Available online at www.sciencedirect.com
SCIENCE DIRECT®
Information Sciences 176 (2006) 415–437
INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL
www.elsevier.com/locate/ins

Computing with words for text processing: An approach to the text categorization

Sławomir Zadrożny [a,*], Janusz Kacprzyk [b]

[a] *Warsaw School of Information Technology, ul. Newelska 6, 01-447 Warsaw, Poland*
[b] *Systems Research Institute, Polish Academy of Sciences, ul. Newelska 6, 01-447 Warsaw, Poland*

Abstract

The use of the *computing with words* paradigm for the automatic *text documents categorization* problem is discussed. This specific problem of *information retrieval* (IR) becomes more and more important, notably in view of a fast proliferation of textual information available on the Internet. The main issues that have to be addressed here are: document representation and *classification*. The use of fuzzy logic for both problems has already been quite deeply studied though for the latter, i.e., classification, generally not in an IR context. Our approach is based mainly on the classical *calculus of linguistically quantified propositions* proposed by Zadeh. Moreover, we employ results related to fuzzy (linguistic) queries in IR, notably various interpretations of the weights of query terms. Some preliminary results on widely adopted text corpora are presented.

Keywords: Computing with words; Information retrieval; Linguistic quantifiers; Text categorization

* Corresponding author.
E-mail address: slawomir.zadrozny@ibspan.waw.pl (S. Zadrożny).

0020-0255/$ - see front matter
doi:10.1016/j.ins.2005.07.017

1. Introduction

The computing with words paradigm tries to grasp the human ability to effectively process the vague and imprecise information. Basically, this leads to the representation of some modeled variables with *linguistic expressions* rather than with strict numerical values. Moreover, a further processing, notably aggregation, fusion, etc., of such information is proposed to be done using flexible operators, exemplified by *linguistic quantifiers*. Information retrieval tasks are perfectly amenable to such a treatment. The information retrieval process is characterized [24] by such features as uncertainty (most often of the probabilistic nature), partial matching, incompleteness of queries, a vague concept of *relevance*.

Much research has been done in the area of application of the elements of fuzzy logic for the purposes of the information retrieval, cf., e.g., [3–5,17–19,10,11,13]. The main issues of text documents representation and their querying has been addressed within this framework. The use of fuzzy logic related concepts for query structure and interpretation is especially promising. This is due to the fact that some elements of the classical IR system interface may be artificially precise and too rigid for a human user. In addition to the main task of an IR system, i.e., the retrieval of documents relevant for the user needs, there are many other related tasks. Among them, quite an important task is that of text document categorization. Basically, it consists in assigning some thematic categories to the documents. This may be, and often is, done manually. However, in case of huge documents sets, exemplified by those available through the Internet, this becomes ineffective and inefficient. Thus, automatic approaches are more and more often applied. They are usually based on results obtained in machine learning.

The text categorization task exhibits some imprecision. Even a human being may be unsure as to a clear-cut classification of a document to just one category. Moreover, it is quite natural to consider a degree of belongingness to a category. This becomes even more apparent in case of an automatic classification procedure. We may easily expect that the results of classification may be ambiguous. The fuzzy logic approach has proved to be useful in such a context. We have implemented [37] a pilot version of an Internet oriented IR system featuring some elements of fuzzy logic built into the user interface and making it more human consistent. Here, we investigate possibilities of the application of some fuzzy logic related concepts to the very classification process. Our approach is mainly based on the use of linguistically quantified propositions in the sense of Zadeh to model some intuitively appealing rules of classification. Moreover, we adopt results on fuzzy extensions to the querying language of an IR system proposed by other authors and referred to earlier ones. From a certain perspective, the task of classification may be treated as a specific querying task in the other than the original space of text documents of an IR system.

In Section 2, we briefly overview the main concepts of information retrieval. The next section presents Zadeh's calculus of linguistically quantified propositions and the concept of a linguistic variable. Section 4 discusses some known extensions to the Boolean model of IR. Section 5 discusses the task of text documents categorization and possible approaches to it using the methodology presented in the previous sections. In this section we also describe computational experiments and their preliminary results.

2. Brief overview of information retrieval

The most important issues for virtually any research in IR are those of document representation and query language. Usually, it is assumed that there exist three main approaches (models): Boolean, vector space and probabilistic. Fuzzy logic based concepts have been so far primarily discussed within the framework of the two first models. In this paper we follow this way. For a brief description of these models we assume the following notation:

$D = \{d_i\}_{i=1,N}$ – a set of text documents,

$T = \{t_j\}_{j=1,M}$ – a set of index terms.

The Boolean model represents a document as a set of terms assigned to it

$$d_i = \{t_k\}_{k=1,K}, \quad d_i \in D, \ t_k \in K, \tag{1}$$

and the representation of documents may be formally expressed via the following function:

$$F : D \times T \to \{0, 1\}. \tag{2}$$

In the vector space model a document is represented as a vector:

$$d_i = [w_1, \ldots, w_M], \quad d_i \in A^M, \ A \subseteq R, \tag{3}$$

where each dimension corresponds to an index term and the value of w_j (*weight*) determines to what extent a term $t_j \in T$ is essential for the description of the content of the document. Most often, $A = [0, 1]$ is assumed, and thus function F takes the form:

$$F : D \times T \to [0, 1]. \tag{4}$$

The index terms may be some general concepts describing the content of documents. In the librarian IR systems these are usually carefully selected terms indicating, e.g., in case of scientific library, the discipline of a book/journal or keywords for a journal article. In such a case, usually an expensive/time-consuming work of an expert is required to assign index terms. An alternative approach, popular in case when an automatic *indexing* is needed, is to select some words actually appearing in a document as its indexing terms. In the

literature there are proposed many forms of function F (4) for this alternative. One of the most popular is based on the requirement that such a function should assign to a term (word) t_j in a document d_i a weight directly proportional to its frequency in this document and inversely proportional to the number of documents in which it appears. It may be formalized as a so-called $tf \times idf$ function:

$$F(d_i, t_j) = \left(f_{ij} / \arg\max_j f_{ij} \right) * \left(\log(N/n_j) / \arg\max_j \log(N/n_j) \right), \tag{5}$$

where f_{ij} is the frequency of term t_j in document d_i, N is the number of all documents in set (collection) D and n_j is a number of documents from D where term t_j appears (*document frequency*). Thus, the first factor is the normalized frequency of term (*tf*, *term frequency*) t_j in document d_i, while the second factor is the normalized inverted frequency (*idf*, *inverted document frequency*) in the collection D of documents in which term t_j appears at least once. Other normalization schemes may be employed, too.

