# Information Retrieval: From Language Models to Fuzzy Logic

Rima Harastani

Internship Report

# Information Retrieval: From Language Models to Fuzzy Logic

Rima Harastani (MR2I parcours P4)

June 4, 2010

**Team:** Texmex, IRISA
**Supervisors:** Vincent Claveau and Laurent Ughetto

# Contents

# 1 Introduction

This internship is concerned in modeling and testing an information retrieval (IR) system. These systems search in the content of a large collection of documents and retrieve the ones that are relevant to a user's demand, which is usually represented in the form of a query. The content of a document is generally indexed by terms, users' queries are also indexed by terms that are used to identify topics of interest. Terms in documents and queries are usually given specific weights in accordance to their significance.

An information retrieval system has a matching function that assigns scores to documents, each score represents a degree of relevance between a document and a query. Documents are then ranked according to their scores, and top n-documents are retrieved to the user.

The subject of the internship aims to exploit fuzzy logic concepts as well as language modeling approaches for IR in order to build and test an information retrieval model. In fact, language models (LM) define an approach to score documents in IR systems. Different language models approaches for IR have shown good performance. On the other hand, fuzzy logic (FL) is an extension of boolean logic, in which truth values belong to the interval [0,1]. It introduces theoretical foundations to term weighting schemes in information retrieval systems. A theoretical work on fuzzy logic in the database (DB) has been continued for many years by the Pilgrim team (IRISA, Lannion). And it has been recently extended to the field of IR in cooperation with Texmex team (IRISA, Rennes) with a theoretical and experimental approach. Texmex obtained results showing the validity of the model, and generalizing classical IR vector space model (VSM) [1].

The aim of this internship is to explore other tracks of work done by Texmex team, by trying to extend a language model approach using the foundations of fuzzy logic, and consequently to conclude if this extension could correspond to or outperform the classical model.

This report starts by reviewing the basic theoretical concept of IR systems. Section 2 introduces language models and some language modeling approaches for information retrieval. Section 3 represents some fuzzy logic concepts, and a fuzzy representation (proposed in this internship) of a language modeling approach. In section 4, we will show how we extended the proposed fuzzy representation using semantically related words.

# 2 Information Retrieval (IR)

## 2.1 IR System

An Information Retrieval (IR) System attempts to retrieve, from a collection of documents, those relevant documents that correspond to a users request. Models of information retrieval systems are characterized by three main

components [1]: the representation of documents, the query language, and the matching mechanism. A document's representation is generated by an indexing process which represents the content of a document as indexing terms.

The retrieval process starts when a user submits his information need to the system, this information is represented by a query in the system's query language. Then, a matching mechanism evaluates the user's query against the representations of documents and retrieves those that are considered to be relevant [1]. Figure 1 represents a classical view of the information retrieval process. Different approaches have been proposed to the problem of IR. One may refer to [3] for further reading about some existing models in IR.

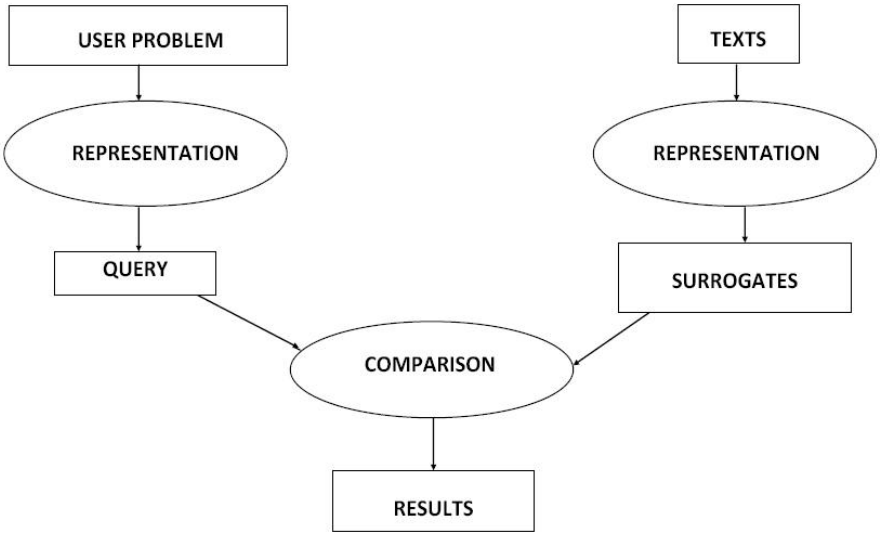


Figure 1: Information Retrieval Process

## 2.2 IR Evaluation

The evaluation of an IR system is the process of measuring the retrieval effectiveness, i.e. how well the system meets the information needs of its users [20]. Some measures have been proposed to evaluate the performance of an IR system. These measures require a collection of documents and a set of queries, in addition to a set of relevance judgment (which documents are relevant to each query).

Most common evaluating measures are precision and recall. Precision is the fraction of a retrieved set that is relevant, and Recall is the frac-

tion of all relevant documents in the collection included in the retrieved set. Other measures are non-interpolated average precision (NIAP), interpolated average precision (IAP), R-precision [1]. We use these measures in our experiments in the next sections. When comparing the change in performance between two systems, we use statistical tests to verify that the difference is significant.

# 3 Language Models for IR

Since the internship subject aims at representing a language model approach using fuzzy concepts, a first part of our work is to investigate the use of language models in IR. We will present, in this section, the concept of language models and propose different experiments to compare their performances.

## 3.1 Language Models

Language models can be defined as the task of estimating a probability distribution over a sequence of words. It is common to estimate this probability using N-grams. In an N-gram model, the probability of a word is predicted based on the previously N-1 seen words. This means that if we take the sequence of words S: $w_1, w_2, ...w_n$, its probability $P(S)$ is calculated as follows [5]:

$$P(S) = \prod_{w_i \in S} (w_i | w_{i-n+1}, ..., w_{i-1}) \tag{1}$$

Generally, $\prod_{w_i \in S}(w_i|w_{i-n+1}, ..., w_{i-1})$ is estimated by counting, in a corpus, the number of occurrences of the sequence $w_{i-n+1}, ..., w_{i-1}, w_i$ over the number of occurrences of the sequence $w_{i-n+1}, ..., w_{i-1}$.

## 3.2 Language Modeling approaches for IR

Language Models have been successfully applied to information retrieval, giving prominent results when compared to other classical information retrieval systems (vector space model, tf-idf weighting...). Moreover, several language modeling approaches have been proposed to the IR problem.

A language modeling approach to IR considers a document as a language sample and estimates the probabilities of producing the terms of a given query from this document [5]. The probability that a term has been generated from a document, $P(t|d)$, is estimated by the maximum likelihood estimator. This estimator calculates a term's probability by counting the number of times the term was seen in the document (i.e. term frequency count, represented as $tf$), and then dividing this count by the document's length:

$$P(t|d) = \frac{tf}{\sum_{t \in d} tf} \tag{2}$$

The probability of a document (d) given a query (q), which is also the score given to the document, decomposes into a product of the probabilities of individual terms.

$$score(d, q) = \prod_{t \in q} P(t|d) \tag{3}$$

### 3.3 Smoothing

No matter what the size of the corpus is, there would always exist some absent terms or sequences of terms [6]. This is called the sparse data problem; since giving a zero probability to an absent term from the document would set the score of this latter into zero. To address this problem, a technique called smoothing is used, and it has become an important part of any language model, and is therefore important for an IR system based on language models. It intends to deal with the problem of data spareness by giving non-zero probabilities to absent terms.

Different smoothing techniques have been proposed. Two types of smoothing, Backoff Smoothing and the Interpolation Smoothing, are detailed hereafter. We will study some smoothing techniques that can be used in a language model approach for IR. Besides, we will make a comparison of the results obtained after applying these models on two collections of documents.

### 3.4 Back-off Smoothing (Model of Ponte and Croft)

Back-off smoothing is based on the idea of preserving the probabilities of high count terms, while using lower order N-grams to estimate the less common terms [7].

Ponte and Croft model [8] was the first language model approach to IR, it assumes that the terms that exist in the document but not in the query affect the document's score:

$$score(d, q) = \prod_{t \in q} P_s(t|d) \times \prod_{t \notin q} (1 - P_s(t|d)) \tag{4}$$

This model uses a backoff-like smoothing by considering that the probability of a term that does not appear in a document is no more than what it would be expected by chance in the collection [8]. While the probability of an existing term is calculated by mixing some averaged information from the corpus with the local probability of term in document, these two information are then combined by a weighted product to form the final estimation of term's probability $P_s(t|d)$:

$$P_s(t|d) = \quad P(t|d)^{(1-\hat{R}_{t,d})} \times P_{avg}(t)^{\hat{R_{t,d}}} \quad if \quad tf > 0 \tag{5}$$

$$= \frac{cf_t}{cs} \qquad otherwise \tag{6}$$

Where $P_{avg}$ is the average probability of $t$ in the documents in which $tf(t) > 0$. And $\hat{R_{t,d}}$ is the risk for the term $t$, the intuition behind the use of this risk is that as the $tf$ gets further away from the mean term frequency $\overline{f}$ of term $t$ in documents where $t$ occurs, the mean probability $P_{avg}$ becomes riskier to use as an estimate. This risk function is used as a mixing parameter in the calculation of $P(t|d)$.

