

Organizing Web Documents into Thematic Subsets Using an Ontology

B.Nguyen
INRIA-FUTURS
Domaine de Voluceau
78153 Le Chesnay CEDEX FRANCE
(33) 1 39 63 51 88
Benjamin.Nguyen@inria.fr

M.Vazirgianis, I.Varlamis, M.Halkidi
Athens University of Economics and Business
Computer Science Department
76 Patision Street
Athens, 10434, GREECE
(301) 8203 519

{mvazirg, varlamis, mhalk}@aueb.gr

Abstract

The volume and diversity of documents in the WWW, and the fact that these documents are connected to each other by links that bear important semantics generate requirements for effective organization of web content. This would enable effective browsing and searching. Both tasks are based on similarity between documents at the semantic level. In this context we propose a novel similarity measure between web documents, characterized by their incoming links, that can be used in conjunction with a clustering algorithm to discover groups of web documents on the same topic. Assuming each document is characterized by a sets of terms of an ontology, the similarity measure can compute a collective similarity between sets of terms seen as a whole, rather than between terms that would be independent. The similarity measure is embedded in different clustering algorithms and is extensively evaluated against real web document collections classified by human experts, in order to prove its efficiency.

1. INTRODUCTION

“A particular word is like the center of a constellation; it is the point of convergence of an indefinite number of co-ordinated terms.”

Course in general Linguistics, Ferdinand Saussure

The Web Today

Nowadays, the World Wide Web contains far more data than anyone could hope reading in their whole lifetime; it has now become the main source of information many people use. Web Search Engines [1],[2] have contributed to this tremendous success, by delving through this mountain of data to provide Web Surfers with the answers they are looking for. Nonetheless, when once upon a time a query with a few keywords was sufficient to return pages of interest that a user could browse through, it is becoming increasingly difficult to maintain such efficiency, given the growing size of the web, and the length of the list of answers that queries tend to return. The results presented to users should preferably be grouped into meaningful subsets that can be easily browsed. This has lead to the flourishing of *meta-search* engines, that use the data outputted from various search engines and try to organize results better: Vivissimo [3] queries a search

engine, and clusters the outputted snippets using hierarchic document clustering and labeling techniques. Kartoo [4] presents the results of search engines in a graphical display showing the links between groups of web pages.

Enhancing the semantics of Web pages

Other approaches involve trying to characterize the pages themselves better, using the fact that the Web is a graph, and by taking into account its link structure. Clustering documents using the incoming links of a page has been proposed by [5]. The results of the system proposed by Haveliwala *et.al.* are good, however the characterization of Web pages simply uses the words contained in the neighborhood of the link that leads to it, and involves no further processing. In this article we investigate new techniques in order to access the *semantics* of the links, rather than simply the keywords, and we do it automatically.

Context of our problem

The general context is that of a user looking to construct a data warehouse on a specific domain, by using the information found on the Web. For a complete outlook on the problem, we refer the reader to [6], in the following paragraph we detail the core ideas of the system and the assumptions we hold for this article.

We suppose that this user is a “specialist” of the domain, and has at his disposal (or has constructed) an *ontology* (see definition further down in this section) describing the domain. In [6] we propose the architecture of a system called THESUS, that lets a user construct a data warehouse on the topic described by this ontology, using the following modules: thematic crawler, information extraction module, semantics enhancer, classifier, query engine. In this article, we focus on the classifier and provide a short description of the other modules. It is important to note that the input of the classifier is a set of documents, that have attached to them a set of weighted *terms* of the domain ontology. We base most of our reasoning in the article on the assumption that **we are provided with such a set of terms**. For more details on how this set is constructed, which is beyond the scope of this paper, we refer to [7] and [6]. We define the goal of this article as the following: *Given documents that are characterized by a*

(short) set of weighted terms from an ontology, find a way of clustering related documents together. In this article, we propose a clustering scheme, based on a novel similarity measure between sets of terms that are hierarchically related.

Traditionally, in order to achieve this goal, such a user would apply Information Retrieval techniques such as those described in [8]. However, these techniques most often rely on **exact** keyword matching, and do not take into account the fact that the keywords may have similar meanings, some *semantic proximity* between each other. Let us stress that we are running the clustering on the sets of terms that describe the document, and that this list can be quite short. For instance a document might be characterized by the words “cat, food” and another with the word “feline, menu”. By using traditional techniques these documents would be judged unrelated, however we will show that by using a distance between terms in an ontology, we are able to compute a meaningful similarity measure. Another advantage of representing the documents using terms of an ontology instead of using keywords extracted from the page, even when using exact matching techniques, such as the vector space model, is that we reduce the dimensions of the space. This reduction is achieved not by removing keywords from the documents’ description but by abstracting them to terms in the ontology.

In [6], we investigate how to construct a whole system that could cluster related web pages together, using novel techniques based on the use of a thesaurus, *WordNet* [9] and a pre-made ontology of the thematic domain that the set of web pages belongs to. In this article, we review the more specific and novel problem of clustering documents that are characterized by a set of weighted terms of an ontology, thus **we assume given** i. the initial set of web pages, ii. the domain ontology. Our view of each page in our data set is simply a set of weighted terms of the domain ontology. We detail in [7], [6] how these terms are extracted, using WordNet.

Contributions of the article

- 1- A novel similarity measure between sets of terms belonging to an ontology.
- 2- The application of this measure to a modified version of the DBSCAN [10], [11] clustering algorithm, providing semantic clustering of web documents, and to the COBWEB algorithm [12].
- 3- Testing, and proof of the quality of the results, using external clustering quality measures and a document set classified by human experts as a reference.

Notations/Assumptions

Ontology: We define an *ontology* as a IS-A tree, but all the results in this paper can be extended to a tree with any sort of “relation” between a parent and child node, given the fact that this relation indicates that parent and child node are “close” to each other from a semantic

point of view. We call a word in the ontology a **term** of the ontology. Most results are also extendable if the *ontology* is a direct acyclic graph, we simply cannot use the Wu and Palmer measure as such and need to change it slightly.

Note : this is a “weaker” definition of an ontology than the ones traditionally proposed in the Semantic Web community, that usually take into account multiple ancestors. The results in this paper can directly be applied in the case of multiple ancestor hierarchies, the time to pre-process the equivalent of the Wu and Palmer similarity is slightly increased, since when trying to evaluate the first common ancestor of two nodes, we may need to explore all ancestor hierarchies for each node. However, since this process is done offline, this does not cause a reduction in performance of our algorithm. We do not have the room here to detail the specific adjustments that need to be made to the similarity measure in this case.

Document: In this article, we assume that a document d_n is a web document, found at URL_n . It can be abstracted by an identifier (URL_n) and a finite set of weighted terms of the ontology. We note $d_n = \{URL_n, (w_1, k_1); (w_2, k_2); \dots\}$. For simplicity we will drop the URL_n and simply talk about “document n ”.

In Section 2, we will present the related work. In Section 3, we will present an overview of the whole system. In Section 4, we will detail our new similarity measure. In Section 5, we show the experimental results of our algorithm. Section 6 is the conclusion.

