An Example-Based Mapping Method for Text Categorization and Retrieval

YIMING YANG and CHRISTOPHER G. CHUTE
Mayo Clinic

A unified model for text categorization and text retrieval is introduced. We use a training set of manually categorized documents to learn word-category associations, and use these associations to predict the categories of arbitrary documents. Similarly, we use a training set of queries and their related documents to obtain empirical associations between query words and indexing terms of documents, and use these associations to predict the related documents of arbitrary queries. A Linear Least Squares Fit (LLSF) technique is employed to estimate the likelihood of these associations. Document collections from the MEDLINE database and Mayo patient records are used for studies on the effectiveness of our approach, and on how much the effectiveness depends on the choices of training data, indexing language, word-weighting scheme, and morphological canonicalization. Alternative methods are also tested on these data collections for comparison. It is evident that the LLSF approach uses the relevance information effectively within human decisions of categorization and retrieval, and achieves a semantic mapping of free texts to their representations in an indexing language. Such a semantic mapping leads to a significant improvement in categorization and retrieval, compared to alternative approaches.

Categories and Subject Descriptors: G.1.2 [**Numerical Analysis**]: Approximation—*least squares approximation*; H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*indexing methods*; H.3.3 [**Information Storage and Retrieval**]. Information Search and Retrieval—*retrieval models*; I.2.6 [**Artificial Intelligence**]. Learning—*parameter learning*

General Terms: Experimentation, Theory

Additional Key Words and Phrases. Document categorization, query categorization, statistical learning of human decisions

1. INTRODUCTION

Categorization and retrieval are two important issues in the processing of natural language texts. The two issues are separated, partly because of the different nature of the problems, and partly because the methods used to solve one problem often do not apply to the other. Many retrieval methods share the basic assumption that a query and a document are related only if there are shared words in the query and the document. We call such

This work is supported in part by NIH grants LM-07041, LM-05416, and AR30582.
Authors' address: Section of Medical Information Resources, Mayo Clinic/Foundation, Rochester, MN 55905.

relevance judgment "surface-based matching." Poor performance of surface-based matching is unavoidable in text categorization because of the large gap between the unrestricted vocabularies of documents and the restricted vocabularies of indexing categories. The deficiency of surface-based methods has led to a widespread belief in using the original texts for document representation in retrieval, and a tendency to exclude the use of subject categories for document representation.

We believe that the vocabulary gap problem is common to both text categorization and text retrieval, and that a concept-based vocabulary transformation is crucial for improving the effectiveness of categorization and retrieval. Our solution is to learn a vocabulary transformation from relevance judgments made by humans, that is, to learn associations between document words and indexing categories from manually categorized documents, or to learn associations between query vocabulary and document vocabulary from queries and their related documents assigned by humans. For convenience we will use "term" as a generic term for an element of vocabulary, which can be a word in a free vocabulary, a word in category descriptions, or identifier of a category. We use a Linear Least Squares Fit (LLSF) technique to estimate the likelihood of association between terms of different vocabularies, and refer to the approach as "the LLSF mapping." The method has been partly described in previous papers [Yang and Chute 1992; 1993a; 1993b]; here, we focus on the unified model for categorization and retrieval, on its evaluation in different applications, and on the fundamental differences between this method and alternative approaches.

The LLSF mapping is fundamentally different from surface-based matching, because the relevance judgments are made based only on human assessments in a training sample, not on shared words. We call such relevance judgments "example-based relevance judgments." Relevance feedback is close to our approach in this sense. Relevance feedback has the user indicate which documents are related to a query in an initial retrieval based on surface matching. It then adds the words in the related documents to the query in order to find the documents that are related but were not retrieved by the initial query. The relevance judgments in such an approach are based partly on shared words and partly on relevance assessments by humans. So relevance feedback is a combination of surface-based matching and example-based matching. Relevance feedback has shown significant improvement over surface-based methods [Salton and Buckley 1990]. The major limitation, however, is that it requires relevance information for each query, and cannot use this information for predicting the answers to other queries. Such a restriction makes relevance feedback only rarely applicable to text categorization. For example, most documents in a bibliographic database differ from each other, so documents categorized previously by humans cannot be useful for relevance feedback in the categorization of new ones. Relevance feedback is also not applicable in the categorization of patient records, where a diagonsis often belongs to a unique category of disease, and a procedure description usually matches a unique insurance category for billing. Once the category is known, there is no need for categorization. Relevance feedback requires at least one category to be indicated by humans, and thus is useless for such an application.