### 3.5 Interpolation Smoothing

In interpolation smoothing, the counts of seen terms are first discounted, and then the extra counts are shared by both the seen and the unseen terms [7].

We will see in this section how three different interpolation methods could be applied as smoothing techniques in language modeling approaches for IR. It is important to mention that in these approaches, the score of a document is calculated by multiplying the probabilities of seeing each query term in the document.

$$score(d, q) = \prod_{t \in q} P_s(t|d) \tag{7}$$

#### 3.5.1 Jelinek-Mercer smoothing (Model of Hiemstra)

Hiemstra [4] uses the Jelinek-Mercer smoothing technique to calculate the probability of each term in query. This is done by adding the probability of its appearance in corpus to the probability of its existence in the document. The probability $P_s(t|d)$ of a term $(t)$ produced by document $(d)$ would be calculated as follows:

$$P_s(t|d) = \lambda_1 P(t|d) + \lambda_2 P(t|C) \quad ; \quad \lambda_1, \lambda_1 \geq 0, \quad \lambda_1 + \lambda_2 = 1 \tag{8}$$

In the last formula, $P(t|C)$ is the probability of seeing a term in the corpus, estimated by counting the number of documents the term appeared in called $df$, and dividing it by the sum of $df$ of all terms in the collection. The probabilities of seen terms are discounted after being multiplied by $\lambda_1$ . This means that global information of a term is mixed with its local information by linear interpolation.

Hiemstra has proven that the language modeling approach for IR corresponds to the vector space model (VSM) [1], but using a different term weighting scheme [4]. Best results were obtained when $\lambda_1$ was set to 0.15.

### 3.5.2 Absolute Discounting

Absolute Discounting [7] is another interpolation method that can be used to calculate terms' probabilities while trying to address the sparse data problem. Here, discounting is done by subtracting a constant from the frequency count of each seen term before calculating its probability. The formula is given as:

$$P_s(t|d) = \frac{\max(tf - \delta, 0)}{\sum_{t \in d} tf} + \sigma P(t|C) \quad ; \quad \delta > 0 \quad and \quad \sigma = \delta \frac{|d|_u}{|d|} \tag{9}$$

Where $|d|_u$ is the number of unique terms in the document, and $|d|$ is the total count of all terms in the document. Experiments carried in [7] has set $\delta = 0.7$ to obtain good results.

### 3.5.3 Dirichlet

A third interpolation technique that has been introduced in [7], is the Dirichlet method.

$$P_s(t|D) = \frac{tf + \mu P(t|C)}{\sum_{t \in d} tf + \mu} \tag{10}$$

If we see this formula as the sum of two parts, we conclude that the probability of a seen term is discounted after dividing the term's count by the sum of a constant $\mu$ and the document's length. A value of $\mu = 2000$ was used in experiments given in [7].

## 3.6 Smoothing Probabilities by Other Methods

Additive smoothing [9] is a simple smoothing technique; in which we add a non-zero constant to the frequency count of each term. We have:

$$P(t|d) = \frac{\delta + tf}{\delta |V| + \sum_{t \in d} tf} \tag{11}$$

Where $|V|$ is the size of all occurring terms in collection.

Another smoothing method is Good-Turing [9], it was used in [10] and claimed to give good results.

We have proposed to use additive smoothing to calculate probabilities before applying other smoothing techniques. In other words, if we take the model of Hiemstra (8), we see that it consists of the weighted sum of two probabilities. Thus, we can smooth each of these probabilities individually before combining them to produce an interpolation smoothing.

### 3.7 Experimental Setup

Before continuing our work of integrating fuzzy logic to language modeling approach to IR, we have implemented the above-mentioned methods to better understand the models and to compare the results.

Experiments were carried out on two collections of documents written in French language. The first one is the INIST collection, it contains 163,308 documents (paper abstracts from various scientific disciplines) and a set of 30 long queries. The second is the ELDA collection, which contains 3500 documents and a set of 30 short queries. The queries are composed of several fields, a title, a subject, a description and a set of associated concepts. In the experiments given in this report, titles and subjects where used as actual queries. The implementation that we made was based on a program developed by Texmex team, which was an implementation of Hiemstra's model [4], we adapted this program to implement the new models that we wanted to test. The language used for implementing models was Perl, as it provides powerful text processing facilities.

Additionally, we used an implemented program made by Texmex team to evaluate the information retrieval models. This program allows to evaluate a retrieval run given the results file and a standard set of judged results. It calculates different evaluating measures, and it indicates which observed changes are statistically different.

### 3.8 Comparison of Models

Tables 1 and 2 show the results obtained by applying the above mentioned models on ELDA and INIST collections respectively. The change showed beside each value is the improvement when compared to the value obtained with Hiemstra's model used as the baseline for all experiments. And the symbol **S** indicates that the change is significant.

| Measure | Hiemstra | Dirichlet $\mu = 300$ | Absolute $\delta = 0.7$ | Ponte & Croft |
|---|---|---|---|---|
| NIAP | 53.26% | 53.45% (0.37%) | 52.55% (-1.32%) **S** | 52.38 (-1.64%) |
| IAP | 53.81% | 53.89% (0.15%) | 53.12% (-1.28%) **S** | 53.43 (-0.70%) |
| Rprec | 52.68% | 53.86% (2.24%) **S** | 51.72% (-1.82%) | 52.75 (0.13%) |
| P5 | 72.00% | 72.67% (0.93%) | 72.67% (0.93%) | 69.33 (-3.70%) |
| P10 | 68.67% | 68.00% (-0.97%) | 67.00% (-2.43%) | 68.00 (-0.97%) |
| P100 | 24.23% | 23.90% (-1.38%) | 23.70% (-2.20%) **S** | 23.83 (-1.65%) |
| P500 | 5.95% | 6.07% (2.13%) **S** | 6.05% (1.68%) | 6.05 (1.68%) |

Table 1: Language models approaches applied on ELDA

The results obtained after applying the previous three language modeling approaches that use interpolation smoothing, are of a slightly higher

| Measure | Hiemstra | Dirichlet $\mu = 300$ | Absolute $\delta = 0.7$ | Ponte & Croft |
|---|---|---|---|---|
| NIAP | 15.16% | 15.17% ( 0.09%) | 14.56% (-3.94%) | 14.14% (-6.68%) |
| IAP | 12.68% | 12.40% (-2.20%) | 12.60% (-0.56%) | 13.14% (3.67%) |
| Rprec | 19.11% | 19.24% (0.69%) | 19.04% (-0.40%) | 18.18% (-4.90%) |
| P5 | 42.67% | 42.00% (-1.56%) | 44.00% (3.13%) | 42.00% (-1.56%) |
| P10 | 35.33% | 36.00% (1.89%) | 35.67% (0.94%) | 33.67% (-4.72%) |
| P100 | 12.80% | 13.37% (4.43%) | 12.57% (-1.82%) | 12.17% (-4.92%) |
| P500 | 4.63% | 4.66% (0.58%) | 4.34% (-6.04%) | 4.33% (-6.62%) |

Table 2: Language models approaches applied on INIST

precision than this one of Ponte and Croft's, which actually uses a back-off smoothing method. It is in accordance with [7], where it is stated that interpolation smoothing gives better performance than back-off smoothing in general. This could be due to the fact that, with back-off smoothing, a possibly high value of $P(t|C)$ is assigned only to unseen terms, a term could then contribute more to the score of a document in which it does not occur. While in interpolation smoothing models, $P(t|C)$ contributes to the score of a document whether the term $t$ is seen or not. In [8], it is said that one must think of a better way of estimating $P(t|C)$, which is calculated by $\frac{cf(t)}{\sum_{t_i} cf(t_i)}$. We tried to use a different estimation method for $P(t|C)$, taking for example this used by Hiemstra i.e. $\frac{df(t)}{\sum_{t_i} df(t_i)}$, and we got a slightly better NIAP for both ELDA and INIST.

As for comparing interpolation smoothing methods, we see that Hiemstra and Dirichlet have better precision average compared to this one of Absolute counting when applied on both ELDA and on INIST.

Finally, we also tried to smooth the probability of a term in document $P(t|d)$ and in corpus $P(t|C)$ by Additive smoothing method, before applying Hiesmtra's interpolation smoothing. We got an NIAP of 15.26% for INIST collection when taking $\sigma = 10^{-6}$, this gives a change of 0.66% compared to Hiemstra's original NIAP, and a change of 0.3% when taking $\sigma = 10^{-5}$ for ELDA. This may be because terms' probabilities in the documents and in the corpus were normalized.