2. RELATED WORK

2.1 Similarity Measures

2.1.1 General ideas

We are faced with the task of defining a similarity measure between **documents**. In this article, we assume this boils down to finding the similarity between **sets of weighted terms of an ontology** (or hierarchy).

2.1.2 Existing similarity measures between sets

A number of similarities/distances between sets of elements already exist in the literature [13], [14]. The most widely used one, the Jaccard coefficient is simple: Let A and B represent two sets of elements. The similarity between A and B is defined by:

$$\text{Jaccard similarity: } S_1(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

Although this similarity respects the three intuitions given in Section 4.1, it only takes into account exact matches. Yet in our situation we define a measure that takes into account the proximity of words, when comparing sets of words, rather than a simplistic binary comparison.

In [13] the problem of measuring the similarity or distance between two finite sets of points in a metric space is considered. Some of the distance functions are

reviewed. All of them take polynomial time algorithms for their computation. In order to be competitive, our measure should be tractable in polynomial time.

In [14] the idea of calculating similarities between sets, using the Jaccard coefficient, is investigated. The indexing issue for distance / similarity between sets of values is treated in recent work [15], again using the Jaccard coefficient to calculate the similarity. [16] also investigate this with their mediator approach.

The traditional cosine measure from the Information Retrieval literature (see [8]) can be viewed as a direct application of the Jaccard coefficient.

However, in all the aforementioned efforts, all values are independent from each other, therefore only exact matching between values is taken into account. With our approach –defining a document by terms of the ontology, and calculating the similarity between these sets of terms– documents defined by different sets of words may end up having a very high similarity rating. For instance, recall the example given in the introduction, two documents defined by (“cat”, ”food”) and (“feline”, “menu”) would have a similarity value different from 0, which is its similarity value according to the Jaccard coefficient. In our case, we want to take into account the proximity of terms given that they are part of a hierarchy, in our case an ontology.

2.1.3 Similarity between two elements of an ontology

In order to compute the distance between sets of terms of an ontology, we were lead to investigate the existing similarity measures in the more simple case of calculating the similarity between two given terms of the ontology. [17], [18] and [19] propose different measures of similarity between terms in a hierarchy such as

WordNet, and [20] proposes a comparison between these measures and others, such as Wu and Palmer [21], Miller and Charles, and a novel similarity measure. [22] also propose the use of Wu and Palmer measure in the ontology context. The Wu and Palmer measure is the fastest to compute, and is arguably as good as the others (see [20]), which is why we chose to use it. We detail its definition and properties in the following paragraph.

Wu and Palmer similarity measure

Given a tree, and two nodes a,b of this tree, we first of all find their deepest (in terms of depth in the tree) common ancestor c. The



Figure 1: Wu and Palmer Similarity

case is similar whether the nodes have single or multiple ancestors. The similarity measure is computed as follows:

$$S_{W\&P}(a,b) = \frac{2 \cdot \text{Depth}(c)}{\text{Depth}(a) + \text{Depth}(b)}$$

For example, in Figure 1: $S_{W\&P}(a,b) = \frac{2 \times 2}{4 + 3} = 0.57$

Note that we consider the first node to be of depth 1 and not 0. As a side note, we can define a distance using this similarity: the similarity between words is a real number that ranges from 0 to 1. We can define the following distance measure on this similarity in the following way:

$$D(a,b) = 1 - S_{W\&P}(a,b)$$

It is trivial to see that this distance verifies:

$$D(x,y) = 0 \text{ iff } x=y \text{ and } D(x,y) = D(y,x)$$

It is straightforward but lengthy to show that this distance is a metric, i.e., that it verifies the triangular inequality:

$$D(x,y) \leq D(x,z) + D(z,y)$$

A similarity measure only needs to verify the first two conditions (except that the first is replaced by $S(x,y) = 1$ iff $x=y$). The clustering algorithms we use are based on the similarity between documents and not on their position in a metric space, so we only need a measure and not a metric. In the rest of this article we assume Wu & Palmer as a similarity measure, and not a metric.

2.2 Document Clustering

There has been large interest in this problem in the Information Retrieval community [12], [23]. Our problem is more specific since we are clustering Web documents [24], and take *links* into account [25]. Since we are working with categorical data we need a distance or similarity measure and a density based algorithm. We have implemented the COBWEB algorithm [12] for categorical data and a variation of the DBSCAN algorithm [11] using our similarity measure.

2.3 Related Systems

- Kartoo [4] : a meta search engine that exploits links
- Vivissimo [3] : a clustering engine that uses the output of a search engine.
- Haveliwala *et al.* [5] also propose a methodology for evaluating strategies for similarity search on the Web. According to this approach a Web document, u , is represented by the following bag

$$B_u = \{(w_u^1, f_u^1), (w_u^2, f_u^2), \dots, (w_u^k, f_u^k)\}$$

where w_u^i are terms used in representing u and f_u^i are their corresponding weights.

Then the similarity between documents is measured as the similarity between their bags. The metric used for

computing the similarity of documents is the *Jaccard coefficient*.

- A method for classifying and describing Web documents is discussed in [26]. They use inbound links and words surrounding them to describe the Web pages. SVM classifier is trained and used to categorize Web pages. Also a method for selecting features and characterizing the classes of Web pages is proposed that uses the expected entropy loss metric. The results of the proposed approach show that the text in citing documents has great descriptive and descriptive power than the text in the target document itself.

- Systems using a thesaurus: Using a thesaurus to improve results or expand queries is a common idea. There are many articles that deal with this problem, such as [27].

- Systems that classify web documents into predefined categories: [28] proposes a system for automatic classification of web documents into predefined categories using a training set of pre-classified documents. The documents are represented using word frequency vectors.

3. A BRIEF OVERVIEW OF THE THESUS SYSTEM

We refer to [6], [7] for a longer version of this paper and a more detailed description of the THESUS system. In this paragraph, we briefly explain the operations we will not detail in this article. We will explain how we construct the ontology, and how we get the set of documents from the web.

3.1 Ontology creation

In order to provide semantic clustering functions, we need to refer to an ontology of terms that are relevant to our domain of interest. The ontology used is based on DMOZ directory [29], manually trimmed and rearranged, in order to form a hierarchical taxonomy of terms. Examples of ontologies that can be used, are those suggested by The DARPA Agent Markup Language Program [36] (i.e. ontology on music <http://www.daml.org/ontologies/276>). In order for our system to work, we need to also add a mapping from each term of the ontology to a set of senses (synset) in WordNet.

3.2 Gathering relevant Web documents

We start off by creating a collection of Web documents that relate to the ontology. The acquisition process starts from an initial core of documents and expands it with some of the documents that they point to. The documents are selected only if the links that point to them carry specific semantics (i.e., the words in the incoming links can be mapped with high trust to terms in our ontology, as shown in the following paragraph). This happens recursively for a number of times or until a number of documents defined by the user are collected. The initial core contains well-known web directory pages such as

DMOZ, Yahoo or Google Directory. We do not argue here the importance of using incoming links in order to find concise words that describe documents, and we refer to [6] for a complementary discussion on the subject.

