

Probabilistic Indexing and Categorisation Tool, Intermediate Prototype

Project ref. no.	LE4-8303
Project title	EUROSEARCH

Deliverable status	Restricted
Contractual date of delivery	Month 7
Actual date of delivery	Month 8
Deliverable number	D 4.2
Deliverable Title	Probabilistic Indexing and Categorisation Tool, Intermediate Prototype
Type	Prototype
Status & version	Final 1.0
Number of pages	24
WP contributing to the deliverable	WP 4
WP Task responsible	UNIDO
Authors	Norbert Fuhr, Norbert Gövert, Mounia Lalmas, Fabrizio Sebastiani
EC Project Officer	Marina Manzoni
Keywords	categorisation, classification, probabilistic indexing, category description
Abstract	WP 4 deals with automatic categorisation of web documents that is based on a description oriented approach to document indexing. This deliverable describes further progress with respect to the work done in Deliverable 4.1 as well as an Intermediate Prototype which implements parts of the architecture given in Deliverable 4.1.

Summary

The goal of Work Package 4 (WP4) is to implement a categorisation tool to allow for an automatic categorisation of web documents.

In Deliverable D 4.1, entitled “Categorisation Specification”, the specification of the classification tool was described, and an overall architecture was defined with a number of components. In the present deliverable, we report on further progress with respect to the work carried out for Deliverable 4.1 as well as on the implementation of an Intermediary Prototype of the categorisation tool. We describe the implemented components and some initial results.

Contents

- 1 Introduction** **4**
- 2 Test-bed creation** **5**
- 3 Document indexing** **8**
 - 3.1 Term extraction step 8
 - 3.2 Description step 8
 - 3.3 Decision step 10
 - 3.4 Refinement 11
- 4 Category description generation** **14**
- 5 Categorisation tasks** **16**
- 6 Evaluation methods for categorisation tools** **18**
- 7 Conclusion** **21**

1 Introduction

The aim of Work Package 4 (WP 4) is to implement a categorisation tool to allow for an automatic categorisation of web documents. In Deliverable D 4.1, entitled “Categorisation Specification”, the specification of the categorisation tool was described, and an overall architecture was defined with the following components:

Test-bed creation The categorisation of documents requires a test-bed of pre-categorised documents, upon which the categorisation tasks will be experimented and validated. For this purpose, the *Computers and Internet* category of the *Yahoo!* catalogue is used. Documents from this catalogue must be spidered and then normalised to handle the various structures of web documents.

Document indexing Assigning categories to documents, or vice versa, requires suitable representation of the documents. This necessitates the indexing of documents. The indexing task consists of four steps:

1. Term Extraction step: Terms representing documents are extracted from the normalised documents.
2. Description step: Features are extracted and assigned to document-term pairs, referred to as relevance description.
3. Decision step: Probabilistic weights are assigned to terms on the basis of the relevance descriptions. The weights are determined using the probabilistic approach developed in [FB91].
4. Refinement: The number of terms are reduced through a learning process to allow for a more efficient categorisation of documents.

Category description generation One of the categorisation task in this project works by issuing to the retrieval system a query consisting of a description of the category of interest, where this description is a vector of weighted terms. For this reason, categories must be represented as vectors of weighted terms. We use the Rocchio method to generate category description.

Classification tasks Using both the probabilistic indexing and the category description, documents can then be classified according to some given categories. There are two categorisation tasks. The first one is that given a document to identify its category: *document-centred categorisation*. The second task is to search through a database the documents that satisfy best a given category: *category-centred categorisation*. We use two distinct approaches to carry out the two tasks, respectively, k NN, or k -nearest neighbours, and standard information retrieval.

In the category-centred categorisation, to search documents from a given collection that represent a category best, the retrieval function takes as input the document vector and the category vector. The document vector comes from the indexing

process applied to the web documents, and the category vector is generated by the Rocchio method.

The following modules have been implemented: test-bed creation, term extraction step, description step, and refinement. We have also implemented a baseline indexing method using the standard $tf \times idf$ [vR79], and an initial version of the document-centred categorisation. The implemented components led to an Intermediary Prototype of the categorisation tool.

Other components are currently being investigated, so that to be effectively and efficiently implemented. These are the decision step, the category description generation, and a full implementation of the two categorisation tasks.

In this deliverable, we describe the implemented components, our on-going investigations, and some initial results. Following are four sections, reflecting the main components of our architecture:

- Test-bed creation (Section 2)
- Document indexing (Section 3)
- Category description generation (Section 4)
- Categorisation tasks (Section 5)

There are two other sections. Section 6 introduces evaluation methods used to evaluate the effectiveness of a categorisation tool. Section 7 gives a conclusion.

2 Test-bed creation

The creation of the test-bed required two steps, the spidering of documents and the document normalisation, shown in Figure 1.

Documents from *Yahoo!'s Computers and Internet* catalogue were spidered. Documents directly referenced by *Yahoo!* categories are often a list of entry points to a group of documents with very little content¹, so the documents referenced by the documents directly referenced by *Yahoo!* categories were also spidered, and the corresponding anchors were retained.