Another method using human assessments in retrieval is "the least squares polynomial" (LSP) approach [Fuhr and Buckley 1991]. The LSP retrieval model is essentially a surface-based matching in which only the shared terms of a query and a document are counted in relevance judgments; nonshared terms are ignored. The LSP technique is used for optimizing the weights of shared terms. LSP and LLSF are similar in the sense that both use relevance judgments by humans, and both employ a least squares fit technique; they differ at a higher level of the modeling. LLSF uses human assessments to relate query terms and document terms, while LSP limits *related* terms to *shared* terms. Fuhr et al. [1991] also used the LSP technique in text categorization, known as the Darmstadt Indexing Approach (DIA). The DIA/LSP method establishes term-category associations based on manually categorized documents, which is similar to the LLSF mapping. A major difference between these two methods is the context sensitivity of categorization. DIA/LSP determines term-category associations based on the occurrences of term/categories pairs in the training sample. LLSF determines the associations based on the occurrences of term-combination/category-combination pairs, where the term combination is a training document, and the category combination comprises the categories of the document. The LLSF mapping learned in this way is, therefore, context sensitive. This context sensitivity plays an important role in categorization and retrieval, and is discussed in Section 4.1.

2. METHOD

2.1 Relevance Ranking via Vocabulary Transformation

The goal of text categorization is to assign categories to a document according to the content of the document. The goal of text retrieval is to assign documents to a query according to the information interests of the user. These two problems are similar in that they require splitting a given set of objects into two groups, i.e., the objects that are related to a textual request, and the objects that are unrelated. We use "request" as a neutral term for an input text that can be a document requiring categorization, or a query for retrieval. The "objects" are categories in the case of text categorization, and documents in the case of retrieval. The splitting problem can be further reduced to a problem of relevance ranking of objects, i.e., once the objects are ranked with respect to a request, they can be distinguished as related or unrelated using a relevance score threshold.

The basic question is how to determine the relevance between a request and an object. We are looking for a ranking method that is based on the conceptual similarity between a request and an object, rather than on words shared by the request and the textual description of the object. We consider the vocabulary of requests and the vocabulary of object descriptions to be different. We do not assume the two vocabularies share any words; if they do, the shared words may or may not have the same meaning. For example, a word used by an author in an article may not mean the same as the word in a category description. We need a concept-based transformation of the original words of a request into the vocabulary of object descriptions, and then we can measure the similarity between the transformed request and the objects for a relevance judgment.

We refer to the vocabulary of requests as "the source vocabulary," and the vocabulary of objects as "the target vocabulary." The following are six cases of source/target vocabulary combinations to which our method is applicable:

Case 1. *Text categorization where the requests are documents, and the objects are categories.* A category is represented by an identifier. The source vocabulary comprises the document words, and the target vocabulary comprises the category identifiers.

Case 2. *Text categorization where the source vocabulary comprises the words of documents, and the target vocabulary comprises the words in category descriptions.* An object is a category, and is represented by the words in its description.

Case 3. *Text retrieval where the requests are user queries, and the objects are documents.* A document is represented by its categories (identifiers). The source vocabulary comprises the words of queries, and the target vocabulary comprises the identifiers of categories.

Case 4. *Text retrieval where the source vocabulary comprises the words of queries, and the target vocabulary comprises the words in category descriptions.* An object is a document, and is represented by the words of its categories.

Case 5. *Text retrieval where the source vocabulary comprises the words of queries, and the target vocabulary comprises the words of documents.* An object is a document, and is represented by its own words.

Case 6. *Text retrieval where the source vocabulary comprises the words of queries, and the target vocabulary comprises the words in document titles.* An object is a document, and is represented by the words of its title.

We will use the text categorization of Case 1 to explain our approach. The other cases have been described in previous papers: Case 2 in Yang and Chute [1992], Cases 3 and 4 in Yang and Chute [1993a], and Case 5 in Yang and Chute [1993b]. The effectiveness of LLSF in the different cases is compared in Section 4.4.