3.3 Mapping keyword sets to ontology-term sets

As we mentioned in the introduction the clustering algorithm runs on documents described by a set of terms from the ontology. Usually keyword extraction processes on documents generate for each document d_i a set of keywords $\{k_i\}$ with a respective set of weights. In THESUS the weights represent the number of documents that point to d_i using a particular keyword. So each document d_i has the following description: $d_i = (\text{URL}, \{(k_j, n_j)\})$ where n_j represents the number of documents that point to d_i using keyword k_j . It is widely accepted that the keyword importance increases proportionally to n_j [1].

For a verb or noun WordNet provides a *synset*, a set of different senses that the verb or noun may have. Verbs and nouns senses, are organized as topical hierarchies forming a “forest” of 25 trees (having the IS-A relationship) of nouns and 15 of verbs. For adjectives and adverbs WordNet provides synonyms. For each keyword in WordNet we can have a set of senses and in the case of nouns and verbs, a generalization path from each sense to the root sense of the hierarchy.

We map each keyword k , ($k \in \{k_j\}$, where $\{k_j\}$ is the set of keywords describing a document), to the ontology term t_i , when k and t_i have a pair of senses (k^x, t_i^y) that gives the maximum Wu & Palmer similarity among all pairs of senses of the keyword and the ontology terms. Mapping is refined by removing the senses of k that do not relate to the keywords of $\{k_j\}$.

To provide an example, for the words *guitar*, *flute* and *wind*, WordNet provides one, three and eight senses respectively. For all the 24 triplets of senses the average Wu-Palmer similarity is computed in WordNet’s hierarchies. The self-correlation of the keyword set (*guitar*, *flute*, *wind*) gives a score of 0.8 to the triplet of meanings (*guitar*, *flute/transverse flute*, *wind instrument/wind*) and less than 0.5 to any other combination. Similarly the self-correlation of the set (*storm*, *cloud*, *wind*) gives a score of 0.8 to the triplet of meanings (*storm/violent storm*, *cloud*, *wind/air moving*) and lower scores to any other combination. These are indications that “*wind*” has the sense of “*wind instrument*” when it appears to the set (*guitar*, *flute*, *wind*), whereas it has the sense of “*wind as weather phenomenon*” in the set (*storm*, *cloud*, *wind*).

Based on the remaining senses, each keyword in $\{k_j\}$ is mapped to the closest ontology term as previous. In the previous example the set (*guitar*, *flute*, *wind*) will be mapped to the set of ontology terms (*string instrument*, *wind instrument*, *wind instrument*), whereas the set (*storm*, *cloud*, *wind*) will be mapped to an empty set,

when the “*music*” ontology is used. As a result each k_j is mapped to a term t_i with a similarity s_j . It is common that more than one keywords in $\{k_j\}$ are mapped to the same ontology terms, so the cardinality of $\{t_i\}$ is usually smaller than that of $\{k_j\}$. The weight assigned to each term t_i is computed using the following formula:

$$r_i = \frac{\sum_{k_j \rightarrow t_i} (n_j \cdot s_j)}{\sum_{k_j \rightarrow t_i} n_j}$$

Each document d_i is represented as $(URL_i, \{(t_i, r_i)\})$, where $r_i \in [0,1]$ since $s_j \in [0,1]$.

We see that the weights are important, since they measure the relevance of our initial mapping, that takes into account the exact words found in the links of the document. This motivates our construction of a similarity measure to take this simple characterization of a document into account.

3.4 Web Document Clustering Algorithms

Clustering aims at organizing patterns into groups, allowing us to discover similarities and differences, as well as to derive useful conclusions about them [30]. Our objective here is to cluster web documents in order to discover meaningful groups of pages. The classic clustering problem is the case of points in a metric space. In our case, the objects to be clustered are sets of (weighted) terms of a domain ontology, between which we have created a similarity measure.

In this space there are no coordinates and ordering as in a Euclidean metric space. We can only compute the similarity between documents given an appropriate similarity measure between sets of terms (proposed in the next section). We embedded the similarity measure in two clustering algorithms, DBSCAN and COBWEB that we used for evaluating the behavior of the similarity measure.

COBWEB is an incremental hierarchical algorithm that in each iteration adds a document to all the levels of the hierarchy. The root of hierarchy contains the whole set of classified instances and the nodes the set of instance that belong to the cluster that each node corresponds to. To insert a new instance into an existing node, COBWEB starts from the root and considers the following four choices at each level as it descends the hierarchy: a) recursively incorporate the instance into an existing node, b) create a new node for the instance, c) merging two nodes to host the instance, and d) splitting an existing node to host the instance. Then the choice that results in the highest *category utility score* is selected.

DBSCAN is a density based clustering algorithm. It relies on the density notion of clusters and can efficiently handle noise (outliers) and arbitrary shaped clusters. In this work we have extended the core idea of DBSCAN clustering to handle sets of documents. More specifically, the algorithm searches for sets of density-

connected documents to identify clusters in the collection of documents. To define a set of density connected documents, the algorithm starts with an arbitrary document d_i and retrieves all the documents that are density reachable from d_i with respect to *MinSim* (the minimum similarity we want to have so as two documents to be considered as neighbors) and *MinDocs* (the minimum number of documents we want to have in the neighborhood of a document). If there is a set of documents that are density reachable from d_i , a cluster is defined. On the other hand if no objects are density reachable from d_i , the document d_i is considered to be *noise* and DBSCAN visits the next document in the database. This procedure is iteratively applied to each document in the database that has not been yet classified. A difference of DBSCAN compared to COBWEB is that DBSCAN characterizes as “noise” the points that cannot be clustered with the others, whereas COBWEB clusters every document. Another difference is that DBSCAN’s results are affected by the values of parameters *MinDoc* and *MinSim* so we must run the algorithm many times to find the best clustering scheme.

The output of the module is the **partitioned set of documents**, some of which are considered as noise (in the case of DBSCAN).

4. A NEW SIMILARITY MEASURE BETWEEN SETS OF WEIGHTED TERMS

4.1 Motivation

As shown in section 2, traditional similarity measures between sets only take into account exact matching. In this section, we extend the ideas of Wu and Palmer [21] in order to propose a distance between terms of a hierarchy. To be able to run the clustering algorithm, we need a similarity measure over documents. In the following paragraphs, we identify the properties such a measure must have, and propose one that has them.

We have the same intuitions about the properties a similarity measure should possess as proposed in [20]. We quote:

Intuition 1: The similarity between A and B is related to their commonality. The more commonality they share, the more similar they are.

Intuition 2: The similarity between A and B is related to the differences between them. The more differences they have, the less similar they are.

Intuition 3: The maximum similarity between A and B is reached when A and B are *identical*, no matter how much commonality they share.

This similarity measure will be used both when clustering the documents, and when answering queries. We expect it will be called upon very often, and thus it is essential that it runs as fast as possible, even on large numbers of documents. Therefore, we bear in mind the complexity of the calculation of this similarity. For scalability reasons, it must also be independent of the number of documents in the database.

4.2 Extending Wu & Palmer to sets

The main contribution of this article is to combine the use of a similarity measure between individual elements of the set of terms, in order to calculate a more accurate similarity between the sets themselves. Let us stress that we can use any given similarity measure on the elements of the sets themselves. Further work would for instance involve devising a measure to take into account terms semantically close, like frequent item sets of IR. We could then use this (enhanced) similarity between words to be even more precise.