The spidering process also recorded relationships between documents and assigned categories, relationships between super and sub-categories, and symbolic links between

¹This is especially true for documents that only contain a frameset with pointers to documents that fill the frames.

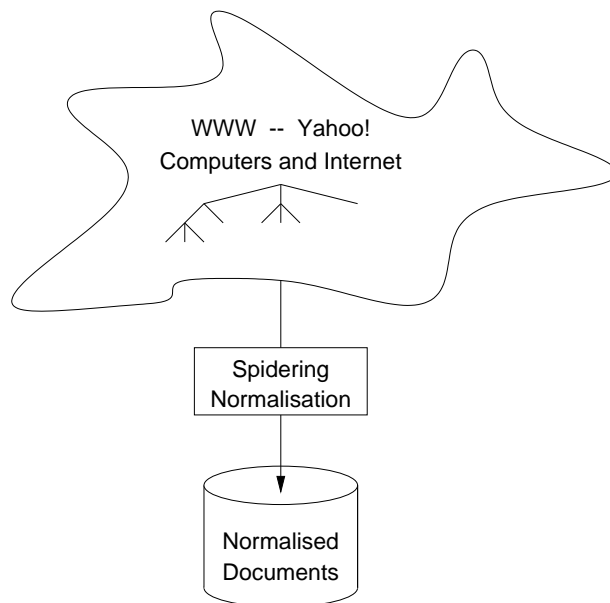


Figure 1: Test-bed creation and normalisation

categories. Access structures have been designed and implemented to efficiently access the relationships and symbolic links.

Table 1 shows various statistics² on the test-bed collection. A leaf document is a document found under a leaf category of the *Yahoo!'s Computers and Internet* catalogue, whereas an inner document is one found under an inner (non-leaf) category.

We have a total of 2806 categories and 18639 documents (some of which may appear under different categories), which means that we have 6.6 documents per category. This number may be changed by merging categories together if such a number does not allow for effective training of the categorisation methods used in this project.

One strategy for merging categories would be to delete all leaf category nodes, and to merge documents referenced by them to their super category, respectively. The effect of this method is shown in Table 2. From the original number of 2806 categories, a number of 809 categories remain, with an average of 23 documents per category.

Table 3 shows the number of categories assigned to a single document. It can be seen that few documents occur twice or more within the *Yahoo!'s Computers and Internet* catalogue. (The number of distinct documents is 17749.)

The normalisation of the documents yielded the documents and those parts of the documents that were considered to index and categorise them. The normalisation approach was as follows:

- Only textual data was considered for indexing purpose.

²Of course Yahoo! changes over time. Our test-bed has been frozen on June, 1998.

	categories			documents			documents/category		
level	inner	leaf	total	inner	leaf	total	inner	leaf	total
0	1	0	1	0	0	0	0.0	-	0.0
1	22	3	25	472	20	492	21.5	6.7	19.7
2	123	171	294	1817	964	2781	14.8	5.6	9.5
3	196	316	512	3084	1804	4888	15.7	5.7	9.5
4	216	529	745	2857	2660	5517	13.2	5.0	7.4
5	157	488	645	1463	1741	3204	9.3	3.6	5.0
6	73	330	403	213	1083	1296	2.9	3.3	3.2
7	21	118	139	57	287	344	2.7	2.4	2.5
8	0	42	42	0	117	117	-	2.8	2.8
total	809	1997	2806	9963	8676	18639	12.3	4.3	6.6

Table 1: Test-bed statistics

	categories			documents			documents/category		
level	inner	leaf	total	inner	leaf	total	inner	leaf	total
0	1	0	1	20	0	20	20.0	-	20.0
1	18	4	22	1217	219	1436	67.6	54.8	65.3
2	62	61	123	1821	1800	3621	29.4	29.5	29.4
3	76	120	196	2571	3173	5744	33.8	26.4	29.3
4	57	159	216	1684	2914	4598	29.5	18.3	21.3
5	27	130	157	797	1749	2546	29.5	13.5	16.2
6	8	65	73	106	394	500	13.2	6.1	6.8
7	0	21	21	0	174	174	-	8.3	8.3
total	249	560	809	8216	10423	18639	33.0	18.6	23.0

Table 2: Test-bed statistics, after applying a merging strategy

categories	documents
1	16933
2	759
3	50
4	5
5	2
total	17749

Table 3: Number of categories assigned to a single document

- A document referred by a *Yahoo!* category (root documents) was indexed on the basis of that document and the documents it links to.
- Only referred documents that are on the same web site of the document referred by *Yahoo!* categories were considered.

We have also split the test-bed into a training sample and a test sample. This was done because the indexing method used in the project is learned-based, that is, it needs data upon which to base its learning process. The way to derive training and test sample is kept flexible. It is possible to determine the ratio at which the whole test-bed is split. The splitting ratio is applied to each category within the test-bed.