### 3.9 Conclusion

In this section we have studied different language modeling approaches to IR. We have tested some smoothing techniques that make an essential difference between one language modeling approach to another. Finally, we concluded that there are little differences between the performance of these models when applied on our two document collections. In the next section, we

are going to investigate the use of fuzzy logic with a language modeling approach.

# 4 Representing Language Models by fuzzy logic

Concepts of fuzzy logic have been applied in different fields of computer science, as well as in IR. Several IR systems have been proposed while inspired by the notions of fuzzy logic (Fuzzy sets, Fuzzy operators, Possibility theory).

## 4.1 Fuzzy sets

Fuzzy logic [11] is a multi-valued extension of binary logic, where truth values belong to the interval [0,1].

A subset is called fuzzy when its elements have degrees of membership that take values in the interval [0,1]. These degrees of membership are subjective measures that depend on the context, and may represent degrees of similarity, of uncertainty, of preference... or etc [12].

## 4.2 Fuzzy Model for IR

Fuzzy set theory can be used in a wide range of domains in which information is incomplete or imprecise, where fuzzy sets may be used to show the relationship or degree of imprecision, of uncertainty or of graduality. In the this section, we will try to represent a language model approach for IR using some notions of fuzzy logic.

In the fuzzy approach for IR introduced in [13], documents and queries are represented by fuzzy sets, where terms represent the elements of these fuzzy sets. Each element has a degree of membership that indicates its significance in the documents or its preference in the queries. These degrees are paired with fuzzy operators.

Keeping this assumption in mind, we will take Hiemstra's language model approach for IR and try to extend it using fuzzy notions. We remind that Hiemtra assumes the following:

$$score(q,d) = \prod_{t \in q} (\lambda_1 P(t|d) + \lambda_2 P(t|C)) \quad ; \lambda_1, \lambda_2 > 0, \quad \lambda_1 + \lambda_2 = 1 \quad (12)$$

In this formula, we notice that probabilities of individual terms are assembled by the product. However, we know that the conjunction logical operator $\wedge$ could be represented by a product. We also see, in the model, that smoothing is achieved by adding the probability of seeing the term in a document to the probability of seeing it in the corpus. Moreover, the sum operator that combines probabilities can be viewed as a disjunction operator

$\vee$.
Considering the previous analysis, we assume that it is possible to replace the product by a fuzzy And (which is a t-norm operator). Additionally, we assume that the smoothing method corresponds to a fuzzy Or (which is a t-conorm operator) that aggregates two degrees, the first is the degree of membership for the term in document $\mu_d$, while the second is the membership of term in corpus $\mu_C$.

Consequently, the formula will be written under the following form:

$$score(q,d) = \top_{t \in q}\Big(\bot(\lambda_1.\mu_d, \lambda_2.\mu_C)\Big) \quad ; \lambda_1, \lambda_2 \in ]0,1[ \tag{13}$$

Moreover, we can generalize the t-norm and t-conorm used in the above formula by replacing the t-norm and t-conorm with two different general fuzzy aggregation functions:

$$score(q,d) = Aggreg_{t \in q}(Aggreg^{'}\Big(\lambda_1.\mu_d(t), \lambda_2.\mu_C(t))\Big) \quad ; \lambda_1, \lambda_2 \in ]0,1[ \tag{14}$$

In the next section, we are going to present some aggregation functions that were used to test and evaluate this fuzzy model.

### 4.3 Fuzzy Aggregation Operators

Fuzzy Aggregation operators [14] are functions that take multiple values and return a single representative number also in [0,1]. They can be useful in IR to aggregate individual term scores into a document's score. Different kinds of aggregation operators that were used in our experiments are detailed here. Figure 2 represents these aggregation operators.

#### 4.3.1 T-norms (Fuzzy AND)

A triangular norm (t-norm) is a function of the form $\top : [0,1] \times [0,1] \rightarrow [0,1]$ that satisfies the following properties:

- Commutativity:$\top(a,b) = \top(b,a)$
- Monotonicity: $\top(a,b) \leq \top(c,d) \quad if \quad a \leq c \quad and \quad b \leq d$
- Associativity: $\top(a, \top(b,c)) = \top(\top(a,b), c)$
- Number 1 acts as identity element: $\top(a,1) = a$.

T-norms are a generalization of the usual two-valued logical conjunction.

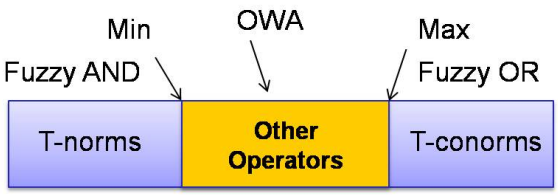


Figure 2: Aggregation Operators

#### 4.3.2 T-conorms Or S-norms (Fuzzy OR)

A t-conorm (also called s-norm) is a dual notation of the t-norm, denoted $\perp$ or S, it is of the form: $\perp : [0,1] \times [0,1] \rightarrow [0,1]$. T-conorms satisfy the same properties as t-norms except that the number 0 acts as the identity element: $\perp(a,0) = a$. T-conorms are used to represent logical disjunctions and union in fuzzy set theory.

#### 4.3.3 OWA

The Ordered Weighted Averaging operators (OWA) extend the space of quantifiers from the *min* to the *max* [15]. They permit to aggregate values $(x_1, x_2, ..., x_n)$ with accordance to a specific criteria. For example, we can order values by descending order, and then we give small or high weights to high values. These weights $(w_1, w_2, ..., w_n)$ can be generated by a geometrical function.

$$OWA(x_1, x_2, ...x_n) = \sum_{j=1}^{n} w_j x_{\alpha(j)} \tag{15}$$

Where $\alpha$ is a permutation that orders the elements: $x_{\alpha(1)} \leq x_{\alpha(2)} \leq ... \leq x_{\alpha(n)}$

#### 4.3.4 OWGA

Ordered Weighted Geometric Averaging operators (OWGA) resemble OWA except for that the latter uses the weighted product instead of the weighted sum to aggregate values.

$$OWGA(x_1, x_2, ...x_n) = \prod_{j=1}^{n} x_{\alpha(j)}^{w_j} \tag{16}$$

#### 4.3.5 Sugeno and Choquet fuzzy Integral

The Sugeno discrete integral can be viewed under aggregation functions [18].

$$Sugeno_\mu(x_1, x_2, ..., x_n) = \max_i^n \Big( \min(x_{\alpha(i)}, \mu(C_{\alpha(i)})) \Big) \tag{17}$$

Where $\alpha$ is a permutation that orders the elements: $x_{\alpha(1)} \leq x_{\alpha(2)} \leq ... \leq x_{\alpha(n)}$, and where $C = \left\{c_{\alpha(1)}, c_{\alpha(2)}, ..., c_{\alpha(n)}\right\}$.

It aggregates values $(x_1, x_2, ..., x_n)$ for criteria $(c_1, c_2, ..., c_n)$ with respect to a fuzzy measure $\mu : P(C) \rightarrow [0, 1]$ which satisfies the following axioms:

- $\mu(\emptyset) = 0$ and $\mu(C) = 1$ Boundary conditions
- For $A, B \in P(C)$, if $A \subset B$ then $\mu(A) \leq \mu(B)$ Monotonicity

Another discrete integral operator is the Choquet integral, it aggregates values $(x_1, x_2, ..., x_n)$ for criteria $(c_1, c_2, ..., c_n)$ with respect to a fuzzy measure $\mu : P(C) \rightarrow [0, 1]$.

$$Choquet_\mu(x_1, x_2, ..., x_n) = \sum_{i=1}^{n} (x_{\alpha(i)} - x_{\alpha(i-1)}).\mu(C_{\alpha(i)}) \tag{18}$$

It has been proven in [18], when $(\mu(A) = max_{x_i \in A} x_i \left[\mu(\{x_i\})\right]$ and $\mu(x_i) = w_i$ for all $i)$ that a Sugeno-integral corresponds to a weighted-maximum operator, which is:

$$\underset{w_1 ... w_n}{\overset{\otimes}{\max}} (x_1, x_2, ..., x_n) = \overset{n}{\underset{i=1}{\max}} \left[\min(w_i, x_i)\right] \tag{19}$$

### 4.4 Results

We made some experiments both on ELDA and INIST collections using the fuzzy model in (13, 14). In the following, we show and discuss some results. We first wanted to see the potential t-norms that could be used in the fuzzy model, so we kept the weighted sum used in (8) to aggregate $\mu_C(t)$ and $\mu_d(t)$.