2.2 The Numerical Representation Using Vector Spaces

Recall that the categorization problem in Case 1 is to determine the relevance between arbitrary documents and categories; we want to obtain a concept-based transformation from document words (the source vocabulary) to category identifiers (the target vocabulary). If any information or knowledge is useful for such a purpose, it should be reflected in human-assigned relevances between documents and categories. Given a training set of manually categorized documents, it should be possible to learn empirical associations between document words and category identifiers. The problem is that a training sample can only be finite, and the possible documents are infinite. Can we obtain a transformation that guarantees most likely predictions of categories for arbitrary documents, including the ones in the training set and the ones not in the training set? In mathematics there are well-established methods for approximating unknown functions from finite data points, interpolation and fitting techniques, for example. These functions enable us to predict unknown responses to new input data. If we can numerically represent a training sample of documents and their categories, we should be able to solve the transformation using existing numerical techniques.

Vector spaces are commonly used in existing methods of retrieval [Salton 1989; Deerwester et al. 1990] and categorization [Evans et al. 1992; Chute and Yang 1992] for handling word distribution over documents or categories. We use vector spaces for a different purpose; that is, we use two vector spaces to define and to solve a transformation between two vocabularies. For document categorization, we define a source space using document words as the dimensions and a target space using category identifiers as the dimensions; we also define the transformation from the source space to the target space. We represent the training documents as source vectors, and their categories as target vectors. We give the source-vector/target-vector pairs as the input to an LLSF algorithm that computes a likely transformation from a source vector to a target vector, that is, from a document representation using its own words to a representation using categories.

Figure 1 shows an example of the matrix representation of a training set. There are four simplified documents (texts), each assigned to a set of categories. We use two matrices, A and B, to represent the training set. A row in matrix A is a document, and the corresponding row in matrix B is the category set of the document. The columns of matrix A are words, and columns of matrix B are categories. Element a_{ij} of matrix A is the weight of word w_j in the ith document. We adopted the following word-weighting schemes [Salton 1989] as options of a_{ij}.

Option 1. A binary weight, i.e., $a_{ij} = 1$ if word w_j is present in the ith document, and $a_{ij} = 0$ otherwise.

Option 2. The within-document word frequency (TF), i.e.,

$$a_{ij} = \mathrm{TF}_{ij} = \text{the number of times words } w_j \text{ occurs in the } i\text{th document.}$$

Option 3. The Inverse Document Frequency (IDF), i.e.,

$$a_{ij} = \mathrm{IDF}_j = \log\left(\frac{\text{number of documents in the entire collection}}{\text{number of documents with word } w_j}\right) + 1.$$

Text 1. neuropathy and Guillain Barre Syndrome $\xleftrightarrow{\textit{matches}}$ Category set 1 = {c3, c4}
Text 2. neuropathy and Guillain Barre Syndrome $\longleftrightarrow$ Category set 2 = {c2, c3, c4}
Text 3. AIDS and Guillain Barre Syndrome $\longleftrightarrow$ Category set 3 = {c1, c4}
Text 4. AIDS and neuropathy $\longleftrightarrow$ Category set 4 = {c1, c3}

Word	**Weight (IDF)**
t1. AIDS	5.0
t2. and	1.0
t3. barre	8.1
t4. guillain	4.9
t5. neuropathy	4.2
t6. syndrome	3.1

Category
c1. acquired immunodeficiency syndrome
c2. nervous system diseases
c3. peripheral nerve diseases
c4. polyradiculoneuritis

$$\begin{array}{r|cccccc|}
 & t1 & t2 & t3 & t4 & t5 & t6 \\
\text{Text 1} & 0.0 & 1.0 & 8.1 & 4.9 & 4.2 & 3.1 \\
\text{Text 2} & 0.0 & 1.0 & 8.1 & 4.9 & 4.2 & 3.1 \\
\text{Text 3} & 5.0 & 1.0 & 8.1 & 4.9 & 0.0 & 3.1 \\
\text{Text 4} & 5.0 & 1.0 & 0.0 & 0.0 & 4.2 & 0.0
\end{array}
\qquad
\begin{array}{r|cccc|}
 & c1 & c2 & c3 & c4 \\
\text{Category set 1} & 0 & 0 & 1 & 1 \\
\text{Category set 2} & 0 & 1 & 1 & 1 \\
\text{Category set 3} & 1 & 0 & 0 & 1 \\
\text{Category set 4} & 1 & 0 & 1 & 0
\end{array}$$

Matrix A
(training texts)

Matrix B
(categories of training texts)

Fig. 1. The matrix representation of matched text/category-set pairs.

Option 4. The combination (TFIDF) of the above two, i.e., $a_{ij} = \mathrm{TF}_{ij} \times \mathrm{IDF}_j$.