3 Document indexing

Assigning categories to documents, or vice versa, requires suitable representation of the documents. This necessitates the indexing of documents. For our Intermediary Prototype, documents are represented as vectors of weighted terms. The determination of the vector follows four step: term extraction step, description step, decision step and refinement step.

3.1 Term extraction step

We extracted terms from the normalised documents. Standard knowledge extraction methods for text was used. Removal of HTML markup was first performed. Then documents were filtered from stop words [vR79] and stemmed [Por80]. The outcome was a list of single words forming the indexing vocabulary, referred to as the term space. Table 4 shows some statistics on the outcome of the term extraction step.

number of documents	17729
average size of documents in bytes	127710
average number of terms per document	4140
average number of distinct terms per document	758
average number of non-root nodes per document	10
size of term space	706624

Table 4: Document statistics

3.2 Description step

The description step consists (Figure 2) of the construction of relevance descriptions for term-document pairs (t, d) . A relevance description $\vec{x}(t, d)$ is defined for each term

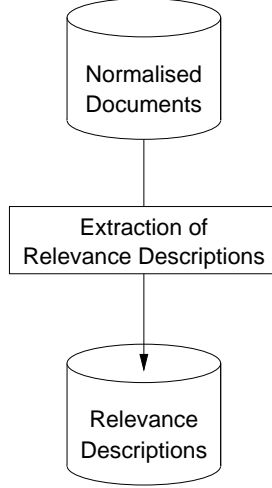


Figure 2: Relevance description extraction

t in the term space and each normalised document d . The vector comprises a set of attributes that are considered to be important for the task of assigning weights to terms with respect to the documents. In our Intermediary Prototype, the dimension of the vector is 10, and its entries are defined as follows:

- $x_1(t, d) = true$ if t is the most frequent term in d ; otherwise $x_1(t, d) = false$;
- $x_2(t, d)$ is the number of terms in document d ;
- $x_3(t, d)$ is the number of distinct terms in document d ;
- $x_4(t, d)$ is frequency of the term t in the document d ;
- $x_5(t, d) = true$ if the term t appears in title of the document d ; otherwise $x_5(t, d) = false$;
- $x_6(t, d) = true$ if the term t is highlighted in the document d (bold, string, emphasised, ...); otherwise $x_6(t, d) = false$;
- $x_7(t, d) = true$ if the term t appears in a heading of the document d ; otherwise $x_7(t, d) = false$;
- $x_8(t, d) = true$ if the term t appears in the first paragraph of the document d ; otherwise $x_8(t, d) = false$;
- $x_9(t, d) = true$ if d is a root document; otherwise $x_9(t, d) = false$;
- $x_{10}(t, d)$ is the inverse document frequency for term t , that is the number of documents in which t occurs.

Note that some attributes depend on both the term and the document, whereas others depend only on the term, or on the document.

This module has been implemented in parallel to the term extraction module. The input of the description step consisted of the database of normalised documents obtained after the spidering phase (Section 2).

The outcome of the description step process is a database of triplets $(t, d, \vec{x}(t, d))$ of term, document, and their associated relevance description $\vec{x}(t, d)$. The relevance description format is a 10-dimension vector, where each dimension corresponds to an attribute of the above list. The dimension entry is the value associated to the attribute as specified above.

3.3 Decision step

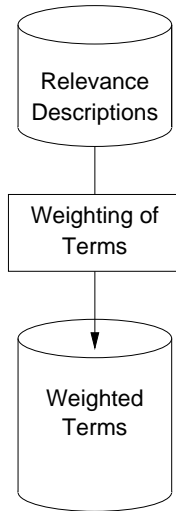


Figure 3: Decision step: probabilistic indexing

In the decision step (Figure 3), documents are indexed by weighted terms using the probabilistic method proposed in [FB91]. The term weights are estimated using a probability $P(R|\vec{x}(t, d))$:

the probability that a document d is judged relevant to an arbitrary category c given that (1) t has relevance description \vec{x} with d , and (2) t has relevance description \vec{x} with a document of the same category c .

These estimates are derived from a learning sample $L \subset D \times D \times \mathcal{R}$ where:

- D is the set of normalised documents;

- $\mathcal{R} = \{R, \bar{R}\}$ for relevant and not relevant;
- $L = \{(d, d', r(d, d')) | d \in D\}$ such that:

$$r(d, d') = \begin{cases} R & \text{if } d \text{ and } d' \text{ belong to same category} \\ \bar{R} & \text{otherwise} \end{cases}$$

Based on L , we form a multi-set of relevance descriptions with relevance judgements

$$L^x = [(\vec{x}(t, d), r(d, d')) | t \in d \cap d' \wedge (d, d', r(d, d')) \in L]$$

This set with multiple occurrences of elements forms the basis for the estimation of the probabilistic index term weights $P(R|\vec{x}(t, d))$. The estimation is done by an indexing function $e(\vec{x}(t, d))$. Different probabilistic or learning algorithms can be applied to derive the indexing function. We use least square polynomials [Kno83] [Fuh89], which was shown effective in [FB93]. The details of the method will be described in the next deliverable.