Besides document representation, each classical model also offers a querying formalism. In the Boolean model the query is a formula in the sense of propositional calculus of mathematical logic. Each term t_j is identified with an atomic formula (proposition), z_j. These may be combined using logical connectives, notably the conjunction and disjunction. Then, such a formula/query is evaluated for each document as it is done in model theory of propositional calculus. More precisely, each proposition z_j of a query is assigned a truth value *true* if $t_j \in d_i$ and a truth value *false* in the opposite case Then, a document is treated as *relevant*, i.e., matching the query, if the whole formula corresponding to the query is true after such an assignment of truth value to its atomic formulae (propositions). Thus, in the classical Boolean model the relevance (matching) is a binary concept: a document is relevant or not, no intermediate situation is possible.

In the vector space model the query takes the form of a vector as in (3), securing a unified representation of documents and queries. The relevance of a document has here a gradual character. The matching degree of a query and a document is computed as a similarity $\mathrm{sim}(d, q)$ of vectors that represent them. Most popular similarity measure used is the cosine of both vectors:

$$\mathrm{sim}(d, q) = \frac{\sum_{i=1}^{l} w_i q_i}{\sqrt{\sum_{i=1}^{l} w_i^2} \sqrt{\sum_{i=1}^{l} q_i^2}}, \tag{6}$$

where $d = [w_1, \ldots, w_l]$ and $q = [q_1, \ldots, q_l]$. Thus, the matching degree is a number from $[0, 1]$.

In both models we can distinguish *elementary queries*, i.e., such queries that refer to just one term. Formally, they are expressed as atomic formulae (single propositions) and one-dimensional vectors in the Boolean and vector

space models, respectively. This concept will be useful for our further considerations.

Here we describe very briefly the basics of the primary task studied in IR, i.e., the organization of text documents and their retrieval by matching queries against documents. There are many other related tasks considered within IR. One of them, *text categorization*, being a primary topic of this paper, is discussed more thoroughly in Section 5.

3. Linguistic approaches to information aggregation and representation

Fuzzy logic makes it possible to represent and quantify imprecise information. This provides for a more flexible information representation and processing. The latter comprises, among others, an aggregation of pieces of information. Basic classical aggregation operators related to logical connectives (AND, OR) and quantifiers (for all, there exists) are often too strict—for more on that, see Section 4. In many practical situations a human being would express a constraint by stating "*Most* of the conditions ... should be fulfilled". The word "most" may be replaced here with some other *linguistic quantifier*: "almost all", "much more than 50%", etc. As it often happens that all/some conditions quantified are of a gradual type, both the conditions and quantifier are best modeled within the framework of fuzzy logic.

Zadeh [36] introduced two types of *linguistically quantified propositions*:

$$QX\text{'s are } G\text{'s} \quad \text{(type I)}, \tag{7}$$

$$QB\text{'s are } G\text{'s} \quad \text{(type II)}, \tag{8}$$

where Q is a linguistic quantifier, and G and B are fuzzy sets in the universe X. Fuzzy linguistic quantifiers are represented by fuzzy sets defined in an appropriate universe. The *absolute* linguistic quantifiers such as "approximately 3", "several", etc. are represented as fuzzy subsets on domain of positive real numbers, R^+; *proportional* linguistic quantifiers such as "most", "almost all", etc. are represented by fuzzy subsets, Q, on the interval $[0,1]$:

$$\mu_Q : [0,1] \to [0,1]. \tag{9}$$

Zadeh proposed an interpretation for the proportional linguistic quantifiers such that the truth degree T of proposition (7) is computed using the following formula:

$$T = \mu_Q\left(\frac{\mathrm{card}(G)}{\mathrm{card}(X)}\right) = \mu_Q\left(\frac{\sum_i \mu_G(x_i)}{n}\right), \tag{10}$$

where μ_Q is the membership function of quantifier Q and n is the cardinality of the universe X. For propositions of type (8) we have:

$$T = \mu_Q\left(\frac{\mathrm{card}(G \cap B)}{\mathrm{card}(B)}\right) = \mu_Q\left(\frac{\sum_i(\mu_G(x_i) \wedge \mu_B(x_i))}{\sum_i \mu_B(x_i)}\right). \tag{11}$$

Thus, the truth of a proposition of type (7) is proportional to the fraction of elements of the universe X that belong to its subset G. An exact form of this relationship is determined by the membership function of Q which may be, for "most" of the following, piece-wise linear, form:

$$\mu_Q(y) = \begin{cases} 1 & \text{for } y \geqslant 0.8, \\ 2y - 0.6 & \text{for } 0.3 < y < 0.8, \\ 0 & \text{for } y \leqslant 0.3. \end{cases}$$

On the other hand, the truth of a proposition of type (8) is proportional to the fraction of elements of a (fuzzy) set $B \subseteq X$ that at the same time belong to $G \subseteq X$. Thus, B plays here a role similar to the scope in case of the classical quantifiers. However, due to the nature of a linguistic quantifier, the type II proposition is not equivalent to the type I propositions with B connected with G using the implication or conjunction as it is true for the classic general and existential quantifier, respectively.

This interpretation of fuzzy linguistic quantifiers is very attractive due to its simplicity both in defining and using a quantifier. The type II quantifiers offer a capability of a weighted aggregation. However, it may become inconvenient in some applications, mainly due to the use of a simple cardinality of fuzzy sets, so-called $\sum$Counts. This makes a number of elements with a low membership degree to count as one element with a high degree.

A convenient approach to handle fuzzy linguistic quantifiers is to use the ordered weighted averaging (OWA) operators, introduced by Yager [30]; see also Yager and Kacprzyk's book [31].

Linguistic quantifiers provide for a flexible processing of fuzzy information. However, information to be aggregated has still to be provided in a strict, numerical form. It is argued that for some applications it may be counter-intuitive, and not human consistent. Thus, often, also in IR related fuzzy logic applications, Zadeh's concept [35] of a linguistic variable is employed. It may be briefly described as follows. A linguistic variable is a variable taking on the values in the form of linguistic terms (labels). Formally, a linguistic variable is a tuple $(H, T(H), U, G, M)$, where H is a name of the variable, $T(H)$ is a set of its values (linguistic terms); $U = \{u\}$ denotes the universe under consideration [i.e., fuzzy sets defined over U provide the interpretation for particular terms belonging to $T(H)$], G is a rule that generates values for the linguistic variable [if $T(H)$ is finite, then G may be just a simple enumeration of linguistic terms]; M is a semantic rule providing for each value $l \in T(H)$ its meaning $M(l) \subsetneq U$. For example, treating age as a linguistic variable, one may assume: $T(\text{"age"}) = \{\text{"very young"}, \text{"young"}, \text{"middle aged"}, \text{"old"}, \text{"very old"}\}$, $U = [1, 100]$, M associates with particular values of $T(\text{"age"})$ fuzzy numbers

defined over the interval $[0,100]$ and intuitively corresponding to individual descriptions of the age. For example, with the term "young" a trapezoidal fuzzy number $(0;0;25;35)$ may be associated. Thus, basically, the meaning of linguistic terms is provided by fuzzy numbers (usually triangular or trapezoidal) associated via M and then all operations on linguistic terms, are done on these fuzzy numbers, see, e.g., [6]. This may lead to some problems when the results of such an aggregation are to be again expressed in terms of the original linguistic terms (e.g., in case of the averaging of values of a linguistic variable). Then, *linguistic approximation* has to be applied and the results may not be fully reliable.