We are no longer limited by a binary relation (equality) on words. [13] proposed an interesting study of different measures between sets, and evaluates their complexity. For the algorithms proposed, the complexity ranges from polynomial (in the number of words in each set) to NP. Thus to be competitive, our algorithm needs to have a polynomial behavior with regards to the number of elements in each set.

We are calculating similarity between sets of *weighted* words, and not simply words, but for the sake of simplicity let us first consider the similarity between sets of non-weighted keywords. There has been very little research on creating a similarity measure on **sets of elements** of a space with a similarity measure defined (see [13], [31]). In the next paragraph, we propose a novel similarity measure that is relevant to our problem. This similarity measure is a generalization of Wu & Palmer. We start by giving a similarity between sets of terms of an ontology, then we introduce weights.

4.3 A proposal for a similarity measure between sets of terms of an ontology

4.3.1 Notations

1. Let Ω represent an ontology. We view Ω as a set of terms (words).
2. Let $S_{W\&P}(a,b)$ represent the Wu and Palmer similarity measure between words a and b of Ω .
3. Let $\zeta(A,B)$ represent the similarity measure between subsets A and B of Ω . We call this similarity measure the **extended Wu and Palmer similarity measure**.

Intuition: We interpret in this context the *commonalities* as the terms that are similar for two documents, and the *differences* as the terms that are completely unrelated for the two documents. In order for our measure to reflect this, we calculate first of all the “best” matching for every term in the first document with a term in the second. This tells us how common A is with B . Then we calculate the “best” matching term for every term in document B . This lets us determine the differences, since in the first part of our calculation, we only take into account the terms in B “close” to those in A . If in B there is a term that is not very similar to any term in A , it is taken into account here.

Remark: We say two documents are identical if and only if they are represented by the same subsets of Ω . We define $\zeta(A,B)$ in the following way:

$$\zeta(A,B) = \frac{1}{2} \left(\frac{1}{|A|} \sum_{a \in A} \max_{b \in B} (S_{W\&P}(a,b)) + \frac{1}{|B|} \sum_{b \in B} \max_{a \in A} (S_{W\&P}(a,b)) \right)$$

Note: $|A|$ represents the cardinality of set A .

4.3.2 Properties of $\zeta(A,B)$

1. It is straightforward to check that $\zeta(A,B)$ is a similarity measure: i. $\zeta(A,B)=1$ iff $A=B$ ii. $\zeta(A,B)=\zeta(B,A)$

2. This similarity extends the Wu and Palmer similarity, because the similarity between two sets A and B where $|A|=|B|=1$ is exactly the Wu and Palmer similarity between the (single) element both sets contain (i.e., Let $A=\{a\}$ and $B=\{b\}$, then $\zeta(A,B)=S_{W\&P}(a,b)$.)

3. This similarity measure is tractable in $O(|A| \times |B|)$, assuming that the time to calculate $S_{W\&P}$ is $O(1)$. In general, the complexity of the calculation of $S_{W\&P}$ is in the *worst case* $O(n)$, where n is the depth of the ontology tree (We have not analysed the average complexity calculation of $S_{W\&P}$). So in order to have $S_{W\&P}$ of complexity $O(1)$, we need to cache all the similarity values between each term of the ontology and the others. The preprocessing complexity is $O(n^2)$ since for every term we calculate n similarities with the other terms, and we do this for all the n terms. However, this is done offline, and insures that the online processing is comparable in terms of complexity to the measures proposed in [13]. The size in memory for this caching operation is of $n^2 \cdot \text{sizeof}(\text{similarity})$. For example for an ontology of 400 terms storing the similarity on a short integer of 1 byte length gives a table of size 160KB.

4.3.3 Example

Let us compute the similarity between $A=\{\text{cat, CD}\}$ and $B=\{\text{feline, record, tiger}\}$. First of all we need to calculate the Wu and Palmer similarities between all the words:

$$S_{W\&P}(\text{cat, feline}) = 0.95; S_{W\&P}(\text{cat, record}) = 0.59; S_{W\&P}(\text{feline, CD}) = 0.13; S_{W\&P}(\text{CD, record}) = 0.83; S_{W\&P}(\text{tiger, cat}) = 0.95; S_{W\&P}(\text{tiger, CD}) = 0.2.$$

Let us also note that $|A|=2$ and $|B|=3$.

Thus:

$$\zeta(A,B) = 0.5 \times (1/|A| \cdot (S_{W\&P}(\text{cat, feline}) + S_{W\&P}(\text{CD, record})) + 1/|B| \cdot (S_{W\&P}(\text{feline, cat}) + S_{W\&P}(\text{record, CD}) + S_{W\&P}(\text{tiger, cat})))$$

$$\zeta(A,B) = 0.5 \times (0.5 \times (0.95 + 0.83) + 0.33 \times (0.95 + 0.83 + 0.95)) = 0.89$$

A	B	$\zeta(A,B)$
Cat, CD	Cat, CD	1.0
Cat, CD	Feline, Record,	0.89

	Tiger	
Cat, Tiger, Lynx	Feline	0.80
Spy, Microfilm	Feline, Record	0.47
Book, Dog	Car, Rental	0.35

Table 1. Wu and Palmer similarity between sets

4.4 A proposal for similarity measure between weighted sets

To better characterize our documents, we use sets of weighted words. For example, a document that talks about the Beatles records could be characterized by the following weighted set $\mathcal{A}=\{(1, \text{"Beatles"}), (1, \text{"Records"})\}$, meaning both keywords are equally very important, or $\mathcal{B}=\{(1, \text{"Beatles"}), (0.2, \text{"Records"})\}$ meaning the document mainly deals with the Beatles, and only a little about records. In THESUS these values are determined for each document based on the mapping of keywords found in the incoming links of a document to terms of the ontology, using WordNet, and by calculating the similarities in WordNet using the Wu and Palmer similarity, but this is out of the topic of this paper. We say that two documents are identical if they are represented by the same weighted sets.

However, in order to respect our intuitions of what properties our similarity measure must have, if a word such as "Record" has a low weight, this means that the document does not talk much about records. Therefore, this keyword must not contribute so much in the commonality or difference computation.

4.4.1 Notations

Let Ω represent the ontology (set of words in a hierarchy)

We use *CURSIVE CAPITALS* \mathcal{A}, \mathcal{B} to represent sets of weighted words, such as: $\mathcal{A}=\{(w_i, k_i)\}$, with $k_i \in \Omega$ and $w_i \leq 1$.