This module is currently being implemented. The input to the decision step will consist of:

- the database generated by the description step (Section 3.2);
- the relationships between documents and categories (to build the learning sample L). These relationships were derived from the spidering process (Section 2).

The output of the decision step will consist of a database of triplets (t, d, w) of term t , document d , and weight w , where $w = P(R|\vec{x}(t, d))$ is the weight of term t in the document.

3.4 Refinement

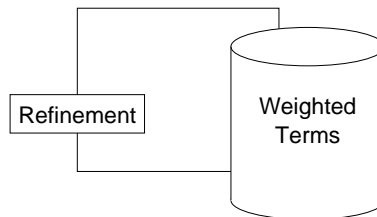


Figure 4: Refinement: Term space reduction

The purpose of the refinement phase is the reduction in size of the term space. This means that, while before refinement a document is represented as a vector of n weighted

terms, with n being the cardinality of the term space, after refinement a document is represented as a vector of m weighted terms, with $m \ll n$. The rationale for this reduction is one of computational efficiency:

- in the document-centred categorisation task, the representation of a document d to be categorised is compared with the representations of all documents in the document set. If the representations of both d and the documents in the collection consist of just $m \ll n$ weighted terms each, the task can be performed much more efficiently than if the representations have size n ;
- in the category-centred categorisation task, the representation of a category c to which relevant documents have to be assigned is compared with the representations of all documents d_i to be categorised. If the representations of both c and the d_i 's consist of just $m \ll n$ weighted terms each, the task can be performed much more efficiently than if the representations have size n .

Because of this, term space reduction is a commonly adopted step in text categorisation efforts [Lew92, LR94, NGL97, SPH95, Yan97].

Term space reduction may be either local or global. Local term space reduction (LTSR) [Lew92, LR94, NGL97, SPH95] has been used in category-centred categorisation, and consists in reducing the size of the vector representing a given category from n to $m \ll n$; the value of m , and hence the size of the reduction, is the same for all categories, but each category may be represented by different terms.

It has recently been suggested that global term space reduction (GTSR) [YP97] produces better effectiveness. GTSR applies both to category-centred and document-centred categorisation, and consists of reducing the size of the vectors representing either categories or documents from n to $m \ll n$. The difference with LTSR is that in GTSR the m selected terms are the same for all categories and for all documents. [YP97] reported surprising results for GTSR. Their experimental results show that the use of sophisticated GTSR techniques, such as those based on the information gain measure (also called expected mutual information measure [vR79, page 41]) or on the χ^2 measure, allow reducing the term space by a factor of 100 without any loss in categorisation effectiveness (actually, even a small improvement in categorisation effectiveness has been reported). They have also shown that a very unsophisticated GTSR measure, such as the one based on document frequency, allows reducing the term space by a factor of 10 also without any loss in categorisation effectiveness. These results are not episodic, since they have been obtained for different categorisation methods [YC94, Yan94] and different document test collections.

For this project, we have decided to adopt GTSR. The reason for this adoption is three-fold:

- as mentioned above, GTSR (and all the various measures that have been proposed for it) apply both to document-centred and category-centred categorisation.

This fact, together with the fact that the m selected terms are the same for all category representations and all document representations, results in a uniform representation of our problem space.

In contrast, LTSR has been applied to category-centred categorisation only. Although similar techniques could in principle be used for document-centred categorisation, this has never been experimented before;

- one of the categorisation methods on which GTSR has already been shown to give good results is exactly the k NN method we intend to adopt for implementing document-centred categorisation (see Section 5);
- the GTSR techniques that have been experimented with in [YP97] allow for an incremental approach to term space reduction: an unsophisticated but computationally easy GTSR measure such as document frequency can be used first, and then replaced by a more effective, albeit computationally more demanding, measure such as information gain. This is not readily possible for LTSR, as no effective application of an “easy” measure has been reported.

In the Intermediary Prototype, GTSR is implemented by means of document frequency. The input to the refinement phase consists of:

- a database of pairs (t, d) of term t and document d , from which document frequencies are computed.
- the relationships between documents and categories (derived from the spidering process).

The rate of the reduction, in the form of a value in the $[0,1]$ interval, has also to be specified as input; this is a parameter set by the designer. In the experimentation phase, different experiments of the complete system may be run with different values for this parameter; this allows us to determine the term space reduction rate for which effectiveness is highest.

The output of the refinement phase consists of a database of pairs (t, d) where t is a term that has not been eliminated, and d is a document.

For producing the Final Prototype, we plan to do an implementation of GTSR by means of information gain, and to replace the document frequency implementation by means of the information gain implementation if this latter gives, as we expect, better results than the former.

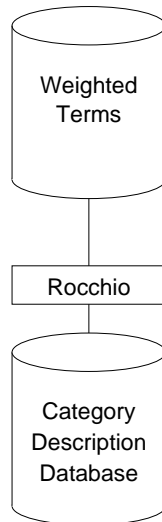


Figure 5: Category description generation with Rocchio

4 Category description generation

Category-centred categorisation works by issuing to an information retrieval system a query consisting of a description of the category of interest, where this description is a vector of weighted terms. As a result of this query, the information retrieval system ranks the documents to be categorised in order of decreasing similarity between the document and the query representations as computed by the retrieval function. The top-ranked documents are categorised under the category c . Exactly how many top-ranked documents have to be categorised under c is defined by a threshold t_c .