In another approach, the set of linguistic terms is assumed to be finite and ordered. Thus, the semantics of a term is provided just by its position in the order imposed—no fuzzy numbers are associated. In such a case all operations on the linguistic terms have to be specifically defined. For details, see [7,10,11]. Such an *ordinal linguistic approach* has been proposed for IR systems [10,11].

4. Fuzzy extensions of the Boolean model

The vector space model has been widely accepted as an effective and efficient way of dealing with the tasks addressed within IR. On the other hand, the query language of this model is rather limited. Practically, a query matches such documents that are represented with the terms weighted similarly as in the query. This corresponds, more or less, to the logical conjunction of the elementary queries. That is in contrast with the query language of the Boolean model in which the user may freely combine elementary queries using logical connectives. The classical Boolean model suffers from an oversimplified representation of documents as sets of terms. It has been observed that the combination of flexibility of the Boolean querying language and the vector representation of documents may be worthwhile. This has led to many proposals for extensions of the Boolean model. The extended Boolean model has also become a starting point for many proposals for the use of fuzzy logic concepts in IR. A vector representing a document via function F such as in (5) may be easily interpreted as a *fuzzy set of terms*.

Extensions to the Boolean model may modify only the documents representation or both documents representations and queries. In the former case, the documents are represented like in the vector space model and the query language remains unchanged. The evaluation of classic Boolean queries against documents represented with weighted terms may take various forms.

First of all, queries may be interpreted as formulae of *fuzzy propositional calculus*. Thus, they may be true to a degree from the interval $[0,1]$. As in the classical case, matching of a query against a document is computed as the truth degree of the query/formula under the evaluation of the propositions/terms

provided by the document, or more precisely, by weights of the query terms in the document. Classical fuzzy operators/connectives of min and max are used in place of AND and OR, respectively. Obviously, various variants of this approach may be obtained by using some *triangular norms* and *conorms* to represent logical connectives.

It is widely acknowledged that classical aggregation operators corresponding to the AND and OR connectives often fail to represent actual requirements of the user (not only in the context of querying, but in broadly meant decision making). The present authors were among the first to propose a solution for that problem in the context of querying of databases. We discuss it briefly later in the paper and now let us look at a more precise statement of that problem and a solution proposed to it even earlier by Salton and coworkers [26,1]. Let us assume that a query is the conjunction of elementary queries:

$$q = t_1 \wedge \cdots \wedge t_l \tag{12}$$

(we equate here proposition z_i with term t_i, for readability). Under a classical interpretation of the AND connective, even if just one term of $t_1, \ldots, t_l$ is absent in a document it makes the document completely irrelevant (non-matching) to the query. Even under a fuzzy interpretation the document is relevant only to a degree determined by a term $t_1, \ldots, t_l$ to which the lowest weight is assigned in the document, thus possibly also to a degree 0. It seems to be fully rational to expect that such a matching degree should vary depending on the number of terms of the query that are well/poorly matched in the document. Salton et al. observed that the relevance of a document should be inversely proportional to the distance between two l-dimensional vectors $[w_1, \ldots, w_l]$ and $[1, \ldots, 1]$, where the former gathers weights in the document of terms used in the query, $t_1, \ldots, t_l$. Analogously, for a query being a disjunction of elementary queries $q = z_1 \vee \cdots \vee z_l$, its matching degree should be proportional to the distance of vectors $[w_1, \ldots, w_l]$ and $[0, \ldots, 0]$. This idea has been adopted in an extension to the Boolean model, called a *p-norm model*. The distance between the vectors is computed using a p-norm (more often referred to as an l-norm) for a selected value of parameter p. For $p = 1$ we obtain the classical vector space model, and for $p = \infty$ we obtain a simple fuzzy model described above which employs the min and max operators.

The very same problem of some deficiencies of the classical AND and OR, as well as their fuzzy counterparts the min and max were addressed by Kacprzyk and coworkers [14–16] in the context of fuzzy querying of a classical relational database (thus the setting assumed there is somehow dual to the one considered here: queries are fuzzy while the content of a database is crisp; however the idea of a linguistic quantifier guided aggregation applies in both cases). They proposed to aggregate elementary queries using *linguistic quantifiers*, such as, e.g., "most". Thus, instead of insisting on the fulfillment of all elementary queries as it is required by the AND connective (and the general quantifier, $\forall$,

corresponding to it) or just one elementary query as it is allowed by the OR connective (and the existential quantifier, $\exists$, corresponding to it) the user may require to have "most", "almost all", "many", etc. of elementary queries satisfied. As the above observation of Salton et al. illustrates, the use of such a flexible quantifier may be convenient even in case when we have in mind a strict conjunction of the elementary queries. If there are no documents fully meeting our requirements, we get an *empty answer* with the classical aggregation operator AND, while using a linguistic quantifier, we get a list of documents almost meeting the query.

The ordered weighted min (OWmin), cf. [8], operator provides another scheme for the evaluation of a query (12). The motivation here is exactly the same as in case of linguistic quantifier guided aggregation, i.e., instead of requiring that all elementary queries in (12) are matched, we are satisfied with *most* of them being matched. This is formalized in a way slightly different to that of linguistic quantifiers. Namely, the concept of a requested majority of matched elementary queries, e.g., *most*, is modeled as a fuzzy set I in the space $\{0,1,2,\ldots,l\}$, such that $\mu_I(0)=1$; $\mu_I(i) \geqslant \mu_I(i+1)$, cf. (9). Thus (cf. [8]), if we require that "at least k elementary queries are matched", then we set $\mu_I(i)=1$ for all $0 \leqslant i \leqslant k$ and $\mu_I(i)=0$ for all $i>k$. Moreover, let us assume that $t_i(d_j)$ denotes the matching of document d_j to the elementary query t_i. Then, we sort vectors $\langle t_1(d_j), t_2(d_j), \ldots, t_l(d_j)\rangle$ in a non-increasing order to obtain $t_{1^*}(d_j) \geqslant t_{2^*}(d_j) \geqslant \cdots \geqslant t_{l^*}(d_j)$ where $t_{1^*}(d_j)$ is the greatest value from among $t_1(d_j), t_2(d_j), \ldots, t_l(d_j)$; $t_{2^*}(d_j)$ is the second greatest value, etc. Then, to obtain a matching degree of document d_j to the overall query (12) we compute:

$$\min_{i=1,l} \max(1-\mu_I(i), t_{i*}(d_j)). \tag{13}$$

In this approach the concept of a linguistic quantifier in the sense of Zadeh may be directly employed to provide a definition of a fuzzy set I. This may be interpreted in a more general setting as the Sugeno integral; for details cf. [8].

