories in the repository depends on the random choices made during the crawling stage, it could happen that webpages from some business companies cover categories that are different from those randomly chosen. This means that the repository is not to be considered as representative of a low number of categories (that is 150) but as a reasonable collection of hundreds of categories. In this work we have considered 50 test questions (queries) extracted from 50 out of the initial 150 categories (topics). Each original query corresponds to the name of the topic, for instance if we search for information about the topic ”generator” therefore the query will be exactly ”generator”. Obviously, all the initial queries have been expanded through the methodologies explored in section 4.3. Here we show the summary results obtained on all the 50 topics.

5.2 Evaluation Measures

For each example the procedure that obtains the reformulation of the query, is explained as follows. A person, who is interested in the topic ”generator”, performs the initial query ”generator” so interactively choosing 3 relevant documents for that topic, which represent the minimal positive feedback. From those documents the system automatically extracts the two *vectors of features*. In table 2 we show the average size of the list of words and the list of pairs, that is 55 and 72 respectively. The user has interactively assigned the relevance of the documents by following an *xml* based schema coding his intentions and represented as suggested by *TREC* [5].

The expanded queries have been performed again and for each context we have asked different humans to assign graded judgments of relevance to the first 100 pages returned by the system. Due to the fact that the number of evaluations for each topic, and so the number of topics itself, is small, the humans have judged, in contrast to the Minimum Test Collection method [8], all the results obtained. The assessment is based on three levels of relevance, *high relevant*, *relevant* and *not relevant*, assigned, to avoid cases of ambiguity, by following the *xml* based schema coding the user intentions.

The accuracy has been measured through standard indicators provided by [9] and based on *Precision* and *Recall*,

[5] The Text Retrieval Conference (TREC).

$$eAP = \frac{1}{ER}\sum_{i=1}^{k}\frac{x_i}{i} + \sum_{j>i}\frac{x_i x_j}{j} \tag{10}$$

$$ePrec@k = eP@k = \frac{1}{k}\sum_{i=1}^{k} x_i \tag{11}$$

$$ERprec = \frac{1}{ER}\sum_{i=1}^{ER} x_i \tag{12}$$

$$ER = \sum_{i=1}^{n} x_i \tag{13}$$

where eAP indicates the average precision on a topic, x_i and x_j are Boolean indicators of relevance, k is the cardinality of the considered result set (k=100) and ER is a subset of relevant documents[6]. The factor $ERprec$ is the precision at the level ER, while the measure $eMAP$ is the average of all eAPs over topics. The measure $eP@k$ is the precision at level k (for instance $eP5$ is the precision calculated by taking the top 5 results).

In table 3 we find summary results across topics for each *vector of features* and for the baseline ($\mathbf{b}$).The overall behavior of the $\mathbf{g}$ method is better than the $\mathbf{w}$.

6 Conclusions

In this work we have demonstrated that a mixed *Graph of Terms* based on a hierarchical representation is capable of retrieving a greater number of relevant documents than a less complex representation based on a list of words, even if the size of the training set is small and composed of only relevant documents.

These results suggest that our approach can be employed in all those text mining tasks that consider matching between patterns represented as textual information and in text categorization tasks as well as in sentiment analysis and detection tasks.

The proposed approach computes the expanded queries considering only endogenous knowledge. It is well known that the use of external knowledge, for instance WordNet, could clearly improve the accuracy of information retrieval systems, but we consider this as a future work.

References

1. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. ACM Press, New York (1999)
2. Bhogal, J., Macfarlane, A., Smith, P.: A review of ontology based query expansion. Information Processing & Management 43(4), 866–886 (2007)
3. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer (2006)

[6] Note that, $ER = |R_{mGT} \cup R_{GT} \cup R_{LT} - R_{mGT} \cap R_{GT} \cap R_{LT}|$, where R_{vf} is the set of relevant and high relevant documents obtained for a given topic and *vf*=*vector of features*.

4. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
5. Callan, J., Croft, W.B., Harding, S.M.: The inquery retrieval system. In: Proceedings of the Third International Conference on Database and Expert Systems Applications, pp. 78–83. Springer (1992)
6. Cao, G., Nie, J.Y., Gao, J., Robertson, S.: Selecting good expansion terms for pseudo-relevance feedback. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, pp. 243–250. ACM, New York (2008)
7. Carpineto, C., de Mori, R., Romano, G., Bigi, B.: An information-theoretic approach to automatic query expansion. ACM Trans. Inf. Syst. 19, 1–27 (2001), http://doi.acm.org/10.1145/366836.366860
8. Carterette, B., Allan, J., Sitaraman, R.: Minimal test collections for retrieval evaluation. In: 29th International ACM SIGIR Conference on Research and Development in Information retrieval (2008)
9. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University (2008)
10. Collins-Thompson, K., Callan, J.: Query expansion using random walk models. In: Proceedings of the 14th ACM International Conference on Information and Knowledge Management, CIKM 2005, pp. 704–711. ACM, New York (2005), http://doi.acm.org/10.1145/1099554.1099727
11. Dumais, S., Joachims, T., Bharat, K., Weigend, A.: SIGIR 2003 workshop report: implicit measures of user interests and preferences. SIGIR Forum 37(2), 50–54 (2003)
12. Efthimiadis, E.N.: Query expansion. In: Williams, M.E. (ed.) Annual Review of Information Systems and Technology, pp. 121–187 (1996)
13. Griffiths, T.L., Steyvers, M., Tenenbaum, J.B.: Topics in semantic representation. Psychological Review 114(2), 211–244 (2007)
14. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer (2009)
15. Jansen, B.J., Booth, D.L., Spink, A.: Determining the informational, navigational, and transactional intent of web queries. Information Processing & Management 44(3), 1251–1266 (2008)
16. Jansen, B.J., Spink, A., Saracevic, T.: Real life, real users, and real needs: a study and analysis of user queries on the web. Information Processing & Management 36(2), 207–227 (2000)
17. Ko, Y., Seo, J.: Text classification from unlabeled documents with bootstrapping and feature projection techniques. Inf. Process. Manage. 45, 70–83 (2009)
18. Lang, H., Metzler, D., Wang, B., Li, J.T.: Improved latent concept expansion using hierarchical markov random fields. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, CIKM 2010, pp. 249–258. ACM, New York (2010), http://doi.acm.org/10.1145/1871437.1871473
19. Lee, C.-J., Lin, Y.-C., Chen, R.-C., Cheng, P.-J.: Selecting Effective Terms for Query Formulation. In: Lee, G.G., Song, D., Lin, C.-Y., Aizawa, A., Kuriyama, K., Yoshioka, M., Sakai, T. (eds.) AIRS 2009. LNCS, vol. 5839, pp. 168–180. Springer, Heidelberg (2009)
20. Noam, S., Naftali, T.: The power of word clusters for text classification. In: 23rd European Colloquium on Information Retrieval Research (2001)
21. Okabe, M., Yamada, S.: Semisupervised query expansion with minimal feedback. IEEE Transactions on Knowledge and Data Engineering 19, 1585–1589 (2007)
22. Piao, S., Rea, B., McNaught, J., Ananiadou, S.: Improving Full Text Search with Text Mining Tools. In: Horacek, H., Métais, E., Muñoz, R., Wolska, M. (eds.) NLDB 2009. LNCS, vol. 5723, pp. 301–302. Springer, Heidelberg (2010)

23. Robertson, S.E., Walker, S.: On relevance weights with little relevance information. In: Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1997, pp. 16–24. ACM, New York (1997)
24. Robertson, S.E.: On term selection for query expansion. J. Doc. 46, 359–364 (1991)
25. Ruthven, I.: Re-examining the potential effectiveness of interactive query expansion. In: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval, SIGIR 2003, pp. 213–220. ACM, New York (2003), http://doi.acm.org/10.1145/860435.860475
26. Salton, G., McGill, M.J.: Introduction to modern information retrieval. McGraw-Hill (1983)