The purpose of the category description generation phase is two-fold:

1. the generation of a description of each category c in the catalogue in the form of a vector of weighted terms;
2. the individuation of a threshold t_c for each category c in the catalogue, determining how many top-ranked documents have to be categorised under c .

Various methods have been applied to Task (1) in the literature [Lew92, LR94, Fuh89, Roc71, SB90, CS96, LSCP96, NGL97].

The method that we will implement in the Final Prototype (this task is not yet implemented in this Intermediary Prototype) is the Rocchio method used in [CS96, ILA95]. The motivation for choosing it are three-fold:

- the Rocchio method is conceptually simple and computationally efficient. It is also extremely well-understood, having formed the basis for extensive experimentation of relevance feedback within the vector space model [SB90];

- other algorithms seem best suited to applications in which the training documents become available at different stages, which is not true in our case;
- the Rocchio method allows learning a category description not only from positive instances of the category, but also from negative instances. In our case this seems interesting because our catalogue is tree-shaped, which means that documents categorised under sibling categories of c are extremely interesting negative instances; a similar intuition has been successfully explored in [NGL97], although not in the context of the Rocchio method.

As for Task (2), different notions of threshold may in principle be applicable. For instance, the threshold can be in the form of a percentage t_c , learned from the training sample: only $t_c\%$ of the documents to be categorised are to be categorised under c . This strategy is called proportional assignment. Otherwise, it may be in the form of a similarity value t : only documents whose similarity with the category description, or whose probability of relevance to the category, exceeds t , are categorised under c ; in a probabilistic context, this strategy is called probability thresholding. In the former case the threshold is dependent on the category c , while in the latter case it is the same for all categories.

In the Final Prototype, we plan to use proportional assignment. The reason behind this choice is two-fold:

- we expect proportional assignment to yield better effectiveness than probability thresholding. In fact, in a previous comparative study [Lew92], proportional assignment has yielded better results than probability thresholding when evaluation was performed by microaveraging, although it yielded worse results when evaluation was performed by macroaveraging (see Section 6 for the definitions of microaveraging and macroaveraging). This means that, if proportional assignment is chosen, we can expect a higher number of documents to be categorised correctly, although the results may be more uneven across categories (categories populated with more documents faring better than the others); conversely, if probability thresholding is chosen, more even results across categories are paid in terms of a lower total number of correctly categorised documents. We think the former situation is preferable in our context;
- proportional assignment is easier to reinterpret in terms of different information retrieval engines that may produce a different range of similarity (or estimated probability of relevance) values. This issue is relevant to the porting of this Prototype to the information retrieval systems employed by other partners of the EUROSEARCH consortium.

In the Final Prototype, the input to the category description generation phase will consist of:

- a database of triplets (t, d, w) of term t that has not been eliminated from the refinement process (Section 3.4), document d , and weight w , where w is the weight of term t in document d obtained from the decision step (Section 3.3);
- the relationships between documents and categories (derived from the spidering process).

The output of this phase will consist of:

- a database of triplets (t, c, w) of term t , category c and weight w . w is an element in the $[0,1]$ interval representing the weight of term t in the description of category c ;
- pairs $\langle c, t_c \rangle$, where c is a category and t_c is its computed threshold.

5 Categorisation tasks

Using both the probabilistic indexing and the category description, documents can then be classified according to some given categories. Two categorisation tasks (figure 6) will be tackled in parallel, namely:

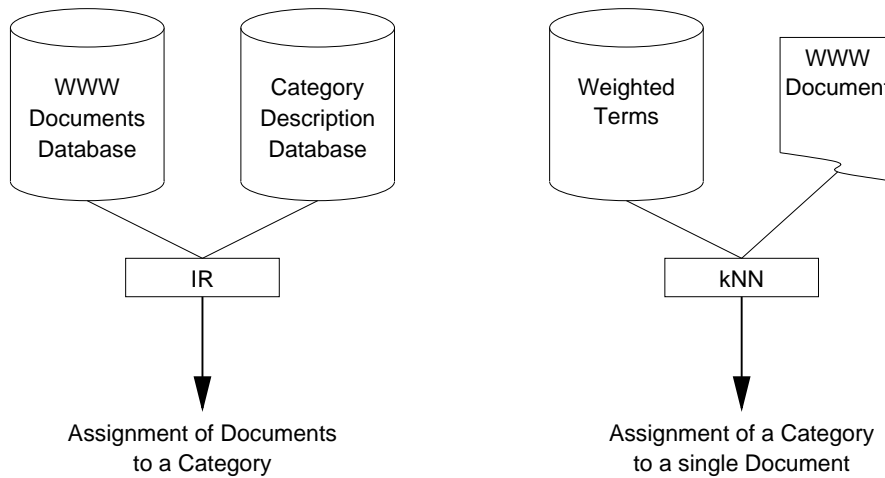


Figure 6: Category-centred and document-centred categorisation

document-centred categorisation starting from document d , the most appropriate category (or categories) under which to categorise d are individuated;

category-centred categorisation starting from category c , the documents most relevant to c are individuated. In this task, a document may be categorised under one or several categories, depending whether it is considered relevant to one or multiple topics, or also under no categories at all.