All the above min (or more generally t-norm), p-norm and linguistic quantifier guided models produce an aggregated evaluation (matching degree) of the conjunction of elementary queries (12). Sometimes this is more then needed and we may be quite satisfied with just an ordering of the documents from the best matching the query to the least matching one. For such a purpose many more methods are available. For example, LEXIMIN [8], compares two documents in terms of their matching to the query (12) in the following way. Let us assume the same notation as in case of OWmin discussed above, however this time the vector $\langle t_1(d_j), t_2(d_j), \ldots, t_l(d_j)\rangle$ is sorted in the non-decreasing order. Then, d_1 is said to better match the overall query (12) than d_2 if there exist such a k that $t_{i^*}(d_1) = t_{i^*}(d_2)$ for all $i<k$ and $t_{k^*}(d_1) > t_{k^*}(d_2)$. Thus [8], LEXIMIN favors documents failing to match as few elementary queries as

possible. LEXIMAX [8], on the other hand, favors documents matching as many elementary queries as possible. This, assuming the same notation as for OWmin, boils down to declaring d_1 as better matching the overall query (12) than d_2 if there exist such a k that $t_{i^*}(d_1) = t_{i^*}(d_2)$ for all $i < k$ and $t_{k^*}(d_1) > t_{k^*}(d_2)$.

Other fuzzy extensions to the Boolean model assume weights assigned to terms of the query. In the classical vector space model the interpretation of such weights in queries is quite simple: the documents are sought having terms weighted similarly as in the query. In the extended Boolean model more interpretations are possible. In order to describe them briefly, let us assume a weighted query in the disjunctive normal form:

$$q = ((t_{11}, w_{11}) \wedge \cdots \wedge (t_{1u}, w_{1u})) \vee \cdots \vee ((t_{d1}, w_{d1}) \wedge \cdots \wedge (t_{\mathrm{dw}}, w_{\mathrm{dw}})), \tag{14}$$

where t_{ij} denotes a term and w_{ij} its weight in the query. Assuming such a canonical form we then focus on the matching degree of a single conjunction:

$$(t_{11}, w_{11}) \wedge \cdots \wedge (t_{1u}, w_{1u}) \tag{15}$$

referred to as a *disjunct*. The matching degree of the whole query is obtained via an aggregation, e.g., using the max operator, of matching degrees of all disjuncts. In the literature three interpretations of the query weights w_{ij} are considered [3]:

1. relative importance, (16)
2. thresholds of importance, and (17)
3. ideal weights. (18)

According to the first interpretation, weight w_{ij} of term t_{ij} in a query indicates to what extent the appearance of term t_{ij} in a document is important for the document to satisfy the query. If the weight is low (close to 0), then the absence of term t_{ij} in a document (i.e., low, possibly equal 0, weight of this term in the document) does not exclude the matching of this document against the query. If the weight of a term in a query is high (close to 1), then the document has to contain the term (i.e., have a high weight assigned to this term) to qualify for matching the query.

Due to the second interpretation, the weights of particular terms in the documents sought have to be higher than threshold values w_{ij} given in the query. There are further possible interpretations depending on how the under-satisfaction of query terms is treated—a further discussion is given below. Herrera-Viedma [11] proposed a modified interpretation of query weights in this interpretation. Namely, high query weights are treated as above, but low weights require that the corresponding weights in the documents have to be lower.

The third interpretation is somehow analogous to that assumed in the vector space model: the documents sought should be characterized by weights of the terms similar to those specified in the query.

The existence of these various interpretations of query term weights poses certain theoretical difficulty. Fortunately, their analysis may be to some extent unified due to the results obtained in the area of using fuzzy logic in multicriteria decision making, fuzzy querying of databases and fuzzy information retrieval, cf. [9]. These may be summarized in terms of IR as follows. Let us denote the matching degree of document d and query q of the form (15) by $\gamma(q,d)$. Moreover, let us assume the matching degree of an elementary query $q_i = t_i$ (without a weight) and a document d, equal to the weight of the term t_i in the document d (4), i.e.,

$$\gamma(q_i = t_i, d) = F(d, t_i).$$

The matching degree of the whole query (15) is computed as

$$\gamma(q,d) = \min_i(\gamma(q_i,d)). \tag{19}$$

Dubois and coworkers [9,8] analyzed several interpretations, cf. [2,29], of importance weights assigned to elementary queries. They observed that a number of them may be treated as a special case of the following general scheme:

$$\gamma(q_i,d) = w_i \to F(d,t_i), \tag{20}$$

where $\to$ is a fuzzy implication operator. Then, using different implication operators we recover various interpretations of importance weights of the terms.

For the Kleene–Dienes implication:

$$x \to y = \max(1-x, y) \tag{21}$$

we get

$$\gamma(q_i,d) = \max(F(d,t_i), 1-w_i), \tag{22}$$

that is Yager's interpretation [29], which may be expressed as follows. If an elementary query is completely unimportant ($w_i = 0$), then it does not pose any constraints on the form of the document that has to meet it (the matching degree is always equal 1). Otherwise, a document to satisfy the query has to contain term t_i with a high weight w_i assigned. Thus, this interpretation corresponds to the concept of a relative importance (16).

For the Gödel implication:

$$x \to y = \begin{cases} 1 & \text{if } x \leqslant y, \\ y & \text{otherwise} \end{cases} \tag{23}$$

we get

$$\gamma(q_i,d)=\begin{cases}1 & \text{if } F(d,t_i)\geqslant w_i,\\ F(d,t_i) & \text{otherwise,}\end{cases} \tag{24}$$

that is another Yager's interpretation (cf. [9]), and requires that term t_i has in a document the weight higher than indicated in the query. Thus, this directly refers to the concept of an importance threshold (17). If the weight of the term in the document, $F(d,t_i)$, does not reach the query weight of this term w_i, such a document satisfies this elementary query to a degree equal to $F(d,t_i)$. Another, continuous treatment of such an undersatisfaction is obtained while using the Goguen implication:

$$x\rightarrow y=\begin{cases}1 & \text{if } x\leqslant y,\\ y/x & \text{otherwise}\end{cases} \tag{25}$$

and we obtain:

$$\gamma(q_i,d)=\begin{cases}1 & \text{if } F(d,t_i)\geqslant w_i,\\ F(d,t_i)/w_i & \text{otherwise.}\end{cases} \tag{26}$$

Yet another characterizations of importance thresholds has been provided in the literature, cf. [23].