Similarly, elements of matrix B are category weights in the corresponding category set. Category weighting is the same as word weighting described above (except that "word" is replaced by "category"). The matrices in Figure 1 use TFIDF weights for words, and binary weights for categories.

2.3 The LLSF Problem and the Solution

Matrices A and B together give the cooccurrence information about words and categories, and the contextual information about these cooccurrences. Having matrices A and B, we are ready to compute the transformation from the source space to the target space. The transformation is a solution of the LLSF problem as defined below.

Definition 2.3.1 The LLSF problem is to find a matrix $F_{l \times n}$ that minimizes the sum of residual squares

$$\sum_{i=1}^{m} \|\vec{e}_i\|_2^2 = \|F\vec{a}_i^T - \vec{b}_i^T\|_2^2 = \|FA^T - B^T\|_F^2,$$

where

$A_{m \times n}$ and $B_{m \times l}$ are given matrices representing the training pairs;
A^T and B^T are the transposes;
$\vec{a}_i$ and $\vec{b}_i$ are the ith pair in the training set;
$\vec{e}_i = F\vec{a}_i^T - \vec{b}_i^T$ is the error of F in the mapping from $\vec{a}_i$ to $\vec{b}_i$;

$\|\ldots\|_2 = \sqrt{\sum_{j=1}^{l} v_j^2}$ is a vector 2-norm of an $l \times 1$ vector, and

$\|\ldots\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{l} M_{ij}^2}$ is the Frobenius matrix norm of an $l \times m$ matrix.

Theoretically, the LLSF problem always has at least one solution. A conventional method for solving the LLSF is to use singular value decomposition (SVD) [Lawson and Hanson 1974; Golub and Van Loan 1989]. The solution is

$$F = B^T(A^+)^T = B^T U S^{-1} V^T$$

where

A^+ is the pseudoinverse of matrix A and $A^+ = VS^{-1}U^T$;
$(A^+)^T$ is the transpose of A^+;
$U_{m \times p}$, $V_{n \times p}$, and $S_{p \times p}$ are obtained by the SVD in that $A = USV^T$;
U and V contain the left and right singular vectors, respectively;
$S = \mathrm{diag}(s_1, \ldots, s_p)$ contains p nonzero singular values $s_1 \geq \ldots \geq s_p > 0$ and $p \leq \min(m, n, l)$;
$S^{-1} = \mathrm{diag}(1/s_1, \ldots, 1/s_p)$ is the inverse of S.

The SVD is the major part of the LLSF computation. The standard algorithm, known as LINPACK [Dongarra et al. 1979], has a time complexity of $O(m^2 n)$ (assume $m \geq n$), where m is the number of pairs in the training set and n is the number of distinct words in the source space.

The solution matrix $F_{l \times n}$, also referred to as "the LLSF mapping function," is a word-category association matrix where the columns are words in the source space, and the rows are the categories in the target space. The elements of F are the likelihood scores of transforming source words into target categories. These scores are also called "weights." Figure 2 shows an LLSF solution (there can be more than one solution) of the sample training set from Figure 1.

Note the difference between the weights on word-category associations and the word weight or the category weight mentioned in Section 2.2. Word weight measures the importance of a word in a document, and it depends on the word frequency in the document and the word distribution over a document collection. Category weight is similar. Word weights and category weights are estimated without using any relevance information from human assessments. The weight on a word-category association measures how likely it is that a word is related to a category. It depends entirely on the relevance information in a training set. The LLSF assigns weights to the associations in such a way as to minimize the mapping errors globally over the training pairs. A more informative word has weighted associations biased toward some particular categories, while a less informative word has relatively even weights on the associations toward all categories. In Figure 2, for example, the word "AIDS" is biased toward the category "acquired immunodeficiency syndrome" whereas the word "and" is not particularly biased toward any category. Note that these weights are not probabilities; they can be real numbers in any range, including negative numbers. Negative numbers are

$$\begin{array}{l|cccccc|}
 & & & & \textbf{Word} & & \\
\textbf{Category} & \text{AIDS} & \text{and} & \text{barre} & \text{guillain} & \text{neuropathy} & \text{syndrome} \\
\text{acquired immunodeficiency syndrome} & .198 & .020 & -.001 & -.001 & -.002 & .000 \\
\text{nervous system diseases} & -.050 & .003 & .020 & .012 & .059 & .008 \\
\text{peripheral nerve diseases} & -.003 & .028 & -.001 & -.001 & .234 & .000 \\
\text{polyradiculoneuritis} & -.001 & .005 & .082 & .049 & -.001 & .032
\end{array}$$