Here are some t-norms that were used in experiments, some of the t-norms are parametrized, taking $\gamma$ as a parameter, and we had to try many values as parameters in order to see which operators perform well:

- Hamacher: $\frac{a.b}{\gamma+(1-\gamma).(a+b-a.b)} \quad where \quad \gamma \geq 0$
- Einstein: $\frac{a.b}{2-(a+b-a.b)}$
- Dubois and Prade: $\frac{a.b}{max(a,b,\gamma)} \quad where \quad \gamma \in [0, 1]$

  We also tested Dombi, Drastic, Yager, Frank, Lukasiewicz, Schweizer-Skalar, Weber and Yu t-norms [14].

The use of *min* t-norm as a replacement of product is not convenient to the problem since it considers only the lowest $\mu_d(t)$ for all query terms.

| | Hiemstra | Hamacher ($\gamma = 3$) | Einstein | Dubois & Prade ($\gamma = 0.003$) |
|---|---|---|---|---|
| NIAP | 53.26% | 53.27% (0.02%) | 53.27% (0.02%) | 53.86% (1.12%) |
| IAP | 53.81% | 53.81% (0.01%) | 53.81 (0.01%) | 54.18 (0.69%) |
| Rprec | 52.68% | 52.68 (0.00%) | 52.68% (0.00%) | 53.76 (2.05%) |
| P5 | 72.00% | 72.00% (0.00%) | 72.00% (0.00%) | 73.33% (1.85%) |
| P10 | 68.67% | 68.67% (0.00%) | 68.67 (0.00%) | 70.00% (1.94%) |
| P100 | 24.23% | 24.27% (0.14%) | 24.27 (0.14) | 23.77% (-1.93%) |
| P500 | 5.95% | 5.95% (0.00%) | 5.95% (0.00%) | 5.89% (-1.01%) |

Table 3: Fuzzy model on ELDA collection using t-norms

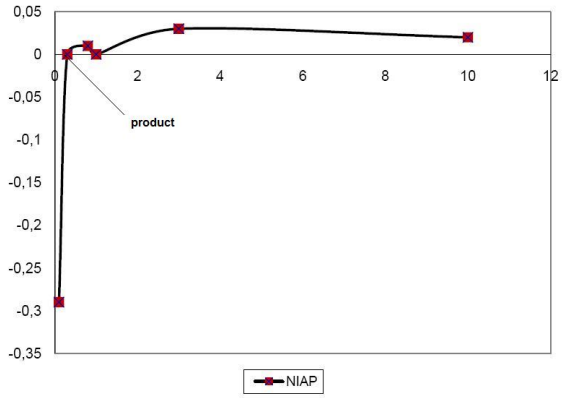


Figure 3: Choosing Parameters for Dubois & Prade t-norm

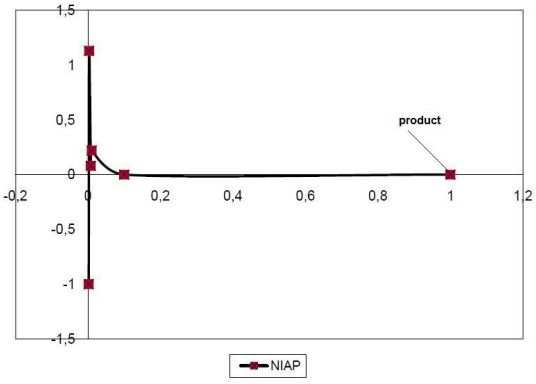


Figure 4: Choosing Parameters for Hamacher t-norm

While the product operator let all weights contribute equally to the final score. Results obtained after trying different t-norms showed that these operators performed best when they function similarly to product. For example, Weber operator performs always worse than product when the

| | Hiemstra | Hamacher ($\gamma = 0.8$) | Einstein | Dubois & Prade ($\gamma = 0.008$) |
|---|---|---|---|---|
| NIAP | 15.16% | 15.16% (0.05%) | 15.16% (-0.00%) | 15.20% (0.28%) |
| IAP | 12.68% | 13.08% (3.21%) | 12.47% (-1.60%) | 16.72% (31.87%) |
| Rprec | 19.11% | 19.11% (0.00%) | 19.11% (0.00%) | 18.80% (-1.63%) |
| P5 | 42.67% | 42.67% (0.00%) | 42.67% (0.00%) | 43.33% (1.56%) |
| P10 | 35.33% | 35.33% (0.00%) | 35.33% (0.00%) | 35.67% (0.94%) |
| P100 | 12.80% | 12.80% (0.00%) | 12.80% (0.00%) | 12.70% (-0.78%) |
| P500 | 4.63% | 4.63% (0.00%) | 4.63% (0.00%) | 4.64% (0.14%) |

Table 4: Fuzzy model on INIST collection using t-norms

parameter is of a small value, it only gives a somewhat reasonable result when the parameter is of a very big value. The reason is that as Weber's parameter converges to infinity, the operator itself converges into product. Other operators such as Lukasiewicz, Frank, Drastic, or Yager do not give good precision results.

The t-norms that gave almost similar or slightly better results than the original model, were: Hamacher, Einstein, and Dubois and Prade. The three operators can be seen as a sort of normalized product. Figures 2 and 3 show the variance in NIAP for different chosen parameters.

We also tried to see how could some t-conorm work as a smoothing operator along with t-norm (Eintein, Hamacher, and Duboi and Prade). Some t-conorms operators used in experiments were:

- Hamacher: $\frac{a+b-a.b-(1-\gamma).a.b}{1-(1-\gamma).a.b} \quad where \quad \gamma \geq 0$
- Weber: $\min(\frac{(a+b-\gamma.a.b)}{(1+\gamma)}, 1)$

| | | Hamacher T-norm $\gamma = 0.8$ | | |
|---|---|---|---|---|
| | Hiemstra | Weber ($\gamma = -25$) | Weber-Sugeno ($\gamma = -200$) | Hamacher ($\gamma = 3$) |
| NIAP | 53.26% | 53.26% (0.012%) | 53.26% (0.012%) | 53.26% (0.00%) |
| IAP | 53.81% | 53.81% (0.00%) | 53.81% (0.00%) | 53.82% (0.03%) |
| Rprec | 52.68% | 52.68% (0.00%) | 52.68% (0.00%) | 52.68% (0.00%) |
| P5 | 72.00% | 72.00% (0.00%) | 72.00% (0.00%) | 72.00% (0.00%) |
| P10 | 68.67% | 68.67% (0.00%) | 68.67% (0.00%) | 68.67% (0.00%) |
| P100 | 24.23% | 24.23% (0.00%) | 24.23% (0.00%) | 24.23% (0.00%) |
| P500 | 5.95% | 5.95% (0.00%) | 5.95% (0.00%) | 5.95% (0.00%) |

Table 5: Fuzzy model on ELDA collection using Hamacher t-norm with t-conorms

| | | Dubois and Prade T-norm $\gamma = 0.005$ | | |
|---|---|---|---|---|
| | Hiemstra | Weber ($\gamma = -25$) | Weber-Sugeno ($\gamma = -25$) | Hamacher ($\gamma = 3$) |
| NIAP | 53.26% | 53.72% (0.86%) | 53.90 (1.21%) **S** | 53.72% (0.86%) |
| IAP | 53.81% | 54.08% (0.50%) **S** | 54.25% (0.82%) **S** | 54.07% (0.49%) |
| Rprec | 52.68% | 53.70% (1.94%) **S** | 53.48% (1.51%) **S** | 53.70% (1.94%) |
| P5 | 72.00% | 72.67% (0.93%) | 72.00% (0.00%) | 72.67% (0.93%) |
| P10 | 68.67% | 70.00% (1.94%) **S** | 69.33% (0.97%) | 70.00% (1.94%) |
| P100 | 24.23% | 24.00% (-0.96%) | 23.90% (-1.38%) | 23.97% (-1.10%) |
| P500 | 5.95% | 5.95% (0.00%) | 5.95% (0.11%) | 5.95% (0.00%) |

Table 6: Fuzzy model on ELDA collection using Dubois & Prade t-norm with t-conorms

| | | Hamacher T-norm $\gamma = 0.8$ | | |
|---|---|---|---|---|
| | Hiemstra | Weber ($\gamma = -25$) | Weber-Sugeno ($\gamma = -200$) | Hamacher ($\gamma = 3$) |
| NIAP | 15.16% | 15.16% (0.05%) | 15.29% (0.90%) **S** | 15.16% (0.058%) **S** |
| IAP | 12.68% | 13.08% (3.20%) **S** | 13.08% (3.17%) | 13.09% (3.27%) **S** |
| Rprec | 19.11% | 19.11% (0.00%) | 18.94% (-0.91%) | 19.11% (0.00%) |
| P5 | 42.67% | 42.67% (0.00%) | 42.00% (-1.56%) | 42.67% (0.00%) |
| P10 | 35.33% | 35.33% (0.00%) | 35.67% (0.94%) | 35.33% (0.00%) |
| P100 | 12.80% | 12.80% (0.00%) | 12.73% (-0.52%) | 12.80% (0.00%) |
| P500 | 4.63% | 4.63% (0.00%) | 4.67% (0.86%) | 4.63% (0.00%) |