In the Intermediary Prototype, we accomplished a basic implementation of the document-centred categorisation task. This task is performed using k NN or k -nearest neighbours classifier (Figure 6).

Given an arbitrary document, the method ranks its nearest neighbours among classified documents (the test-bed in our case), and uses the categories of the k top-ranked documents to predict the categories of a new document. The similarity score of each of the (k) neighbour documents is used as a weight of its categories, and the sum of category weights over the k nearest neighbours are used for category ranking.

The technique relies on the existence of an initial corpus $D = \{d_1, \dots, d_s\}$ of documents previously categorised under $C = \{c_1, \dots, c_m\}$. That is, the corpus comes with a “correct” decision matrix.

	d_1	d_j	d_s
c_1	ca_{11}	ca_{1j}	ca_{1s}
...
c_i	ca_{i1}	ca_{ij}	ca_{is}
...
c_m	ca_{m1}	ca_{mj}	ca_{ms}

ca_{ij} is 1 if category c_i has been assigned to document d_j , and 0 otherwise.

To experiment with the k NN classifier, a training phase is necessary. For this purpose, the corpus D is divided into a training set Tr and a test set Te such that $Tr \cap Te = \{\}$ (the two sets are disjoint):

- training set $Tr = \{d_1, \dots, d_g\}$, for which we keep the assigned categories. Tr is used to induce the categories of documents to be classified .
- test set $Te = \{d_{g+1}, \dots, d_s\}$, for which we test the k NN classifier in order to tune it.

The aim is to determine whether k NN assigns or not a category c_i to a document d_j of the test set, this assignment being denoted a_{ij} , such that $a_{ij} = ca_{ij}$.

The k NN algorithm basically comes down to computing for each document d_j in the test set Te :

$$sim(d_j, c_i) =_{def} \sum_{d_z \in TR_k(d_j)} sim(d_j, d_z) \cdot ca_{iz}$$

where $TR_k(d_j)$ is the set of the k documents d_z (the k nearest neighbours) for which $sim(d_j, d_z)$ is maximum, where sim is a similarity score between documents.

We have implemented an initial version of k NN based on standard $tf \times idf$. This can be accessed at the following site:

<http://ls6-www.cs.uni-dortmund.de/projects/ir/eurosearch/consortium/>

The implementation uses $k = 30$ which has been demonstrated by [YC94] to give good performance. The similarity score between documents is the cosine function [vR79].

This implementation will be used as a baseline to be compared with our final implementation of k NN. This implementation is necessary in order to evaluate the effectiveness of the implementation of k NN using the weighted terms resulting from our probabilistic indexing of web documents.

We give now an example of a positive categorisation by k NN of a document of the test-bed. Figure 7 shows some of the documents that are found under the *Yahoo!* sub-category *Communications and Networking:BBSs* (the sub-categories of *Communications and Networking:BBSs* are also displayed). Documents are displayed by means of their URL. URLs of documents from the training set appear in italic, whereas those from documents of the test set appear in bold. For these documents, we can execute the k NN classifier by clicking on the button “[k NN]”. (The button “[external]” provides a link to the original documents.)

We applied k NN to the second document (URL <http://www.ablelink.org/>). The result is shown in Figure 8.

The first section shows the ranked categories as determined by k NN, along with their respective weights. Categories appearing in bold font are perfect matchings (here the category *Communications and Networking:BBSs*). The display of the other categories show two parts: a correct matching at a higher level (shown in italic font) and a mismatch at the lower level (shown in regular font). For instance, the second category is correct up to the level *Communications and Networking*, but *ATM* is incorrect.

The second section shows some of the test documents used to derive the category ranking (the k nearest neighbours). The documents are displayed in decreasing order of their similarity to the document being categorised.

6 Evaluation methods for categorisation tools

To evaluate our indexing approach with respect to the document-centred categorisation task, we will perform an evaluation based on the classic notions of precision (P) and recall (R) but adapted to text categorisation:

precision the probability that, if d_j is categorised under c_i (i.e. $a_{ij} = 1$), this decision is correct (i.e. $ca_{ij} = 1$);



EuroSearch

Categorisation of web documents, prototype



[\[Show Document Nodes\]](#)

Computers and Internet:Communications and Networking:BBSs

Subcategories

- [Computers and Internet:Communications and Networking:BBSs:Adult Oriented](#) [external]
- [Computers and Internet:Communications and Networking:BBSs:Amiga](#) [external]
- [Computers and Internet:Communications and Networking:BBSs:Lists](#) [external]
- [Computers and Internet:Communications and Networking:BBSs:Macintosh](#) [external]
- [Computers and Internet:Communications and Networking:BBSs:OS 2](#) [external]
- [Computers and Internet:Communications and Networking:BBSs:Indices](#) [external]
- [Computers and Internet:Communications and Networking:BBSs:Usenet](#) [external]
- [Computers and Internet:Communications and Networking:BBSs:Channel 1 Communications](#) [external]