A similar analysis may be provided for the disjunction of elementary queries, cf. for details [4,5].

Here we add a following observation to the characterization of ideal weights (18) provided in [4,5]. Namely, the following characterization of logic of the ideal weights interpretation may be proposed:

$$\gamma(q_i,d)=w_i\leftrightarrow F(d,t_i), \tag{27}$$

where $\leftrightarrow$ is a fuzzy equivalence operator. Then, using a definition of a fuzzy equivalence based on the Goguen implication:

$$x\leftrightarrow y=\min(x\rightarrow y,y\rightarrow x)=\begin{cases}1 & \text{if } F(d,t_i)=w_i,\\ w_i/F(d,t_i) & \text{if } F(d,t_i)>w_i,\\ F(d,t_i)/w_i & \text{if } F(d,t_i)<w_i,\end{cases} \tag{28}$$

we obtain a reasonable characterization of the ideal weights interpretation.

5. Fuzzy concepts in document categorization

The primary task considered in IR is that of retrieving documents relevant to the user. This task decomposes into the representation of documents and queries, and their matching. In Sections 2 and 4 we briefly reviewed fuzzy logic based approaches proposed for this purpose. In what follows we briefly discuss

a related problem of automated text categorization. Next, we describe our idea of using fuzzy logic based models from the previous section for this purpose. Then, in the next section we present our computational experiments and their results.

5.1. The concept of automatic text categorization

In order to make clearer the concept of automated text categorization, let us consider scenarios in which it may be applicable. The first one is that of a Web Spider, an agent software "traversing" the Web and automatically classifying documents found aimed at providing the user only with the documents of interest to him or her (i.e., belonging to a pre-specified category/categories). The second scenario is that of a "translation agency". In this case, the aim of the system is to automatically assign to interpreters documents sent by customers. Interpreters prefer certain categories of documents and the aim is to match their preferences so as to secure a high efficiency of the whole translation process. In both cases the classification may be done manually. However, it may be not such a good solution as it may seem. Firstly, in particular in the former case, it is unreasonable to expect that all documents are classified by their authors or some other bodies (see, e.g., Yahoo!). Secondly, the classification provided by the author may be useless for, or inconsistent with, the purposes of the document "consumer". Both scenarios assume a set of pre-specified categories of documents.

Basically, we can distinguish two classes of approaches to automatic text categorization. The first consists in a manual construction of a set of explicit classification rules that are then automatically applied to classify the documents. Thus, this methodology belongs to the field of expert systems. The second approach consists in using techniques of machine learning to automatically produce a classifier. We follow the latter and try to use some elements of fuzzy logic, notably those mentioned in Sections 3 and 4. Another dimension along which the text categorization tasks may be distinguished is that of how many classes are considered and how many categories may be assigned to one document. The most general approach (that we adopt here) assumes a *multiclass multilabel* task, i.e., there are more than two categories and more than one may be assigned to a document. Still another dimension that is possible to distinguish the categorization tasks divides them into two classes depending on whether the documents to be classified are available one at a time ("on-line categorization") or in larger portions ("batch categorization"). This distinction is to some extent formal, but is important from the point of view of thresholding strategies considered later.

Thus, text categorization as discussed here is a typical example of the classification task, cf., e.g., [20]. More precisely, the process consists of two phases:

- learning of classification rules (explicit or implicit; building a classifier) from examples of documents with known class assignment (*supervised learning*),
- classification of documents unseen earlier using rules derived in Phase I.

We start with a numerical representation of documents as discussed in Section 2 and formalized by (4). Then, any of numerous classifier construction algorithms may be applied, including rule-based systems, decision trees, artificial neural networks, etc. One of classical algorithms developed in the area of IR is that of Rocchio [25,12]. The learning phase consists in computing a *centroid* vector for each category of documents. Then, in the classification phase, a document is classified to a category whose centroid is most similar to this document. The similarity may be meant in several ways—in the original Rocchio's approach it corresponds to the Euclidan distance. In the next subsection we propose to apply some fuzzy logic related concepts to build such a classifier. Here, we further precisiate the classification task that is addressed and steps that have to be taken to develop a classifier of the type considered.

In our computational experiments, cf. Section 5.3, we use the Reuters corpus [21] that is widely accepted as a testbed for text categorization algorithms. This is a collection of newswire stories that are usually classified to a number of categories. Thus, this calls for methods dealing with a multiclass and multilabel case. The multilabel categorization requires the solution of an additional problem while building a classifier. Namely, a classifier such as the one considered here produces for a document a list of categories to which it possibly belongs. These categories are ordered non-increasingly according to their matching with the document. Then, a decision has to be made which of them, or more precisely, how many of those from the top of the list are to be assigned to a document under consideration. This is referred to as the thresholding strategy [27,32,33]. Usually [33], the following strategies are considered:

- rank-based thresholding (RCut), (29)
- proportion based assignment (PCut), (30)
- score-based local optimization (SCut). (31)

The first strategy consists in choosing r top categories for each document. Parameter r may be set by the user or automatically tuned (learned) using a part of the training set of documents. The next strategy works for "batch categorization" and assigns to each category such a number of documents from a batch of documents to be classified so as to preserve a proportion of the cardinalities of particular categories in the training set. The last method assigns a document to a category only if a matching score of this category and document is higher than a certain threshold. Thresholds are tuned using a part of the training set of documents, separately for each category. In the next section we propose other strategies using the concept of a linguistic quantifier.

To summarize, decisions related to the following issues have to be made while building a classifier of the type considered here:

(i) representation of documents and queries, of which an integral part is feature selection,
(ii) definition of a distance (matching degree) of a centroid and a document,
(iii) choice of a thresholding strategy.

The next subsection describes our experiments applying the elements of fuzzy logic mainly for the purposes of (ii) and (iii), but also to some extent of (i). The computational experiments require some measures of effectiveness. In our tests we use standard measures as discussed in Section 5.3.