We use the corresponding **regular capitals** A, B to represent the sets of keywords without weights associated with a weighted set:

$$A = \{k_i \mid \exists i, (w_i, k_i) \in \mathcal{A}\}$$

We note $\mathcal{A}=\{(w_i, k_i)\}$ and $\mathcal{B}=\{(v_i, h_i)\}$

4.4.2 Intuition

From a mathematical point of view, we want the similarity between sets of terms to be continuous, using the following definition: Let $|\mathcal{A}|=n$ and let $\mathcal{A} = \{(w_i; k_i) \mid i \in [1; n]\}$. Let $|\mathcal{B}|=n+1$ and let $\mathcal{B} = \{(w_i; k_i) \mid i \in [1; n]\} \cup \{(\varepsilon; k_{n+1})\}$. We note $\zeta(\mathcal{A}, \mathcal{B})$ the similarity between the weighted sets \mathcal{A} and \mathcal{B} . We want this similarity to have the limit 1 when $\varepsilon \rightarrow 0$. A corollary to this is that two sets -differing by only one term with weight 0- have a similarity equal to 1. This is obvious, since the fact that a document has a term with an associated weight of 0 means that this term is irrelevant to the document. Of course, in practice, this will not be

the case, since only relevant terms of the ontology are added to the sets that characterize the document. Finally, we want two identical sets of weighted words to have a similarity of 1, regardless of the values of their weights.

Remark: $|A|=|\mathcal{A}|$ but we don't necessarily have $|A|=|B|$, i.e., the documents defined by \mathcal{A} and \mathcal{B} don't have the same number of weighted keywords.

We use the following similarity measure, whose properties we will discuss in the following paragraph. It is in fact an approximation of a more accurate similarity measure that we do not have enough space to detail here, but in most cases it behaves correctly. We define:

$$\zeta(\mathcal{A}, \mathcal{B}) = \frac{1}{2} \left[\frac{1}{K} \sum_{i=1}^{|\mathcal{A}|} \max(\lambda_{i,j} \times S_{W\&P}(k_i, h_j)) \right] + \left[\frac{1}{H} \sum_{i=1}^{|\mathcal{B}|} \max(\mu_{i,j} \times S_{W\&P}(h_i, k_j)) \right]$$

In the case of this similarity, the difficulty is to find a way of calculating the weight factors $\lambda_{i,j}$ and $\mu_{i,j}$ in order to respect all the intuitions that we have given above.

$$\text{Where } \lambda_{i,j} = \frac{w_i + v_j}{2 \times \max(w_i, v_j)} \text{ and } K = \sum_{i=1}^{|\mathcal{A}|} \lambda_{i,x(i)} \text{ with}$$

$$x(i) = x \mid \lambda_{i,x} \times S_{W\&P}(k_i, h_x) = \max_{j \in [1, |\mathcal{B}|]} (\lambda_{i,x} \times S_{W\&P}(k_i, h_j))$$

Simply put, K is a normalizing factor that is the sum of all the $\lambda_{i,j}$ that were used.

In a similar way we can define $\mu_{i,j}$ and H .

Intuitive explanation: the weight factors give less importance to terms that do not describe the document with a high weight. For instance if a document is described by a term with a low weight, we do not want this weight to play too great a role in the *commonality* and the *differences* with other terms describing other documents. Thus $\lambda_{i,j}$ and $\mu_{i,j}$ reduce the overall impact of terms with low weights. We also note that regardless of the values of the weights w_i and v_j the maximal value for $\lambda_{i,j}$ (resp. $\mu_{i,j}$) is equal to 1 and is reached for $w_i = v_j$. We do not have the space here to prove all the properties of this measure that we require, but they are straightforward.

Note: The unweighted formula proposed in section 4.3.1 is a simplification of this formula, using the same value for all the $\lambda_{i,j}$ and $\mu_{i,j}$. We simply have:

$$\forall (i, j) K = |A|, H = |B|$$

and

$$\mu_{i,j} = \lambda_{i,j} = 1$$

It is also reached if we set all the weights to the value 1.

4.4.3 Properties of $\zeta(\mathcal{A}, \mathcal{B})$

1. It is straightforward to check that $\zeta(\mathcal{A}, \mathcal{B})$ is a similarity measure: i. $\zeta(\mathcal{A}, \mathcal{B})=1$ iff $A=B$ ii. $\zeta(\mathcal{A}, \mathcal{B})=\zeta(\mathcal{B}, \mathcal{A})$

2. This similarity extends the similarity measure on sets of keywords, if all the weights in A and B are the same (i.e., $A=\{(\chi, k_i)\}$ and $B=\{(\chi, h_i)\}$). Moreover, let $A=\{(1,a)\}$ and $B=\{(1,b)\}$, we have the extension: $\zeta(A,B)=S_{W\&P}(a,b)$.

3. This similarity measure is tractable in $O(|\mathcal{A}| \times |\mathcal{B}|)$

4. We can apply this similarity measure on any other similarity measure between keywords, not just on the Wu and Palmer measure.

4.4.4 Example

Let us consider $A = \{(0.5, \text{cat}), (1.0, \text{magazine})\}$ and $B = \{(0.8, \text{dog}), (0.3, \text{book})\}$. We have the following Wu & Palmer similarities: $S_{W\&P}(\text{cat}, \text{dog}) = 0.80$; $S_{W\&P}(\text{cat}, \text{book}) = 0.43$; $S_{W\&P}(\text{dog}, \text{magazine}) = 0.46$; $S_{W\&P}(\text{book}, \text{magazine}) = 0.83$. We see that “cat” and “dog” are closest and that “magazine” and “book” are closest. The weight factors are the following:

$$\lambda_{1,1} = \mu_{1,1} = (0.5 + 0.8)/1.6 = 0.81$$

$$\lambda_{2,2} = \mu_{2,2} = (1.0 + 0.3)/2 = 0.65$$

$$\lambda_{1,2} = \mu_{2,1} = (0.5 + 0.3)/1 = 0.8$$

$$\lambda_{2,1} = \mu_{1,2} = (1.0 + 0.8)/2 = 0.9$$

We now check which are the ‘best’ pairing choices, taking into account weights and semantic proximity: We need to compare $\lambda_{1,1} \times S(\text{cat}, \text{dog}) = 0.57$ and $\lambda_{1,2} \times S(\text{cat}, \text{book}) = 0.34$. We see that here we will select to take into account the proximity between (0.5,cat) and (0.8,dog) rather than (0.5,cat) and (0.3, book). We calculate all other possibilities. This boils down to the final calculation:

$$\zeta = \frac{1}{2} \times \left[\frac{(0.81 \times 0.8 + 0.65 \times 0.83)}{(0.81 + 0.65)} + \frac{(0.81 \times 0.8 + 0.65 \times 0.83)}{(0.81 + 0.65)} \right] = 0.81$$

\mathcal{A}	\mathcal{B}	$\zeta(\mathcal{A}, \mathcal{B})$
(1,Cat),(0.5,book)	(1,Cat),(0.5,book)	1.0
(1,Feline),(0.5,Cat)	(1.0, Cat)	0.87
(0.5,Cat)(1.0, magazine)	(0.8, Dog) (0.3, book)	0.81
(0.5,Cat),(0.5,book)	(1,Cat),(1,book)	0.75

Table 2. Wu and Palmer similarity between weighted sets

Due to the fact that we use the simplified version of the measure, we can have an error in the rare cases such as $\{(1.0, \text{cat}) (0.5, \text{book})\}$ and $\{(0.5, \text{cat}) (1.0, \text{book})\}$. With the “exact” calculation of the similarity this does not happen however.