Documents

- <http://www.get.es> [external]
- <http://www.ablelink.org/> [k NN] [external]
- <http://www.angelfire.com/mn/MetalX/> [external]
- <http://www.ozemail.com.au/~gtdraven/active> [external]
- <http://www.adstone.com/> [external]
- <http://www.globaldialog.com/AdventureCentral/> [external]
- <http://www.geocities.com/SiliconValley/Peaks/2542/afterhoursbbs.html> [external]
- <http://home.att.net/~deb.ed/afterhours/index.html> [k NN] [external]
- <http://www.ahbqs.com/> [external]
- <http://www.rossnet.com/videofun/default.htm> [external]
- <http://guild.bc.ca/> [k NN] [external]
- <http://alias.flash.net/> [external]
- <http://www.alliance.net/> [external]
- <http://www.almac.co.uk/almac/bulletin/> [k NN] [external]
- <http://home.earthlink.net/~cooper/> [external]
- <http://www.tabbs.com/public/default.htm> [external]
- <http://www.hsv.tis.net/american/> [k NN] [external]

Figure 7: Documents categorised under sub-categories *Communications and Networking:BBSs*

recall the probability that, if d_j should be categorised under c_i (i.e. $ca_{ij} = 1$), this decision is taken (i.e. $a_{ij} = 1$).

Each of them may be estimated on the test set by comparing the $\{a_{ij}\}$ with the $\{ca_{ij}\}$. The comparison may be either:

microaveraging global effectiveness is obtained by globally summing over all individual decisions;

macroaveraging local effectiveness is evaluated separately for each category, and global effectiveness is obtained by averaging over the results of the different categories.

The two approaches are summarised in Table 5.



EuroSearch

Categorisation of web documents, prototype



kNN on <http://www.ablelink.org/>

Computers and Internet:Communications and Networking:BBSs

Category Ranking

```
0.403 Computers and Internet:Communications and Networking:BBSs
0.150 Computers and Internet:Communications and Networking:ATM
0.077 Computers and Internet:Communications and Networking:Electronic Mail:Junk Email
0.062 Computers and Internet:Information and Documentation:Protocols
0.053 Computers and Internet:Communications and Networking:BBSs:Amiga
0.049 Computers and Internet:Communications and Networking:Archives
0.040 Computers and Internet:Communications and Networking:BBSs:Macintosh
0.040 Computers and Internet:Communications and Networking:LANs:Virtual LAN VLAN
0.025 Computers and Internet:Communications and Networking:Conferences
0.023 Computers and Internet:Communications and Networking:BBSs:Adult Oriented
0.020 Computers and Internet:Communications and Networking:Electronic Mail:Information and
0.020 Computers and Internet:Communications and Networking:Electronic Mail:Information and
0.019 Computers and Internet:Programming Languages:Visual
0.019 Computers and Internet:Bibliographies
```

Document Ranking

```
0.039 http://www.tfs.net/~arena/
Computers and Internet:Communications and Networking:BBSs
0.029 http://www.qfc.org/
Computers and Internet:Communications and Networking:ATM
Computers and Internet:Information and Documentation:Protocols
0.025 http://www.idiscover.co.uk/ltango/
Computers and Internet:Communications and Networking:BBSs:Amiga
0.023 http://www.cbl.com.au/
Computers and Internet:Communications and Networking:BBSs
0.023 http://www.blackbox.com/bb/refer.html/tigf012
Computers and Internet:Communications and Networking:Archives
0.023 http://nosys.com/
Computers and Internet:Communications and Networking:BBSs
```

Figure 8: *k*NN applied to document <http://www.ablelink.org/>

Normally, classification algorithms can be tuned so as to improve precision (respectively, recall) to the detriment of recall (respectively, precision). Effectiveness measures should thus combine both. The following options are often chosen:

- effectiveness is computed as 10-point average precision
- effectiveness is computed as the break-even point, the value at which precision equals recall
- effectiveness is computed as the value of the F function

$$F_{\alpha} = \frac{1}{\alpha \frac{1}{P} + (1 - \alpha) \frac{1}{R}}$$

	Recall	Precision
Macroaveraging	$\frac{1}{m} \cdot \sum_{c_i \in C} \frac{\sum_{d_j \in Te} a_{ij} \cdot ca_{ij}}{\sum_{d_j \in Te} ca_{ij}}$	$\frac{1}{m} \cdot \sum_{c_i \in C} \frac{\sum_{d_j \in Te} a_{ij} \cdot ca_{ij}}{\sum_{d_j \in Te} a_{ij}}$
Microaveraging	$\frac{\sum_{c_i \in C, d_j \in Te} a_{ij} \cdot ca_{ij}}{\sum_{c_i \in C, d_j \in Te} ca_{ij}}$	$\frac{\sum_{c_i \in C, d_j \in Te} a_{ij} \cdot ca_{ij}}{\sum_{c_i \in C, d_j \in Te} a_{ij}}$

Table 5: Microaveraging and macroaveraging approaches

If we let α taking values in $[0,1]$, then α may be seen as the degree of importance that the user attributes to precision (if $\alpha = 1$, then F_α coincides with precision, if $\alpha = 0$ then F_α coincides with recall).