Fig. 2. A word-category association matrix obtained by solving the LLSF of the training set from Figure 1.

set by the LLSF algorithm and used in combination with positive numbers, to make a biased association between a word combination and a category. For example, word combination {*blood*, *cancer*} should be strongly related to category *Leukemia*, but word combination {*blood*, *pressure*} should not. The LLSF algorithm would set the weight to 0.5 for both the *blood-Leukemia* association and the *cancer-Leukemia* association, and a weight of -0.5 for the *pressure-Leukemia* association. So the word combination {*blood*, *cancer*} would have a total weight of 1.0 on the association toward category *Leukemia*, whereas the word combination {*blood*, *pressure*} would have a total weight of 0 toward that category. Of course, this is an oversimplified example; however, it explains the role of negative weights.

2.4 Relevance Ranking in the Target Space

Having the mapping function F, we can project an arbitrary source vector $\vec{x} = (x_1, x_2, \dots, x_n)$ into its image $\vec{y} = (y_1, y_2, \dots, y_l)$ in the target space by computing

$$\vec{y} = (F\vec{x}^T)^T.$$

The intuitive meaning of the projection is to transform a document in free words into its representation as a combination of weighted categories. We can then compare the transformed document, i.e., vector $\vec{y}$, with the vector representation of each category for relevance ranking. A category is represented as a target vector $\vec{c} = (c_1, c_2, \dots, c_l)$. When a category is represented by an identifier, there is only one nonzero element in vector $\vec{c}$. When a category is represented by the words of its description, the elements of $\vec{c}$ are word weights. The relevance score between a document and a category is defined below.

Definition 2.4.1 The relevance score of category $\vec{c}$ with respect to document $\vec{x}$ is the cosine value of vectors $\vec{c}$ and $\vec{y}$,

$$\text{relevance}(\vec{x}, \vec{c}) = \cos(\vec{y}, \vec{c}) = \frac{y_1c_1 + y_2c_2 + \dots + y_lc_l}{\sqrt{y_1^2 + y_2^2 + \dots + y_l^2}\sqrt{c_1^2 + c_2^2 + \dots + c_l^2}},$$

where vector $\vec{y}$ is the transformation of $\vec{x}$ using the LLSF mapping function F.

The relevance scores give a ranked list of categories for any document. The score ranges from -1 to 1, and the category with the highest score is considered to be the most likely candidate.

3. EVALUATION

3.1 Testing Data

Three data sets, SURCL, MEDIR, and MEDCL, were constructed for the evaluation. Table I summarizes these data sets.

3.1.1 *SURCL: A Data Set of Clinical Categorization.* The patient records in the Mayo Clinic archive include diagnoses and operative reports in natural language texts written by physicians. There are above 850 practicing staff physicians, and the words used in their diagnoses or procedure reports vary by individual and over time. These texts need to be coded using canonical categories for the purpose of billing and research. About 1.5 million patient records are manually coded by experts each year. During the coding process, the texts are segmented into pieces that correspond to single categories.

From the manually coded texts, arbitrarily we chose a cardiovascular subset from the 1990 surgical records, which we named SURCL. This set contains 6150 procedure/category pairs. We sorted the 6150 pairs by category and arbitrarily split them into odd and even halves. The odd half was used as the training set (SURCL Training, 3075 pairs including 1610 duplicates), and the even half was used for testing (SURCL Testing, 3075 pairs including 1575 duplicates). The average length of procedure texts is about 9 words; 99.8% of them have a uniquely matched category; and the rest have two or three categories. There are 281 categories in the cardiovascular subdomain of the canonical classification system ICD-9-CM (International Classification of Disease, 9th Revision, Clinical Modifications) [CPHA 1986]. The chance of a correct categorization of a procedure text by a random assignment is 1 in 281, or 0.36%.

3.1.2 *MEDIR: A Data Set of MEDLINE Retrieval.* This data set was originally designed for an evaluation of Boolean retrieval on MEDLINE [Haynes et al. 1990], the world's largest and most frequently used online bibliographic database. The data set has also been used for evaluations of other retrieval and categorization systems [Hersh 92; Yang and Chute 1993a]. It contains 75 queries, 2344 documents, and human-assigned relevances between the queries and the documents. Each document consists of the words in the title and the abstract of a MEDLINE citation. Subject categories, Medical Subject Headings (MeSH) [NLM 93], were assigned to each document by indexers of the National Library of Medicine.