5. EXPERIMENTAL EVALUATION

In this section we present results, based both on synthetic and real web documents. Since evaluating the fact that clustering is ‘good’ or not is subjective, we try to use humans, and evaluate how much the results of each

human tester are different from each other, and compare this to the clustering of our algorithm. This is why in the experiment with the synthetic documents presented in section 5.1 we use a group of ‘blind testers’ to evaluate the quality of the clustering (based on our similarity measure) of a set of synthetic (filtered) documents. We use a filtered set in order to control the quality of the documents presented to each tester. In the second set of experiments, presented in section 5.2, we evaluate our similarity measure using real documents. The measure is evaluated in terms of its dependency: a) on the minimum acceptable weight of the ontology terms in each document description ($\text{min}r_i$), b) on the clustering algorithms employed. It is also compared to other similarity measures.

5.1 Experimentation with synthetic data

In this experiment we cluster a small set of synthetic documents (~100) using our similarity measure and DBSCAN. The experimental protocol is the following: we select 20 pairs of documents that may or may not be clustered together, and we present the list to the tester. The tester says if he would group the documents together or not. We report the percentage of the times his decision agrees with what the system predicts. We gave 10 different testers this list of 20 pairs of documents, and we then checked the clustering algorithm to see if they were clustered together. We asked each tester to give a value on a 1-5 scale to determine if the documents are related or not. We assume a document is clustered together from a ‘human’ point of view when the total of the scores a pair of documents receive is at least MinSim of the maximum score (i.e., all testers ranking it 5/5 relevance means 100% of the maximum score. A tester giving 4 and the 9 others giving it 5 means 98% of the maximum score. If this value is greater than the threshold we set, then the documents are considered to be clustered together by humans, with that given MinSim value) We then compare to the results of the clustering algorithm, using $\text{MinSim}=0.80$ and $\text{MinSim}=0.90$. The lines ‘Related pairs’ show the percentage of pairs of documents of the test set that were clustered together on the one hand by humans and on the other by the system. The results are the following:

MinSim=0.80
Related pairs (Human) : 65% of the test set
Related pairs (calculated) : 50% of the test set
Correlation between Human / System : 85%
Average Human / Human correlation : 81%

MinSim=0.90
Related pairs (Human) : 30% of the test set
Related pairs (calculated) : 25% of the test set
Correlation between Human / System : 95%

Average Human / Human correlation : 81%

The Human / Human correlation is calculated using the average dispersion of the results given by testers. We see that the results given by our system are as “reliable” as those given by a random human tester (over 80% correlation between the results of our system and the “average” human). We believe that this shows that the similarity measure and associated clustering algorithm give meaningful results.

5.2 Experimentation with real data

In this section we evaluate experimentally the similarity measure by clustering sets of real web documents. The results (i.e. clusters of documents) are compared to a classification performed by human experts.

The experiments performed aim: a) to find the threshold value for $\min r_i$ that corresponds to the weight of an ontology term to be included in a document’s description, b) to find the clustering algorithm that performs better, c) to compare our similarity measure to the other document similarity measures.

Experimental protocol

The set of documents used for the experiments consists of approximately 400 URLs categorized by DMOZ under 20 sub-categories (paths) of the “arts/music” path. The whole “music” sub-directory of DMOZ contains 2155 categories, excluding “bands and artists” and “by letter” branches. The ontology we adopted contained 177 terms.

The ontology we use in this experiment is not identical to the part of the DMOZ tree from which we retrieved the test document set. This is done because not all terms in DMOZ paths appear in WordNet and thus cannot be mapped to synsets and moreover not all DMOZ topics under arts/music completely express music related concepts. For example, when DMOZ distinguishes between ‘Gamelan’ and ‘Indian’ music, whereas in our ontology exists only the term ‘World Music’ it is expected that documents from both categories of DMOZ will be grouped together in our approach. Thus it is not expected that the clusters in our approach be identical to the groups of documents classified to the DMOZ paths.

The initial document set D is enhanced with keywords extracted from the hyperlinks of pages that point to them (inlinks). We made the choice to extract characterizations for the pages in D from inlinks only. For a justification refer to [7]. We processed the 100 incoming hyperlinks provided by Google [2] for each document (when available) and aggregated the keywords per document. Each document d_h in D is represented as $d_h = (URL_h, \{(k_j, n_j)\})$ where n_j represents the number of documents that point to d_h using keyword k_j .

For each document, we kept only the keywords having $n_j > 3$. The resulting number of distinct keywords for the whole set of documents is 1436.

Next we used WordNet, the adopted ontology and for each document d_i we mapped the set of keywords $\{k_j\}$ to a set of ontology terms $\{t_i\}$ (see section 3.3). So each document d_h is represented as $(URL_h, \{(t_i, r_i)\})$, where $r_i \in [0,1]$.

The semantically enriched set of documents are clustered using COBWEB and DBSCAN with different input parameters and the resulting clustering schemes are compared to the initial DMOZ partitioning.

The quality of our clustering is measured based on the following criterion:

Let documents d_1, d_2 belong to the same class (path) in DMOZ, if these documents are grouped together in our approach, then this is a sign of success in clustering.

In the experiments that follow we use two external clustering validity measures, F-measure, as presented in [32], and Rand Statistics, presented in [30]. Both external measures, measure the quality of clustering by estimating its overlapping to the known DMOZ classification. For all the documents in the set, all possible pairs are examined. When two documents belonging to the same DMOZ category are clustered together, the clustering quality increases, whereas when documents from different categories are clustered together the clustering quality decreases. Both measures’ values are between 0 and 1, where 1 indicates perfect clustering and 0 indicates that no pair of documents for the same DMOZ category was clustered together. The document set is clustered many times using different input parameters and algorithms.

5.2.1 Clustering quality dependency on $\min r_i$

Assume a document $d_h = (URL_h, \{(k_j, n_j)\})$ and let $d_h = (URL_h, \{(t_i, r_i)\})$ be its enhanced form.

When clustering these documents, we can either:

- take all t_i into account, even those with $r_i \rightarrow 0$ (i.e. terms of little relevance to the original keyword k_i)
- take into account only terms t_i of high relevance to original keyword k_i ($r_i \rightarrow 1$)

It is clear that in the second case several documents may be ruled out of the document set in the case that for none of their terms is true that $r_i > \min r_i$

We experimented with various values for minimum relevance $\min r_i$ measuring the dependency of clustering quality on $\min r_i$.

We modified the COBWEB algorithm to use our similarity measure. As regards the clustering quality measurements we used an external clustering validity measure, F-measure to measure COBWEB’s efficiency for different values of minimum weight. We ran experiments for $\min r_i$ in $\{0, 0.5, 0.9\}$. For a small number of documents (< 50) the clustering quality decreases in roughly the same pace regardless of the $\min r_i$ value. For more documents, it is clear that higher

min_i values result in better clustering quality (see **Figure 2**).