Once an effectiveness measure is chosen, classifiers can be tuned (e.g. thresholds can be set such as the value of k for k NN) so that the resulting effectiveness is the best achievable by that classifier.

7 Conclusion

Deliverable D 4.2 consists of an implementation of an Intermediary Prototype of a categorisation tool for web documents. The components of the tool were described in Deliverable D 4.1. Various modules have been implemented:

- **Test-bed creation**
- **Document indexing**
 - term extraction step,
 - description step, and
 - refinement step
- **Classification tasks**
 - document-centred categorisation (baseline implementation)

The others modules are currently being investigated and implemented.

- **Document indexing**

- Decision step
- **Category description generation**
- **Classification tasks**
 - document-centred categorisation (full implementation)
 - category-centred categorisation

This report also presented evaluation techniques that can be used to evaluate our categorisation tool.

Finally, we have designed and constructed a site in which the baseline implementation the document-centred categorisation task is running.

References

- [CS96] W. W. Cohen and Y. Singer. Context-sensitive learning methods for text categorization. In Frei et al. [FHSW96], pages 307–316.
- [FB91] N. Fuhr and C. Buckley. A probabilistic learning approach for document indexing. *ACM Transactions on Information Systems*, 9(3):223–248, 1991.
- [FB93] N. Fuhr and C. Buckley. Optimizing document indexing and search term weighting based on probabilistic models. In D. Harman, editor, *The First Text REtrieval Conference (TREC-1)*, pages 89–100, Gaithersburg, Md. 20899, 1993. National Institute of Standards and Technology Special Publication 500-207.
- [FHSW96] Hans-Peter Frei, Donna Harman, Peter Schäuble, and Ross Wilkinson, editors. *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, 1996. ACM.
- [FIF95] E.A. Fox, P. Ingwersen, and R. Fidel, editors. *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, 1995. ACM. ISBN 0-89791-714-6.
- [Fuh89] N. Fuhr. Models for retrieval with probabilistic indexing. *Information Processing and Management*, 25(1):55–72, 1989.
- [ILA95] David J. Ittner, David D. Lewis, and David D. Ahn. Text categorization of low quality images. In *Proceedings of SDAIR-95*, pages 301–315. 4th Annual Symposium on Document Analysis and Information Retrieval, 1995.
- [Kno83] G. Knorz. *Automatisches Indexieren als Erkennen abstrakter Objekte*. Niemeyer, Tübingen, 1983.
- [Lew92] D.D. Lewis. An evaluation of phrasal and clustered representation on a text categorization task. In Fox et al. [FIF95], pages 37–50. ISBN 0-89791-714-6.
- [LR94] D. Lewis and M. Ringuette. A comparison of two learning algorithms for text categorization. In *Symposium on Document Analysis and Information Retrieval*, Las Vegas, 1994.
- [LSCP96] D. D. Lewis, R. E. Schapire, J. P. Callan, and R. Papka. Training algorithms for linear text classifiers. In Frei et al. [FHSW96], pages 298–306.
- [NGL97] H.-T. Ng, W.-B. Gog, and K.-L. Low. Feature selection, perceptron learning, and a usability case study for text. In Nicholas J. Belkin, A. Desai Narasimhalu, and Peter Willet, editors, *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 67–73, New York, 1997. ACM.

- [Por80] M.F. Porter. An algorithm for suffix stripping. *Program*, 14:130–137, July 1980.
- [Roc71] J.J. Rocchio. Relevance feedback in information retrieval. In G. Salton, editor, *The SMART Retrieval System - Experiments in Automatic Document Processing*. Prentice Hall, Englewood, Cliffs, New Jersey, 1971.
- [SB90] G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4):288–297, 1990.
- [SPH95] Hinrich. Schütze, Jan O. Pedersen, and David A. Hull. A comparison of classifiers and document representations for the routing problem. In Fox et al. [FIF95], pages 229–237. ISBN 0-89791-714-6.
- [vR79] C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 2. edition, 1979.
- [Yan94] L. Yang. A hypertext query language for images. *SIGMOD Record*, 23(1):16–20, March 1994.
- [Yan97] Yiming Yang. An evaluation of statistical approaches to text categorization. Technical Report CMU-CS-97-127, Computer Science Department, Carnegie Mellon University, 1997.
- [YC94] Y. Yang and C. G. Chute. An example-based mapping method for text categorization and retrieval. *ACM Transactions on Information Systems*, 12(3):252–277, July 1994.
- [YP97] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of ICML-97*, 1997.