5.2. Fuzzy and linguistic approaches to the construction of a Rocchio type classifier

The Rocchio type classifier fits into a more general scheme of *profile-based classifiers* [27]. The idea is to compute a profile (referred to elsewhere in the text as a centroid) for each category and then to base the categorization of a document on its distance (more generally, some measure of similarity) from centroids of all categories. In this way, we can order all categories from the best to the worst by matching the document content. The origin of the Rocchio style classifier is related to the formula for relevance feedback in the vector space model. It is a way of modifying an original user query so as to take into account his or her feedback as to the relevance of particular documents retrieved according to this original query. More precisely, the user picks out relevant documents, then the system computes their centroid (average), possibly taking into account irrelevant documents as well, by subtracting their centroid from the centroid of relevant documents. Next, such a centroid of the class of relevant documents is used as a modified query to once again retrieve documents from the whole collection. Thus, in terms of the categorization task it corresponds to a binary (only two classes of relevant and irrelevant documents are considered), one-class (each document is classified either as relevant or irrelevant, but not both), and "batch" rather than on-line problem. It is, then, quite different in comparison to multiclass multilabel on-line categorization task addressed here.

Our approach assumes, classically, the computation of centroids (see the next subsection for details) for all categories. Then, a document to be classified (more precisely, its representation) is treated as a query against the set of centroids. Within the extended Boolean framework such a query may be treated as the conjunction of the form (12) or (15). As we pointed out, a simple interpretation of the conjunction as the min operator does not work well for classical queries considered in IR, i.e., queries constructed by a human user. This is true

even to a further extent in case of the categorization task considered here. Thus, in our experiments we try various flexible schemes of aggregation presented in Section 4 as well as, in case of (15), different weight interpretations discussed in the same section.

For our further discussion let us assume the following notation:

$$C = \{c_p\}_{p\in[1,P]}$$

is a set of centroids, one for each of P categories. Each centroid is represented by a vector:

$$c_p = [c_{p1}, \ldots, c_{pM}], \tag{32}$$

where M denotes, as previously, the number of terms used to index the documents. These centroids are constructed in a different way than in the typical Rocchio type classifier. Namely, weights c_{pi} are not calculated directly as the averages of the weights of all training documents belonging to a given category but according to the following formula:

$$c_{pj} = (f_{pj} * \log(P/n_j)) \Big/ \arg\max_j(\log(P/n_j) * f_{pj}), \tag{33}$$

where f_{pj} is a frequency of term j in all documents belonging to category p and n_j is the number of categories in documents of which term j appears (*category frequency*). By analogy to (5) it may be called a $tf \times icf$ representation where *icf* stands for an *inverted category frequency*. A document to be classified d, which is here treated as query q against a set of centroids, is represented, as previously, by a vector, or more precisely as in (15):

$$d = q = [w_1, \ldots, w_M] \equiv (t_1, w_1) \wedge \cdots \wedge (t_M, w_M), \tag{34}$$

where

$$w_j = (f_j * \log(P/n_j)) \Big/ \arg\max_j(\log(P/n_j) * f_j), \tag{35}$$

where f_j is a frequency of term t_j in the document d and n_j is the category frequency of this term, cf. (33). Thus, this setting most naturally fits to the extended Boolean model where both queries and documents are represented using weights. Now, we base our decision on the classification of document d as pertinent to category p on its similarity to a corresponding centroid c_p, i.e., on the matching of query q (34) and this centroid. In order to compute this matching degree we are going to employ our discussion of Section 4, in particular various interpretations of the query weights given by (16)–(18). Let us observe that, in the given context, the similarity of a query and a centroid intuitively means that terms representing them have weights comparable, relatively or absolutely. Thus, the relative importance and ideal weights interpretations seems to be more suitable than the thresholds of importance. In order to

justify the latter claim let us assume that a document is represented by means of 2 terms with high weights and 10 terms with very low weights. Then, the category whose centroid is represented by all 12 terms with high weights that perfectly fits the document/query according to the threshold logic, does not seem to be a good fit.

Below we list and briefly comment upon the matching schemes we tested in our experiments with an automatic text categorization. Their main goal is to overcome a deficiency of the min operator which is reinforced by a high dimensionality of the representation space of both query (34) and documents (32). We group the tested schemes according to the query weight interpretation adopted.

I. Relative importance

M.I.1. Original weights interpretation via the Kleene–Dienes implication (19) and (20). This concept of matching (i.e., document classification) may be linguistically expressed as: "A document matches a category if *all* of the *important* terms present in the document are also *present* in the centroid of the category".

Matching of the elementary and overall queries are computed using (20) and (19), respectively.

M.I.2. Linguistic majority. This concept of matching (i.e., document classification) may be linguistically expressed in as: "A document matches a category if *most* of the *important* terms present in the document are also *present* in the centroid of the category".

The idea refers directly to our previous experiences with a fuzzy database querying (cf. [14–16]). The above linguistic expression is formalized using Zadeh's calculus of linguistically quantified propositions by, cf. (8):

$$QB\text{'}s \text{ are } G\text{'}s,$$

where X, the universe considered, is a set T of all index terms, B is a fuzzy set of terms important for the document d, i.e.,

$$\mu_B(t_j) = w_j = F(d, t_j),$$

and G is a fuzzy set of terms present in centroid c_p of category p, i.e.,

$$\mu_G(t_j) = c_{pj}.$$

Due to a high dimensionality of the considered space and known deficiencies of linguistic quantifiers in the sense of Zadeh (cf. Section 3) we also tested a modified version where only terms weighted in the query higher than a certain threshold are considered.

II. Ideal weights

M.II.1. Cosine. The classical vector space model formula (6) has been employed.

M.II.2. Fuzzy equivalence based approach. The ideal weight logic is here represented using fuzzy equivalence, as expressed by the formulae (27) and (28). The overall matching degree computes using the min, as in (19) or OWmin, as in (13). Also a modified version where only terms weighted in the query higher than certain threshold are considered, has been tested.

III. Thresholding strategies

Concerning the thresholding strategy we also propose an approach based on fuzzy linguistic concepts. The underlying idea may be expressed as follows:

T.I. "Select such a threshold r that *most of the important categories had a number of sibling categories similar to r in the training data set*".

Thus, for each $r \in [1, R]$ we compute the truth degree of the italicized clause above (R is a parameter). This is again formalized using Zadeh's calculus of linguistically quantified propositions as

$$QB\text{'}s \text{ are } G\text{'}s, \tag{36}$$

where X, the universe considered, is a set C of 10 categories with the highest matching score, B is a fuzzy set of important categories for a given document d, i.e.,

$$\mu_B(c_p) = \gamma(q, c_p),$$

where $\gamma(\cdot,\cdot)$ is the matching function (19) used and q is a query/document to be classified. G is a fuzzy set of categories, that, on the average, had in the training set the number of sibling categories similar to r for which the truth value of (36) is calculated. This similarity is modeled by a similarity relation, which is another parameter of the method. For the purposes of this strategy (and others, see below), for each category the average number of sibling categories in the training data set is first computed. By the sibling category for a category c_p we mean a category that is assigned to the same document as the category c_p.