Table 7: Fuzzy model on INIST collection using Hamacher t-norm with t-conorms

| | | Dubois and Prade T-norm $\gamma = 0.008$ | | |
|---|---|---|---|---|
| | Hiemstra | Weber ($\gamma = -25$) | Weber-Sugeno ($\gamma = -200$) | Hamacher ($\gamma = 3$) |
| NIAP | 15.16% | 15.20% (0.29%) | 15.28% (0.82%) **S** | 15.20% (0.28%) |
| IAP | 12.68% | 16.71% (31.86%) | 16.82% (32.68%) **S** | 16.71% (31.85%) |
| Rprec | 19.11% | 18.80% (-1.63%) | 18.94% (-0.93%) | 18.80% (-1.63%) |
| P5 | 42.67% | 43.33% (1.56%) | 42.67% (0.00%) | 43.33% (1.56%) |
| P10 | 35.33% | 35.67% (0.94%) | 35.33% (0.00%) | 35.67% (0.94%) |
| P100 | 12.80% | 12.70% (-0.78%) | 12.67% (-1.04%) | 12.70% (-0.78%) |
| P500 | 4.63% | 4.64% (0.14%) | 4.67% (0.86%) | 4.63% (0.00%) |

Table 8: Fuzzy model on INIST collection using Dubois & Prade t-norm with t-conorms

Indeed, weights $\lambda_1$, $\lambda_2$ multiplied by $P(t|d)$, $P(t|C)$ respectively in Hiemstra's model (8) do not have the constraint of summing up to 1 in our fuzzy

model. The only constraint is that the degrees must stay in the interval [0,1]. Consequently, we can manipulate the weights more freely. For example, in results obtained in tables 9, 10 from ELDA, we set $\lambda_1$=0.999, and $\lambda_2$=0.3.

| | | Hamacher T-norm $\gamma = 0.8$ | | |
|---|---|---|---|---|
| | Hiemstra | Weber ($\gamma = -25$) | Weber-Sugeno ($\gamma = -200$) | Hamacher ($\gamma = 3$) |
| NIAP | 53.26% | 53.39% (0.26%) | 53.56% (0.56%) **S** | 53.39% (0.25%) |
| IAP | 53.81% | 54.23% (0.79%) **S** | 54.33% (0.97%) **S** | 54.23% (0.78%) **S** |
| Rprec | 52.68% | 53.39% (1.34%) **S** | 52.96% (0.53%) | 53.39% (1.34%) **S** |
| P5 | 72.00% | 70.00% (-2.78%) | 71.33% (-0.93%) | 70.00% (-2.78%) |
| P10 | 68.67% | 69.33% (0.97%) | 69.67% (1.46%) | 69.33% (0.97%) |
| P100 | 24.23% | 24.20% (-0.14%) | 24.30% (0.28%) | 24.20% (-0.14%) |
| P500 | 5.95% | 6.01% (1.01%) **S** | 5.96% (0.22%) **S** | 6.01% (1.01%) **S** |

Table 9: Fuzzy model on ELDA collection $\lambda_1 = 0.999$, $\lambda 2 = 0.3$

| | | Dubois and Prade T-norm $\gamma = 0.005$ | | |
|---|---|---|---|---|
| | Hiemstra | Weber ($\gamma = -25$) | Weber-Sugeno ($\gamma = -200$) | Hamacher ($\gamma = 3$) |
| NIAP | 53.26% | 53.94% (1.29%) | 54.42% (2.18%) | 53.95% (1.31%) |
| IAP | 53.81% | 54.56% (1.40%) | 55.01% (2.24%) | 54.61% (1.48%) |
| Rprec | 52.68% | 54.21% (2.91%) | 54.60% (3.64%) | 54.21% (2.91%) |
| P5 | 72.00% | 74.00% (2.78%) | 73.33% (1.85%) **S** | 74.00% (2.78%) |
| P10 | 68.67% | 71.33% (3.88%) **S** | 72.67% (5.83%) **S** | 71.33% (3.88%) **S** |
| P100 | 24.23% | 23.70% (-2.20%) **S** | 23.77% (-1.93%) **S** | 23.70% (-2.20%) **S** |
| P500 | 5.95% | 5.91% (-0.67%) | 5.91% (-0.67%) | 5.91% (-0.67%) |

Table 10: Fuzzy model on ELDA collection $\lambda_1 = 0.999$, $\lambda 2 = 0.3$

Manipulating the weights ($\lambda_1$, $\lambda_2$) showed that we achieved an improvement in average precision on ELDA collection.

Nevertheless, whether we change($\lambda_1$, $\lambda_2$) or not, the t-conorm operators that gave similar precision to the interpolation smoothing were Weber, Weber-Sugeno and Hamacher. These three operators mixes the sum of two degrees with their product, the result is then normalized and it depends on the used parameter for each operator.

The change in precision between the original and the fuzzy model depends on the operators used in the fuzzy model and the used parameters and weights, sometimes the precision is higher only when comparing the first few retrieved documents, as in the case when using a Dubois and Prade t-norm and a Weber t-conorm with ($\lambda_1 = 0.999, \lambda_2 = 0.3$). While the improvement in change when using Hamacher or Einstein t-norm is not limited by the first few retrieved documents.

After trying different aggregating operators for $Aggreg$ and $Aggreg^{'}$ in (14), we saw that OWGA operators give good results a Aggreg; as $(score(q,d) = OWGA_{t\in q}(Aggreg^{'}(\lambda_1.\mu_d(t), \lambda 2.\mu_C(t))))$. Additionally, by using the OWGA operator, we can assign weights to terms' degrees after ordering these terms. These weights could be obtained by a mathematical function. This means that the degrees are smoothed again using a function like the exponential or polynomial or others. However, the results showed that choosing an exponential function would give higher NIAP than other functions, as it will assign higher weights to higher degrees. The polynomial function ($x^2$) gave good results on ELDA but not on INIST, this might be because INIST contains longer queries, and that the difference between weights assigned to terms using ($x^2$) increases too rapidly as the number of terms increases.

| | | OWGA | |
|---|---|---|---|
| | Hiemstra | Exp | Cube |
| NIAP | 53.26% | 54.76% (2.83%) **S** | 54.79% (2.89%) |
| IAP | 53.81% | 55.49% (3.12%) **S** | 55.29% (2.76%)**S** |
| Rprec | 52.68% | 54.53% (3.52%) | 56.13% (6.56%) |
| P5 | 72.00% | 71.33% (-0.93%) | 73.33% (1.85%) |
| P10 | 68.67% | 68.67% (0.00)% | 69.33% (0.97%) |
| P100 | 24.23% | 24.50% (1.10)% | 24.30% (0.28%) |
| P500 | 5.95% | 6.08% (2.24%) | 5.91% (-0.67%) |

Table 11: Fuzzy model on ELDA collection using OWGA

| | | OWGA | |
|---|---|---|---|
| | Hiemstra | Gaussian | InvGuassian |
| NIAP | 53.26% | 51.97% (-2.41%) | 52.98% (-0.50%) |
| IAP | 53.81% | 52.96% (-1.58%) | 53.88% (0.13%) |
| Rprec | 52.68% | 52.87% (0.36%) | 53.14% (0.88%) |
| P5 | 72.00% | 70.00% (-2.78%) | 70.00% (-2.78%) |
| P10 | 68.67% | 64.67% (-5.83%) | 66.00% (-3.88%) |
| P100 | 24.23% | 24.30% (0.28%) | 24.33% (0.41%) |
| P500 | 5.95% | 6.09% (2.4%7) **S** | 6.07% (2.13%) |

Table 12: Fuzzy model on ELDA collection using OWGA

We also thought that a big variance in terms' degrees should penalize the document's score. In other words, documents are given higher scores when the terms' degrees are of close values. For example, we take the following

| | | OWGA | |
|---|---|---|---|
| | Hiemstra | Exp | Cube |
| NIAP | 15.16% | 15.38% (1.49%) | 14.05% (-7.32%) |
| IAP | 12.68% | 17.42% (37.41%) **S** | 16.18% (27.63%) **S** |
| Rprec | 19.11% | 18.66% (-2.36%) | 16.90% (-11.55%) **S** |
| P5 | 42.67% | 43.33% (1.56%) | 41.33% (-3.12%) |
| P10 | 35.33% | 36.00% (1.89%) | 36.00% (1.89%) |
| P100 | 12.80% | 12.90% (0.78%) | 11.10% (-13.28%) **S** |
| P500 | 4.63% | 4.71% (1.73%) | 4.43% (-4.46%) |