5.2.2 Clustering algorithms comparison

In this subsection we aim at: a) finding the optimal input parameter values for the clustering algorithm DBSCAN with regards to the THESUS similarity measure and the specific data set and b) a comparison between the two clustering algorithms, namely DBSCAN and COBWEB, using our similarity measure. We determined these values on a given test set, and verified that the values were also close to the optimal values for another test sets (a set of over 30000 documents on “space”)

Keeping MinDoc constant and equal to 8 and the minimum weight equal to 0.9 we measured the efficiency of DBSCAN for different values of MinSim. We employed both F-measure and Rand Statistic to evaluate the clustering quality. The results are depicted in Table 2: From the above results we can see that the highest value for F-measure is found for MinSim=0.8 and for Rand Statistic for MinSim=0.75. We conclude that the optimal cluster quality is obtained for MinSim values between 0.75 and 0.8. (see Figure 3).

MinSim	Nr. of clusters	Documents Clustered	Rand Statistic	F-measure
1	6	64	0.355	0.509
0.95	6	64	0.355	0.509
0.9	7	74	0.395	0.520
0.85	8	96	0.483	0.596
0.8	9	124	0.576	0.597
0.75	6	213	0.687	0.387
0.7	4	276	0.530	0.302
0.65	2	324	0.212	0.259
0.6	2	346	0.112	0.25
0.55	2	354	0.074	0.260
0.5	2	355	0.070	0.264
0.45	2	355	0.070	0.264
<0.4	1	356	0.065	0.263

Table 2 External quality measures for DBSCAN results

It is also interesting that clustering with THESUS similarity achieves clusters of high similarity to the ones of DMOZ as indicated by the best values of F-Measure and Rand-Statistic. These values will be used in the following experiments where DBSCAN is compared to COBWEB.

In an attempt to compare the efficiency of the two algorithms we measured their F-measure for different values of minimum term weight. Since DBSCAN results are affected by the MinDoc and MinSim parameters, we tested different combinations of them and kept the best

score for the final clustering scheme. The results can be found in Table 3 and Figure 4.

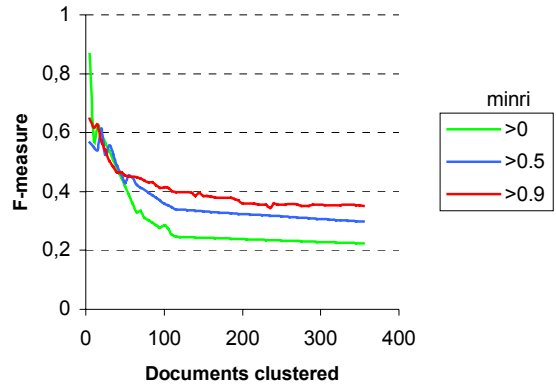


Figure 2 F-measure values for different values of minimum term weight, using COBWEB

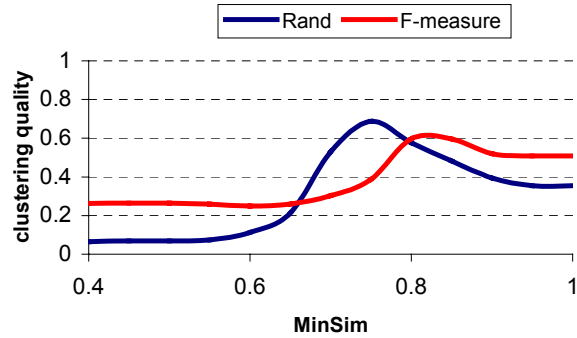


Figure 3 Clustering quality for DBSCAN using different external quality measures

min_i	COBWEB	DBSCAN	MinDoc (DBSCAN)
0	0.221	0.309	1
0.1	0.225	0.298	1
0.2	0.235	0.299	1
0.3	0.231	0.299	1
0.4	0.245	0.32	4
0.5	0.292	0.397	4
0.6	0.315	0.487	8
0.7	0.305	0.459	8
0.8	0.318	0.547	8
0.9	0.350	0.597	8

Table 3 F-measure for the two algorithms

From Figure 4 we can see that the DBSCAN performs better than COBWEB. Again the similarity of the clusters found by the THESUS similarity are sufficiently similar to the original DMOZ clusters (F-measure = 0.597).

In the experiments above, where DBSCAN has been used, we repeat the clustering process many times with different input parameters and keep the clustering scheme with the highest clustering validity score.

5.2.3 Evaluation of the Similarity measure

There are several efforts for document similarity measures such as extended Jacard [33], cosine and the one proposed by Broder et al [34]. All of them are based on exact keyword matching. Here we have to stress that to the best of our knowledge the THESUS similarity measure is the first attempt involving semantic proximity between sets of terms (of an ontology) as opposed to exact keyword matching.

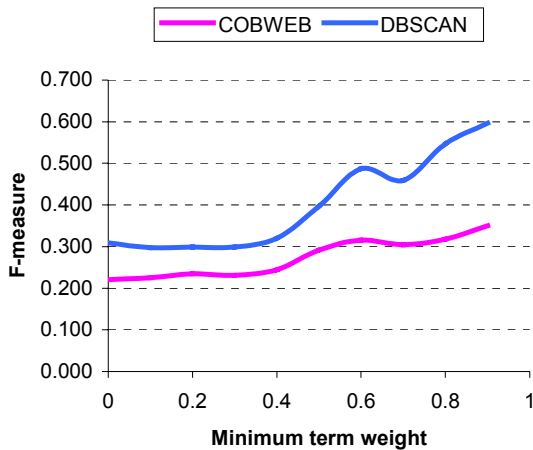


Figure 4 F-measure for DBSCAN and COBWEB

	THESUS similarity	Cosine similarity
MinDoc	8	8
MinSim	0.8	0.5
Documents clustered	124	160
Number of clusters	9	9
F-measure	0.597	0.450
Rand statistic	0.687	0.605

Table 4 Clustering quality using DBSCAN and different document similarity measures

In this subsection we evaluate the THESUS similarity measure in comparison to the cosine one with regards to the F-Measure and Rand-Statistic. We exploited DBSCAN as clustering algorithm and modified it accordingly to use either THESUS similarity of the cosine one. We used the above mentioned document collection and set DBSCAN input parameters MinDoc to 8 while MinSim ranges in (0,1] selecting the value for which the validity results were optimal (0.8 and 0.5 for THESUS and cosine similarity respectively).

The maximum clustering quality value achieved at all levels of MinSim is displayed in Table 4.

Summarizing the results of the experiment we see that our similarity measure outperforms the cosine measure. The clusters created using our measure are more compact than those resulting from the cosine measure, as the MinSim value is higher in our case. Moreover clusters resulting from THESUS similarity are more similar to the original DMOZ clusters than the ones resulting from cosine similarity. This is proven clearly by the values of both external validity measures (see Table 4)

6. CONCLUSION

In this paper we presented a novel similarity measure for documents based on semantic proximity between sets of terms of an ontology as opposed to exact keyword matching (mostly common in other similarity measures). The measure used in the context of a system called THESUS and enables clustering web document collections. The clustering in itself takes only seconds to perform, and we are able to browse the results. By plugging into the system a thesaurus such as WordNet, we are able to answer all sorts of keyword queries.

We proved experimentally that the THESUS similarity measure

- outperforms other similarity measures based on exact keyword matching
- results in document clusters that are sufficiently similar to document classifications produced by human experts.