T.II. Another approach exploiting the concept of sibling categories works as follows. Only categories whose matching score is higher than a certain parameter (in our experiments usually 0.2 is assumed) are taken into account. Their scores are normalized (divided by the sum of their scores) and then the weighted sum of the average number of siblings is taken as a threshold cut (rounded to the nearest integer value).

T.III. For the comparison we also tested the simple RCut (29) with a threshold rank equal 2, i.e., two top scored categories are assigned to each document.

The whole classification procedure proceeds then in the following steps:

Training phase

(1) The training documents are read and data on their frequency in documents and categories are gathered. Also the average number of sibling categories is computed for each category (for the thresholding strategy purposes).

(2) The training documents are read once again and the centroids of particular categories are calculated according to formula (33).

Testing phase

(1) A test document is read and its representation is calculated using formula (35) and normalized (in the experiments all coordinates are divided by the highest one).
(2) For each category the matching degree (score) of its centroid and the document is computed using one of the approaches M.I.1–M.II.2 and this vector of scores is sorted in the non-increasing order.
(3) One of the threshold strategies T.I–T.III is used to decide which categories assign to the document.

5.3. A computational experiment and its results

In our general setting for computational experiments we are following Yang and Liu's work [34]. The text corpus used is Reuters-21578 as made available over the Internet by Lewis [21]. More precisely we are using the Modified Apte ("ModApte") Split of the data, i.e., for the training phase a subset of news characterized by the attributes LEWISSPLIT = "TRAIN" and TOPICS = "YES" and for testing phase a subset LEWISSPLIT = "TEST"; TOPICS = "YES". In both cases, we use only news that actually contain topics and body of the text or at least the title. This gives rise to 7728 training, 3005 test documents and 114 categories. The title of the document and its body are concatenated to produce the document. The documents are preprocessed by removing stop words [28] and numbers. Stemming is done using the standard Porter's algorithm [22]. The terms space dimensionality reduction is done using a simple approach based on document and category frequencies of terms. Namely, only terms with a document frequency higher than 3 and a category frequency lower than 75% are used. This rule yields 5565 index terms.

The evaluation of particular approaches tested has been carried out by using standard measures of recall, precision, F1 measure and 11-point average precision. Both micro- and macro-averaging results are presented. These measures are expressed by the following formulae:

- *micro-averaging*

$$\text{precision} = \frac{\text{number of correct classifications made by the system}}{\text{total number of all classifications made by the system}}, \tag{37}$$

$$\text{recall} = \frac{\text{number of correct classifications made by the system}}{\text{total number of all categories indicated in test documents}}, \tag{38}$$

$$F1 = 2 * \text{precision} * \text{recall}/(\text{precision} + \text{recall}). \quad (39)$$

Note that the number of classifications is higher than the number of test documents as more than one category may be assigned to a document. By a system we mean an automatic classifier based on a given approach.

- *macro-averaging*

  First, precision, recall and F1 measure are calculated separately for each category using formulae (37)–(39) and then the arithmetic mean of them is calculated.

In Table 1, we present some of the results of our experiments. In Table 2 we test various thresholding strategies for the linguistic guided aggregation used in M.I.2.

The results for particular matching schemes are essentially worse than the ones reported for state-of-the-art approaches, cf. [34]. However, there is a number of various factors that influence the effectiveness of an automatic

Table 1
Comparison of matching schemes for T.II thresholding strategy

Matching scheme	Micro-averaging			Macro-averaging			11-Point average precision
	Precision	Recall	F1	Precision	Recall	F1	
M.I.1	0.3763	0.5521	0.4475	0.2212	0.3704	0.2770	0.6793
M.I.2	0.3914	0.8215	0.5302	0.4038	0.5322	0.4592	0.8311
M.I.2a[a]	0.4226	0.6765	0.5203	0.3416	0.6174	0.4398	0.7673
M.II.1	0.2226	0.6462	0.3311	0.1235	0.4943	0.1976	0.6511
M.II.2a[b]	0.5597	0.4597	0.5048	0.5601	0.0934	0.1601	0.5926
M.II.2b[c]	0.5847	0.5015	0.5399	0.5231	0.1349	0.2145	0.6356
M.II.2b[d]	0.3809	0.6961	0.4923	0.2978	0.4287	0.3515	0.7397

[a] Only terms weighted above 0.2 are considered in matching degree computation.
[b] Aggregation via min.
[c] Aggregation via min; only terms weighted above 0.2 are considered in matching degree computation.
[d] Aggregation via OWmin; only terms weighted above 0.2 are considered in matching degree computation.

Table 2
Comparison of different thresholding strategies for the M.I.2 matching scheme

Thresholding strategy	Micro-averaging			Macro-averaging			11-Point average precision
	Precision	Recall	F1	Precision	Recall	F1	
T.I	0.5531	0.7765	0.6460	0.4891	0.4785	0.4837	0.8311
T.II	0.3914	0.8215	0.5302	0.4038	0.5322	0.4592	0.8311
T.III	0.4642	0.7478	0.5728	0.4776	0.4309	0.4530	0.8311

categorization system, including the representation of documents, tuning the parameters, and a thresholding strategy. The very nature of fuzzy linguistic approaches makes it possible to tune parameters that possess an interpretation that is easier to grasp by the human user involved. In this short study we only try to check a general applicability of fuzzy logic based concepts for text categorization.

6. Concluding remarks and further research

In the paper we discuss the problem of an automatic text document categorization that has recently attracted a lot of attention and interest. In our approach to the problem we try to use results obtained by other authors proposing some fuzzy logic based extensions to classical IR models, notably the Boolean model. Although they mainly address the primary task of retrieval of the documents relevant to the human user, they are also applicable to text categorization. Our starting point is fuzzy querying of crisp databases. We try to adapt some of the ideas we proposed earlier in this respect, notably of a linguistically guided aggregation of partial matching degrees, for the purposes of text document categorization. We illustrate the preliminary results on a standard document corpus used to test most sophisticated and successful classifiers. Further research will focus on tuning various parameters that are included in our proposed approaches.