Table 13: Fuzzy model on INIST collection using OWGA

| | | OWGA | |
|---|---|---|---|
| | Hiemstra | Gaussian | InvGuassian |
| NIAP | 15.16% | 15.20% (0.32%) | 15.21% (0.35%) |
| IAP | 12.68% | 17.23% (35.95%) | 17.24% (36.03%) |
| Rprec | 19.11% | 18.14% (-5.07%) | 18.14% (-5.07%) |
| P5 | 42.67% | 42.00% (-1.56%) | 42.00% (-1.56%) |
| P10 | 35.33% | 35.33% (0.00%) | 36.33% (2.83%) |
| P100 | 12.80% | 12.50% (-2.34%) | 12.67% (-1.04%) |
| P500 | 4.63% | 4.60% (-0.72%) | 4.62% (-0.29%) |

Table 14: Fuzzy model on INIST collection using OWGA

equation:

$$OWGA(x_1, x_2, ...x_n) = \prod_{j=1}^{n} (x_{\alpha(j)}^{w_j})^{(x_{\alpha(j)}^{w_j} - x_{\alpha(j+1)}^{w_{j+1}} + 0.5)} \tag{20}$$

We carried some experiments on this idea, table 15 shows the result obtained by applying the above formula. In fact, the new resulting degrees after using

| OWGA | | |
|---|---|---|
| | ELDA | INIST |
| NIAP | 54.84% (2.98%) **S** | 15.40% (1.60%) |
| IAP | 55.59% (3.30%) **S** | 17.44% (37.60%) **S** |
| Rprec | 54.80% (4.03%) | 18.53% (-3.06%) |
| P5 | 71.33% (-0.93%) | 43.33% (1.56%) |
| P10 | 68.67% (0.00%) | 36.33% (2.83%) |
| P100 | 24.83% (1.24%) | 12.80% (0.00%) |
| P500 | 6.08% (2.23%) **S** | 4.71% (1.73%) |

Table 15: Fuzzy model on ELDA and INIST collection using OWGA(20)

*Exp* function that assigns weights to terms' degrees, are more homogeneous.

So we thought of penalizing the document's score by its standard deviation. Table 16 contains the results.

| OWGA | | |
|---|---|---|
| | ELDA | INIST |
| NIAP | 55.22% (3.68%) **S** | 15.45% (2.00%) |
| IAP | 55.78% (3.67%) **S** | 17.56% (38.56%) **S** |
| Rprec | 56.22% (6.72%) **S** | 19.07% (-0.23%) |
| P5 | 74.00% (2.78%) | 42.67% (0.00%) |
| P10 | 68.00% (-0.97%) | 37.00% (4.72%) |
| P100 | 24.83% (2.48%) **S** | 12.97% (1.30%) |
| P500 | 5.97% (0.45%) | 4.85% (4.75%) |

Table 16: Fuzzy model on ELDA and INIST using OWGA & Standard devaition

### 4.5 Conclusion

We have extended Hiemstra's model in this section using fuzzy aggregation operators. We have also tested this representation using different aggregation operators, and we concluded that our fuzzy model gives similar or better results than Hiemstra's model, and that depends on the chosen aggregation operators. In the next section, we will extend this model section using semantically related terms.

## 5 Expanding Documents with Semantically Related Words

Starting from the model that we have proposed in (13), we wanted to smooth probabilities of terms again by taking into consideration the existence of semantically related terms with query terms in the document. In fact, a term $t$ in the query $(q = t_1, t_2, ...t_n)$ might not appear in the document, but a semantically related term of it might be present. The score that should be given to this term is not the same to the score having not seeing the semantically related term.

In addition, there exists two possible ways of smoothing degrees with semantically related terms. The first approach consists of expanding the query with semantically related terms. This would be done by adding a list of semantically related terms of each term into the query, re-weighting query terms, and then calculate the score of document given the expanded query, each term will be replaced by $(t \quad OR \quad s_1 \quad OR \quad s_2 \quad OR \quad ... \quad OR \quad s_n)$, where $(s_1, s_2, ..., s_n)$ are the semantically related terms of term $(t)$.

The other method is done by adding the semantically related terms of each term of the document, then by calculating the scores of expanded documents given a query. The difference between the two methods is that the first one will change the terms' weights in query while the second will re-weight the terms in documents.

In this section, we are going to describe the steps that we followed to accomplish the method of expanding documents by semantically related terms.

## 5.1 Extracting semantically related terms from Corpus

A database of French synonyms was not available. For that reason, we had to create our own list of semantically related couples of terms, in addition to the task of finding the values that represent the resemblance degrees between a term and each of its semantically related terms.

This work was done using statistical extraction measures, in which for each pair of words ($w_1$,$w_2$) in the corpus, we calculate four values (a,b,c,d), where:

- $a$ represents the number of times the two words have appeared together in the same document
- $b$ represents the number of times $w_1$ appeared together with another word than $w_2$
- $c$ represents the number of times $w_2$ appeared together with another word than $w_1$
- $d$ represents the number of appearance of all possible couples in the corpus in which neither of the two words is $w_1$ nor $w_2$.

After calculating these four values for each pair of words, we can apply one of the measures in [17] and get their corresponding resemblance degree. We repeat this operation for each word $w$ with each other word $w_i$ in the corpus, then finally choose to keep a specific number of related words that have the highest resemblance degrees with $w$. After trying different measures, we have seen that using Dice measure [17] gives a good list of semantically related terms:

$$Dice(w_1, w_2) = \frac{2a}{(a+b)+(a+c)} \tag{21}$$

## 5.2 Using Dilation to Compute Scores

After creating the list of semantically related terms and the associated degrees, we choose the number of terms that we want to keep for each term,

let it be $n$, we get for each term in the corpus a list of n most related terms $S(t)$. Then we have:

$$S(t) = \Big\{\Big(s_1, \mu(s_1, t)\Big), \Big(s_2, \mu(s_2, t)\Big), ..., \Big(s_n, \mu(s_n, t)\Big)\Big\} \tag{22}$$

Where $\mu(s_i, t)$ is the resemblance degree between a term and its semantically related term, for example, a list of the 2 most related terms for the term *paper* is:

$S = \{(article, 0.05), (report, 0.001)\}$

Notice that we consider a term as semantically related to itself with a resemblance degree of 1.

The work done in [2] proposes a way to find a term's degree when considering the existence of its synonyms. We were able to calculate the new score of a term using the formula in [2], called smoothing by dilation:

$$\mu_d(t) = \max_{s_i \in S}(\top(\mu_d(s_i), \mu(s_i, t))) \tag{23}$$

The membership degree of term $t$ in document $\mu_d(t)$, is calculated firstly by taking each one of its related terms $s_i$, and aggregating the resemblance degree of $\mu(s_i, t)$ with the value of aggregation between the membership degree of $s_i$, $\mu_d(s_i)$, in document and membership degree of $s_i$, $\mu_C(s_i)$, in corpus, and secondly by choosing the maximum of all values, resulting after applying the first aggregation on each semantically related term, as the new membership degree of term $t$.

We have noticed that this formula can be seen as a weighted-maximum aggregation operator, where $\mu(s_i, t)$ are the weights and $\mu_d(t)$ are the scores. And as we mentioned in section 3, weighted-maximum aggregation operator is a particular case of the Sugeno discrete integral. For that, we aim to explore the use of other discrete integrals (ex. Choquet) as an aggretion operator instead of the Sugeno integral.

Indeed, the t-norm aggregation function in (23) is used to aggregate the membership of semantically related term in document and its resemblance to the term, this will give us a representative degree for each term. But then, the max operator allows us to choose only one of these degrees. Therefore, we thought of choosing an aggregation operator that would permit all semantically related terms to contribute to the final score, so we replaced max by a t-conorm. We now have:

$$\mu_d(t) = \perp_{s_i \in S}(\top(\mu_d(s_i), \mu(s_i, t))) \tag{24}$$

If we replaced $(\mu_d(s_i))$ by interpolation smoothing in fuzzy model (13) we get:

$$\mu_d(t) = \top_{s_i \in S}(\top\Big(\perp(\mu_d(s_i), \mu_C(s_i)), \mu(s_i, t)\Big)) \tag{25}$$

### 5.3 Example

Let us take an example to better understand the process of expanding a document with semantically related terms. Consider that corpus $c$ is represented by a set of (term $t$, membership degree $\mu_C(t)$), as follows:

$C = \{(note, 0.009), (report, 0.007), (article, 0.0011), (mark, 0.0003), (paper, 0.0034)...\}$

A document $d$ is also represented, as a set of couples ($t$, $\mu_d(t)$):

$d = \{(note, 0.01), (report, 0.002), (article, 0.004), (mark, 0.0005), ...\}$

We have a query:
$q = \{note, paper\}$

And a list of triples ($t$, $s$, $\mu(t, s)$):

$LS = \{(note, mark, 0.03), (paper, article, 0.05), (paper, report, 0.001)\}$

If we use formula in (13), we get $\mu_d('paper') = \top(0, 0.0034)$. Taking $\top = Hamacher$,$\mu_d('paper') = 0.0034$, this is its membership in corpus, even though its potential semantically related terms 'report' and 'article' do exist in the document. However, if we apply formula (25) we get:

$\mu_d('paper') = \top(\perp(0.0034, 1), \perp(0.00127, 0.05), \perp(0.00223, 0.001))$.
If we take Product as t-norm and Weber as t-conorm with a parameter=3 for each, we will get $\mu_d('paper') = 0.00085$.
The degree of *paper* is more expressive since the semantically related terms contribute to it, because other documents that do not contain the term or its semantically related terms will get much smaller degree. This is because the membership degree of term *paper* will be aggregated with the degrees of its semantically related terms while calculating this degree for each document in the corpus.