Future Work:

- We plan on improving the way in which we construct the hierarchy of clusters. We are currently experimenting using the COBWEB algorithm to this end.
- The query system can be made more efficient. This involves improving the mapping techniques that we use in order to determine which word of the ontology a generic word should be mapped to. This is a topic that we are currently working in, and that is very closely linked to this article, since it is used to define the small sets of terms that define a web document.

7. ACKNOWLEDGEMENTS

We would like to thank the following people for their helpful discussions on various topics related to this paper:

G. Cobena and S. Abiteboul (general work on the SPIN projet [35], a similar effort to construct a personal thematic warehouse). M. Theodorakis and A. Vlachos (for the implementation of the COBWEB algorithm for web documents), Ch. Froidevaux, C. Nicaud, K. Nørvåg, B. Safar and L. Segoufin (similarity measures and distances). J.P. Sirot (ontologies). P. Rigaux and P. Veltri (R*-Trees). We would especially like to thank our blind testers, Stratis, Yannis, Magdalini, Georges, Christos, Omar, Mathieu, Julien, Sandrine and Antoinette for their help.

8. REFERENCES

- [1] S. Brin, L. Page, "The Anatomy of a Large Scale Hypertextual Web Search Engine", WWW7 Conference (1997).
- [2] The Google.com search engine: <http://www.google.com/>
- [3] Vivisimo: <http://www.vivisimo.com/>
- [4] Kartoo: <http://www.kartoo.fr>
- [5] T. H. Haveliwala, A. Gionis, D. Klein, P. Indyk. "Evaluating Strategies for Similarity on the Web". *WWW 2002*, Hawaii, USA, (2002).
- [6] M. Halkidi, B. Nguyen, I. Varlamis and M. Vazirgiannis, "THESUS: Organising Web Document Collections Based on Semantics and Clustering", Verso Technical Report (2002)
<ftp://ftp.inria.fr/INRIA/Projects/verso/VersoReport-214.ps.gz>
- [7] I. Varlamis, B. Nguyen, M. Vazirgiannis and S. Abiteboul, "Effective Thematic Selection in the WWW based on Link Semantics", Technical Report (2002).
<http://www.db-net.aueb.gr/michalis/papers/thesus-final2.pdf>
- [8] G. Salton, M. McGill, "Introduction to Modern Information Retrieval", McGraw-Hill, New-York (1983).
- [9] WordNet <http://www.cogsci.princeton.edu/~wn/>
- [10] M. Ester, H.P. Kriegel, J. Sander, X. Xu, "A density based algorithm for discovering clusters in large spatial databases with noise", Proceedings of the 2nd International conference on Knowledge Discovery and Data Mining ACM-SIGKDD (1996).
- [11] M. Ester, H.P. Kriegel, J. Sander, M. Wimmer and X. Xu, "Incremental Clustering for Mining in a Data Warehousing Environment", Proceedings of the 24th VLDB Conference (1998).
- [12] D. Fisher, "Knowledge Acquisition via Incremental Conceptual Clustering", in *Machine Learning 2*: p139-172 (1987).
- [13] T. Eiter, H. Mannila, "Distance measures for point sets and their computation", in *Acta Informatica Journal*, 34, 109-133 (1997).
- [14] J. Green, N. Horne, E. Orlowska and P. Siemens, "A Rough Set Model of Information Retrieval", *Theoretica Informaticae* 28, pages 273-296 (1996).
- [15] A. Gionis, D. Gunopulos, N. Koudras, "Efficient and Tunable Similar Set Retrieval", ACM-SIGMOD (2001).
- [16] A. Bidault, B. Safar, Ch. Froidevaux, "Proximité entre requêtes dans un contexte médiateur", *RFIA-2002*, p653-662, (2002).
- [17] R. Richardson, A. Smeaton, J. Murphy, "Using WordNet as a Knowledge Base for Measuring Semantic Similarity between Words". AICS Conference. Dublin (1994).
- [18] P. Resnik, "Using Information Content to Evaluate Semantic Similarity in a Taxonomy", *IJCAI-95*, pages 448-453 (1995).
- [19] P. Resnik, "Semantic Similarity in a Taxonomy: An Information-Based Measure and its Application to *Problems of Ambiguity in Natural Language*", *Journal of Artificial Intelligence Research* 11, p.95-130 (1999).
- [20] D. Lin, "An Information-Theoretic Definition of Similarity", in proceedings of 15th ICML (1998).
- [21] Z. Wu and M. Palmer "Verb Semantics and Lexical Selection", 32nd Annual Meetings of the Associations for Computational Linguistics, pages 133-138, (1994).
- [22] E. Desmontils et C. Jacquin. "Indexing a Web Site with a Terminology Oriented Ontology". In I.F. Cruz, S. Decker, J. Euzenat et D.L. McGuinness, eds., *The Emerging Semantic Web*. IOS Press, pages 181-198 (2002).
- [23] C. Aggarwal, S. Gates and P. Yu, "On the merits of building categorization systems by supervised clustering", *Proceedings of the 5th ACM-SIGKDD* p.352-356 (1999).
- [24] Zamir, Etzioni, "Web document clustering: a feasibility demonstration", *ACM-SIGIR '98*, (1998)
- [25] J. Kleinberg, "Authoritative sources in a hyperlinked environment", *Journal of the ACM* 46 (1999).
- [26] E. J. Glover, K. Tsioutsoulouklis, S. Lawrence, D. M. Pennock, G. W. Flake. "Using Web Structure for Classifying and Describing Web Pages", *WWW 2002 Conference*, Hawaii, USA (2002)
- [27] Y. Qui and H.P. Frei, "Improving the retrieval effectiveness by using a similarity thesaurus" (1994).
- [28] C. Chekuri, M. Goldwasser, P. Raghavan, E. Upfal, "Web Search Using Automatic Classification", 6th International Conference on the World Wide Web, (1997).
- [29] The DMOZ open directory project <http://www.dmoz.org>
- [30] M. Halkidi, Y. Batistakis, M. Vazirgiannis, "On Clustering Validation Techniques", *Intelligent Information Systems Journal*, Kluwer Publishers (2001).
- [31] I. Niiniluoto, "Truthlikeness", D. Reidel Pub. Comp., Dordrecht, Holland (1987).
- [32] M. Steinbach, G. Karypis, and V. Kumar. "A comparison of document clustering techniques". In *KDD Workshop on Text Mining*, 2000.
- [33] A. Strehl, J. Ghosh. "Value-based customer grouping from large retail data-sets". *SPIE Conference on Data Mining and Knowledge Discovery, Orlando*, volume 4057, pages 33-42. SPIE, (2000).
- [34] A. Broder, S. Glassman, M. Manasse, "Syntactic Clustering of the Web", 6th International Conference on the World Wide Web, (1997).
- [35] S. Abiteboul, B. Nguyen, G. Cobena and A. Poggi, "Construction and Maintenance of a Set of Pages of Interest (SPIN)" in *Bases de Données Avancées*. (2002).
- [36] The DARPA Agent Markup Language Ontology Library <http://www.daml.org/ontologies/>