References

[1] R. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval, ACM Press/Addison-Wesley, New York/Reading, MA, 1999.
[2] A. Bookstein, Fuzzy requests: an approach to weighted Boolean searches, Journal of the American Society for Information Sciences 31 (1980) 240–247.
[3] G. Bordogna, P. Carrara, G. Pasi, Fuzzy approaches to extend Boolean information retrieval, in: P. Bosc, J. Kacprzyk (Eds.), Fuzziness in Database Management Systems, Physica Verlag, Heidelberg, 1995, pp. 231–274.
[4] G. Bordogna, G. Pasi, Application of fuzzy sets theory to extend Boolean information retrieval, in: F. Crestani, G. Pasi (Eds.), Soft Computing in Information Retrieval, Physica Verlag, Heidelberg, New York, 2000, pp. 21–47.
[5] G. Bordogna, P. Bosc, G. Pasi, Extended Boolean information retrieval in terms of fuzzy inclusion, in: O. Pons, M.A. Vila, J. Kacprzyk (Eds.), Knowledge Management in Fuzzy Databases, Physica Verlag, Heidelberg, New York, 2000, pp. 234–246.
[6] Ch. Carlsson, R. Fuller, A new look at linguistic importance weighted aggregation, in: Proceedings of the Fourteenth European Meeting on Cybernetics and Systems Research, Austrian Society for Cybernetic Studies, Vienna, 1998, pp. 169–174.
[7] M. Delgado, J.L. Verdegay, M.A. Vila, On aggregation operations of linguistic labels, International Journal of Intelligent Systems 8 (1993) 351–370.
[8] D. Dubois, H. Fargier, H. Prade, Beyond min aggregation in multicriteria decision: (ordered) weighted min, discri-min, leximin, in: R.R. Yager, J. Kacprzyk (Eds.), The Ordered Weighted

Averaging Operators. Theory and Applications, Kluwer Academic Publishers, Boston, Dordrecht, London, 1997, pp. 181–192.
[9] D. Dubois, H. Prade, Using fuzzy sets in flexible querying: why and how? in: T. Andreasen, H. Christiansen, H.L. Larsen (Eds.), Flexible Querying Answering Systems, Kluwer Academic Publishers, Dordrecht, 1997.
[10] E. Herrera-Viedma, An information retrieval system with ordinal linguistic weighted queries based on two weighting elements, International Journal of Uncertainty and Knowledge-Based Systems 9 (2001) 77–88.
[11] E. Herrera-Viedma, Modeling the retrieval process of an information retrieval system using an ordinal fuzzy linguistic approach, Journal of the American Society for Information Science and Technology (JASIST) 52 (6) (2001) 460–475.
[12] T. Joachims, A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization, in: Proceedings of ICML-97, 1997.
[13] J. Kacprzyk, G. Pasi, P. Vojtáš, S. Zadrożny, Fuzzy querying: issues and perspectives, Kybernetika 6 (36) (2000) 605–616.
[14] J. Kacprzyk, S. Zadrożny, Computing with words in intelligent database querying: standalone and Internet-based applications, Information Sciences 134 (2001) 71–109.
[15] J. Kacprzyk, S. Zadrożny, A. Ziółkowski, FQUERY III+: a "human-consistent" database querying system based on fuzzy logic with linguistic quantifiers, Information Systems 14 (1989) 443–453.
[16] J. Kacprzyk, A. Ziółkowski, Database queries with fuzzy linguistic quantifier, IEEE Transations on Systems, Man and Cybernetics, SMC 16 (1986) 474–479.
[17] D.H. Kraft, G. Bordogna, G. Pasi, An extended fuzzy linguistic approach to generalize Boolean information retrieval, Journal of Information Sciences 2 (3) (1994) 119–134.
[18] D.H. Kraft, G. Bordogna, G. Pasi, Fuzzy set techniques in information retrieval, in: J.C. Bezdek, D. Didier, H. Prade (Eds.), Fuzzy Sets in Approximate Reasoning and Information Systems, The Handbook of Fuzzy Sets Series, vol. 3, Kluwer Academic Publishers, Norwell, 1999.
[19] D.H. Kraft, D.A. Buell, Fuzzy sets and generalized Boolean retrieval systems, in: D. Dubois, H. Prade, R.R. Yager (Eds.), Readings in Fuzzy Sets for Intelligent Systems, Morgan Kaufmann Publishers, San Mateo, 1992.
[20] L.I. Kuncheva, Fuzzy Classifier Design, Physica Verlag, Heidelberg, New York, 2000.
[21] D.D. Lewis, Reuters-21578, Distribution 1.0. Available from: http://www.research.att.com/~lewis.
[22] M.F. Porter, An Algorithm for Suffix Stripping, Program 14 (3) (1980) 130–137.
[23] T. Radecki, Fuzzy set theoretical approach to document retrieval, Information Processing and Management 15 (1979) 247–260.
[24] C.J. van Rijsbergen, Information Retrieval, Butterworths, London, 1979.
[25] J. Rocchio, Relevance feedback in information retrieval, in: The SMART Retrieval System: Experiments in Automatic Document Processing, Prentice-Hall Inc, Englewood Cliffs, NJ, 1971, pp. 313–323.
[26] G. Salton, E.A. Fox, H. Wu, Extended Boolean information retrieval, Communications of ACM 26 (11) (1983) 1022–1036.
[27] F. Sebastiani, A tutorial on automated text categorisation, in: A. Amandi, A. Zunino (Eds.), Proceedings of ASAI-99, 1st Argentinian Symposium on Artificial Intelligence, Buenos Aires, AR, 1999, pp. 7–35.
[28] Stop words list, http://www.indiana.edu/cgi-bin-local/doIsearch.pl?Stopwords.
[29] R.R. Yager, A note on weighted queries in information retrieval systems, Journal of the American Society for Information Science 38 (1987) 23–24.
[30] R.R. Yager, On ordered weighted averaging aggregation operators in multi-criteria decision making, IEEE Transactions on Systems, Man and Cybernetics 18 (1988) 183–190.

[31] R.R. Yager, J. Kacprzyk (Eds.), The Ordered Weighted Averaging Operators: Theory and Applications, Kluwer, Boston, 1997.
[32] Y. Yang, An evaluation of statistical approaches to text categorization, Journal of Information Retrieval 1 (1/2) (1999) 67–88.
[33] Y. Yang, A study on thresholding strategies for text categorization, in: Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01), 2001.
[34] Y. Yang, X. Liu, A re-examination of text categorization methods, in: Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99), 1999, pp. 42–49.
[35] L.A. Zadeh, The concept of linguistic variable and its applications to approximate reasoning, Parts I, II, III, Information Sciences 8 (1975) 199–251, 301–357, 9 (1975) 43–80.
[36] L.A. Zadeh, A computational approach to fuzzy quantifiers in natural languages, Comput. Math. Appls. 9 (1983) 149–184.
[37] S. Zadrożny, K. Ławcewicz, J. Kacprzyk, Intelligent linguistic characterization and retrieval of textual documents: an Internet-based application, in: Proceedings of the IPMU'2002 conference, Annecy, France, 2002, pp. 1223–1230.