### 5.4 Results

We have done some experiments using the model in (26), as we tried to variate the use of aggregation operators, tables 17, 18, 19, 20 show the results made both on ELDA and INIST collections.

Adding semantically related terms of terms into documents would not only change the degrees of unseen query terms, but also the degrees of seen terms. This fact is reflected in the results, where we see that the precision has decreased concerning the first retrieved documents, i.e. the change of precision is of a higher value when we consider a bigger number of retrieved documents. From this we conclude, that degrees of seen terms should not

| | Hiemstra | Hamacher T-norm $\gamma = 0.8$ , Weber T-conorm $\gamma = -25$ | OWGA (Exp) |
|---|---|---|---|
| | | Hamacher(0.99) Weber($\gamma = 3$) | Hamacher(0.99) Weber($\gamma = 3$) |
| NIAP | 53.26% | 53.70% (0.82%) | 54.76% (2.82%) **S** |
| IAP | 53.81% | 54.09% (0.52%) | 55.52% (3.17%) **S** |
| Rprec | 52.68% | 53.32% (1.22%) | 54.94% (4.29%) |
| P5 | 72.00% | 72.00% (0.00%) | 71.33% (-0.93%) |
| P10 | 68.67% | 68.33% (-0.49%) | 68.67% (0.00%) |
| P100 | 24.23% | 24.50% (1.10%) | 24.53% (1.24%) |
| P500 | 5.95% | 6.24% (4.93%) | 6.08% (2.24%) |

Table 17: Results on ELDA using 5 semantically related words

| | Hiemstra | Hamacher T-norm $\gamma = 0.8$ Weber T-conorm $\gamma = -25$ | OWGA (Exp) |
|---|---|---|---|
| | | Hamacher(0.99), Weber($\gamma = 3$) | Hamacher(0.99), Weber($\gamma = 3$) |
| NIAP | 53.26% | 54.52% (2.37%) **S** | 54.72% (2.75%) **S** |
| IAP | 53.81% | 54.92% (2.07%) | 55.46% (3.08%) **S** |
| Rprec | 52.68% | 53.60% (1.74%) | 54.53% (3.52%) |
| P5 | 72.00% | 71.33% (-0.93%) | 71.33% (-0.93%) |
| P10 | 68.67% | 67.33% (-1.94%) | 68.67% (0.00%) |
| P100 | 24.23% | 24.73% (2.06%) **S** | 24.50% (1.10%) **S** |
| P500 | 5.95% | 6.39% (7.51%) **S** | 6.08% (2.24%) **S** |

Table 18: Results on ELDA using 10 semantically related words

| | Hiemstra | Hamacher T-norm $\gamma = 0.99$ Weber T-conorm $\gamma = 3$ | OWGA (Exp) |
|---|---|---|---|
| | | Hamacher(0.99), Weber($\gamma = 3$) | Hamacher(0.99), Weber($\gamma = 3$) |
| NIAP | 15.16% | 17.16% (13.25%) | 16.93% (11.71%) |
| IAP | 12.68% | 13.94% (10.01%) **S** | 18.88% (48.96%) **S** |
| Rprec | 19.11% | 20.98% (9.76%) | 20.78 (8.71%) |
| P5 | 42.67% | 38.00% (-10.94%) **S** | 37.33 (-12.50%) **S** |
| P10 | 35.33% | 32.33% (-8.49%) | 33.00 (-6.00%) |
| P100 | 12.80% | 15.03% (17.45%) **S** | 14.77 (15.36%) |
| P500 | 4.63% | 5.37% (15.83%) **S** | 5.37 (15.97%) **S** |

Table 19: Results on INIST using 5 semantically related words

be changed, this is left to be modeled in future work.

Using 10 semantically related words did increase the average precision more than the use of only 5 semantically related words when using t-norm and t-conorm to aggregate degrees. However, the improvement on INIST

| | Hiemstra | Hamacher T-norm $\gamma = 0.99$ Weber T-conorm $\gamma = 3$ | OWGA (Exp) |
|---|---|---|---|
| | | Hamacher(0.99), Weber($\gamma = 3$) | Hamacher(0.99), Weber($\gamma = 3$) |
| NIAP | 15.16% | 17.63% (16.34%) **S** | 17.01% (12.26%) |
| IAP | 12.68% | 14.56% (14.87%) **S** | 18.85% (48.71%) **S** |
| Rprec | 19.11% | 20.38% (6.62%) | 20.02% (4.75%) |
| P5 | 42.67% | 39.33% (-7.81%) | 38.00% (-10.94%) |
| P10 | 35.33% | 34.67% (-1.89%) | 33.33% (-5.66%) |
| P100 | 12.80% | 14.87% (16.15%) **S** | 14.10% (10.16%) |
| P500 | 4.63% | 5.48% (18.27%) **S** | 5.45% (17.70%) **S** |

Table 20: Results on INIST using 10 semantically related words

collection is more obvious than on ELDA, the reason is that INIST contains a much bigger number of documents, so that the extracted semantically related words are more representative than those of ELDA's.

We tried to extend the list of semantically related words used for ELDA by using the documents of INIST along with ELDA's to extract these words. The results showed that we could obtain better NIAP when using expressive semantically related words.

| | Hiemstra | Hamacher T-norm $\gamma = 0.8$ Weber T-conorm $\gamma = -25$ | OWGA (Exp) |
|---|---|---|---|
| | | Hamacher(0.99), Weber($\gamma = 3$) | Hamacher(0.99), Weber($\gamma = 3$) |
| NIAP | 53.26% | 54.34% (2.04%) | 54.89% (3.07%) |
| IAP | 53.81% | 54.83% (1.89%) **S** | 55.58% (3.29%) |
| Rprec | 52.68% | 53.50% (1.56%) | 55.27% (4.92%) |
| P5 | 72.00% | 72.00% (0.00%) | 72.00% (0.00%) |
| P10 | 68.67% | 69.33% (0.97%) | 69.00% (0.49%) |
| P100 | 24.23% | 24.73% (2.61%) | 24.53% (1.24%) |
| P500 | 5.95% | 6.33% (6.50%) **S** | 6.08% (2.24%) |

Table 21: Results on ELDA extended with INIST and using 5 semantically related words

# 6 Possibility theory in IR

Recently, we have thought of using another fuzzy logic mechanism to represent the IR process. This mechanism is based on the possibility theory. Here, we are going to present the concept of possibility theory, and then we will present our work done on using possibility theory measures in IR.

## 6.1 Possibility Theory

The possibility theory [11] is a mathematical theory for handling uncertainty (i.e. values which are not precisely known), and is an alternative to probability theory. While probability theory uses a single number which is the probability in order to describe how likely an event is to occur, possibility theory considers two basic values, the possibility $\Pi$ and the necessity $N$. If one assumes that $\pi$ is a possibility distribution on set $A$. The possibility of a set $A$ would be estimated as follows:

$$\Pi(A) = max_{x \in A}(\pi(x)) \tag{26}$$

A possibility equal to 0 indicates the event is impossible, and consequently the necessity is also 0. While a possibility equal to 1 means that the event is totally unsurprising to happen, but it leaves its necessity unconstrained. A necessity degree equal to one means that the event will happen. Its possibility is equal to 1 and the possibility of other events is 0, as:

$$N(x) = 1 - \Pi(x^c) \tag{27}$$

Where $x^c$ is the complement of $x$. When possibility is 1 and necessity is 0, it implies a total ignorance. Which means that the event is totally possible but not necessary at all.

Moreover, guaranteed possibility was defined by Dubois and Prade in [16]:

$$\Delta(A) = min_{x \in A}(\pi(x)) \tag{28}$$

According to [16], guaranteed possibility estimates to what extent all elements of $A$ are actually possible.

## 6.2 Using Possibility and Necessity Measures in IR

As we already mentioned in section 4, a document and a query could be represented as fuzzy sets where terms are their elements.

In the problem of IR, we want to know the possibility that a document has generated a given query:

$$\Pi(d|q) = \frac{\Pi(d \cap q)}{\Pi(q)} \tag{29}$$

However, $\Pi(q)$, is constant and then does not affect the ranking, so we can ignore it.

$$\Pi(d|q) \propto \Pi(d \cap q) \tag{30}$$

According to the definition in (26), we can write:

$$\Pi(d|q) \propto \Pi(d \cap q) = \max_{t \in d \cap q}(\pi(t)) \tag{31}$$

The necessity of a document given a query is:

$$N(d|q) = 1 - \Pi(d^c|q) = 1 - \frac{\Pi(d^c \cap q)}{\Pi(q)} \tag{32}$$

Again, $\Pi(q)$ does not affect the ranking:

$$N(d|q) \propto 1 - \Pi(d^c \cap q) = \max_{t \in d^c \cap q} (\pi(t)) \tag{33}$$

However, using the max operator allows one term to contribute to the final measure. For that, we replace the $max$ operator with a t-conorm. We get:

$$\Pi(d|q) \propto \Pi(d \cap q) \propto \perp_{t \in d \cap q} (\pi(t)) \tag{34}$$

$$N(d|q) \propto 1 - \Pi(d^c \cap q) \propto 1 - \perp_{t \in d^c \cap q} (\pi(t)) \tag{35}$$

Using DeMorgan Law [11], we write:

$$N(d|q) \propto 1 - \Pi(d^c \cap q) \propto \top_{t \in d^c \cap q} (1 - \pi(t)) \tag{36}$$

Where $\top$ is the dual of $\perp$ in (35) w.r.t the fuzzy negation operator.

We can also calculate the guaranteed possibility of the document given a query:

$$\Delta(d|q) \propto \Pi(d \cap q) = \min_{t \in d \cap q} (\pi(t)) \tag{37}$$

Replacing the $min$ with a t-norm we get:

$$\Delta(d|q) \propto \top_{t \in d \cap q} (\pi(t)) \tag{38}$$

The problem is that any $t$ is totally possible in $d^c$, resulting with a necessity of zero. Here we need to use a smooth the possibilities by not assigning a possibility of 1 to any term.

However, using formula (34) to aggregate possibilities of seen terms did not give good results, because as we have showed in section 4, the only operators that perform well are those who give results close to the result produced by the product. Now we are going to see how (35) and (38) can be related to the language modeling approach to IR.

Indeed, Pont and Croft model as well as Hiemstra's can be seen as the following form:

$$score(q, d) = \prod_{t \in q \cap d} (P_s(t|d)) \times \prod_{t \in q \cap d^c} (P_s(t|d)) \tag{39}$$

Where $P_s(t|d)$ in $(\prod_{t \in q \cap d}(P_s(t|d)))$ is estimated as follows, (see section 3):

$$P_s(t|d) = P(t|d)^{(1-\hat{R}_{t,d})} \times P_{avg}(t)^{\hat{R_{t,d}}} \quad PonteandCroft \tag{40}$$

$$= \lambda_1 P(t|d) + \lambda_2 P(t|C) \quad Hiemstra \tag{41}$$

$$\tag{42}$$

While $P_s(t|d)$ in $(\prod_{t \in q \cap d^c}(P_s(t|d)))$ is:

$$P_s(t|d) = \frac{cf_t}{cs} \quad PonteandCroft \tag{43}$$

$$= \lambda_1 P(t|d) + \lambda_2 P(t|C) \quad Hiemstra \tag{44}$$

$$\tag{45}$$

Taking formula (37), we replace the product by a t-norm, and the probabilities by a possibility distribution $\pi$, we have:

$$score(q, d) = \top(\top_{t \in q \cap d}(\pi(t)), \top_{t \in q \cap d^c}(\pi(t))) \tag{46}$$

The score formula (42) is obtained an aggregation between (33) and (36).

So we conclude that we could have a model based on the the measures of possibility theory, as they have a common link with language models approach to IR. The obvious difference is the way of calculating the possibility distribution $\pi$ and choosing a mechanism of smoothing for possibilities. In fact, we think that this track should be investigated since smoothing possibilities could be easier than this of smoothing probabilities, as they do not have the constraint of summing into one.

# 7 Conclusion

Language modeling approaches have been applied to information retrieval with success. Moreover, fuzzy logic concepts have been introduced to IR [2, 13, 19]. One of these approaches was the graded-inclusion IR model proposed in [13], the model has proven efficiency in imitating and generalizing the vector space model (VSM). To our knowledge, experiments of extending a language model using fuzzy logic notions have not yet been studied. From here, we wanted to explore the possibility of extending language models approaches for IR using fuzzy logic operators and concepts.

In this report, we explored and tested some language models approaches for IR. We have represented in section (3) a language modeling approach for IR using fuzzy logic concepts. The experiments made on this fuzzy model showed that it could give similar or better performance compared with a classical language modeling approach when the right parameters and operators are used.

Query expansion by semantically related words have been used to improve the performance of IR systems. We have explored in section (4) this track by expanding the documents with synonyms using the proposed model in [2], but trying to variate the use of fuzzy operators. Results showed big improvement change on INIST collection.

Finally, we studied the two measures of possibility and necessity introduced in possibility theory, and we think that these measures have a link with language models approaches to IR. We aim to study this track using strong theoretical foundations.

In future work, we envision an investigation on the use of non symmetrical fuzzy operators for the proposed fuzzy model. We also intend to explore other discrete integral operators than the Sugeno's integral used in formula (24), (ex. Choquet integral), as an aggregating function. Aditionally, we think of improving the mechanism of expanding documents by

# 8 References

[1] Introduction to Information Retrieval. Christopher D. Manning, Prabhakar Raghavan and Hinrich Schtze. Cambridge University Press. 2008.
[2] On the use of tolerant graded inclusions in information retrieval. Patrick Bosc and Olivier Pivert. In Proceedings of CORIA'08, pages 321-336. 2008.
[3] Information Retrieval with fuzzy logic. Rima Harastani. Bibliographical report of Internship M2RI IRISA. 2010.
[4] Twenty-One at TREC-7: ad-hoc and cross-language track. Djoerd Hiemstra and Wessel Kraaij. Proceedings of the seventh Text Retrieval Conference TREC-7, NIST Special Publication 500-242, pages 227-238. 1999.
[5] Modèles de langue pour la recherche d'information. Jian-Yun Nie. Université de Montréal. 2004.
[6] Statistical Language Modeling For Information Retrieval. Xiaoyong Liu and W. Bruce Croft. University of Massachusetts. 2003.
[7] A Study of Smoothing Methods for Language Models Applied to Ad Hoc Information Retrieval. Chengxiang Zhai and John Lafferty. ACM Transactions on Information Systems TOIS, pages 179-214. 2004.
[8] A Language Modeling Approach to Information Retrieval. Jay M. Ponte and W. Bruce Croft. In Proceedings of SIGIR'98, pages 275-281. 1998.
[9] An Empirical Study of Smoothing Techniques for Language Modeling. Stanley F. Chen and Joshua Goodman. 1998.
[10] A General Language Model for Information Retrieval. Fei Song and W. Bruce Croft. In Proceedings of the 1999 ACM SIGIR Conference on Research and Development in Information Retrieval, pages 279-280. 1999.
[11] La Logique Floue Bernadette bouchon-meunier. Presse universitaires de France. 'Que sais-je?' no 2702. 1993.
[12] The three semantics of fuzzy sets. Didier Dubois, Henri Prade. Fuzzy

sets and systems 91 ,141-150. 1997.
[13] Implication-Based and Cardinality-Based Inclusions in Information Retrieval. Patrick Bosc, Laurent Ughetto, Olivier Pivert, and Vincent Claveau. In proceeding of FUZZ-IEEE, pages 2088-2093. 2009.
[14] Fundametals on Aggregation Operators. Marcin Detyniecki. Manuscript, Berkeley initiative in Soft Computing. 2001.
[15] Importance weighted OWA aggregation of multicriteria queries. Henrik Legind Larsen. In Proceeding of the North American Fuzzy Information Processing Society conference NAFIPS'99, pages 740-744. 1999.
[16] Knowledge-Driven versus Data-Driven Logics. Didier Dubois, Petr Hajek and Henri Prade. Journal of Logic, Language and Information, pages 65-89. 2004.
[17] Acquisition automatique de lexiques semantiques pour la recherche d'information. Vincent Claveau. PHD thesis. 2003.
[18] On Sugeno integral as an aggregation function. Jean-Luc Marichal. Fuzzy Sets and Systems, pages 347-365. 2000.
[19] Possibilistic networks for information retrieval. Mohamed Boughanem, Asma Brini, Didier Dubois International journal of approximate reasoning 50(7): 957-968. 2009.
[20] The Philosophy of Information Retrieval Evaluation. In Proceedings of the The $2^{nd}$ Workshop of the Cross-Language Evaluation Forum on Evaluation of Cross-Language Information Retrieval Systems, pages 355-370. 2001.