We split the MEDLINE data set into a training set (MEDIR Training) and a testing set (MEDIR Testing). We did not split the queries into two disjoint sets because of the small number of them. A split would result in little word overlap between the training queries and the testing queries, and the mapping learned from the training set would not be very useful for the testing set.

Table I. The Summary of Data Sets SURCL, MEDCL, and MEDIR

		SURCL	MEDCL	MEDIR
TRAINING	Training Pairs	3075 (with 1610 duplicates)	586 (unique)	71 (unique)
	Requests	3067 procedure texts (with 1606 duplicates)	586 documents (unique)	71 queries (unique)
	Target Objects	140 categories	1832 categories	524 documents
	Unique Source Words	935	7813	296
	Unique Target Categories	140	1832	1689
TESTING	Requests	3067 procedure texts (with 1575 duplicates)	1820 documents (unique)	68 queries (unique)
	Target Objects (AVG objects/request)	281 categories (1 category/text)	3584 categories (17 categories/doc)	1820 documents (14 documents/query)
	Average Words per Request	9	168	8
	Source Words Covered by Training Set (including duplicates)	99%	92%	97%
	Target Categories Covered by Training Set (including duplicates)	97%	90%	78%
	Requests Covered by Training Set	58% of procedure texts	0% of documents	88% of queries
	Objects Covered by Training Set	97% of categories	90% of categories	0% of documents

We split the data in a way that would maximize the overlap between training queries and testing queries, and would make no overlap between the training documents and the testing documents. The former made the mapping function maximally useful for the testing queries; the latter enabled us to verify how much the mapping could capture the relevance among unknown documents. There are 1074 query/document pairs in the given data set; we sorted them by query and split these into odd and even halves. The odd half of the training set contains 524 documents and 71 queries, each having at least one related document in the 524 documents. The testing set contains 1820 documents, none of which is included in the training set, and 68 queries, each having at least one related document in the testing set. The average length of queries is 8 words; the average number of related documents per query is 14. The chance of retrieving a related document of a query by random assignment is 14 in 1820, or 0.77%.

3.1.3 *MEDCL: A Data Set of MEDLINE Document Indexing.* The MEDLINE documents were also used for categorization evaluation. We split the 2344 documents arbitrarily into a training set (MEDCL Training, 586 documents) and a testing set (MEDCL Testing, 1758 documents). There are no duplicates in the entire document collection, and consequently, no overlap between the training documents and the testing documents. The training set contains 7813 unique words and 1832 unique categories. The testing set contains 14,339 unique words and 3430 unique categories; about 42% of these words and 46% of the categories are covered by the training set. The average length of a document is 168 words; the average number of categories per document is 17. The total number of unique categories, including the training set and the testing set, is 4020; the chance of a correct categorization of a document by random assignment is 17 in 4020, or 0.42%.

3.2 Evaluation Measurements

The conventional measures, recall and precision, are used for our evaluation.

Definition 3.2.1 The recall and precision of finding relevant objects for request x are

$$\text{recall}\,(x) = \frac{\text{number of objects found and relevant to } x}{\text{the total number of objects relevant to } x}$$

$$\text{precision}\,(x) = \frac{\text{number of objects found and relevant to } x}{\text{total number of objects found}}.$$

There are two common methods for computing the averaged recalls and precisions over a set of request:

(1) For recall thresholds at 10%, 20%, 30% ... 100%, find as many objects as needed for each request, and average the precisions of the points where the threshold is achieved.

(2) For $k = 1, 2, 3, \ldots,$ compute the precision and recall among the top k objects found for each request, and then average them.

There are arguments about which method is more suitable for evaluation of categorization and retrieval effectiveness [Lewis 1991]. In our opinion, which method is more "suitable" depends on how a categorization system or retrieval system is likely to be used in a particular application. Method (1) is more suitable for situations where an end user checks through the ranked list of candidates until satisfied. The categorization of SURCL texts is a typical example of this situation. Suppose we use the LLSF mapping to rank categories and send them to a coding expert for confirmation. Most (99.8%) of the patient records have a unique category; once the correct category appears to the coding expert, he or she is usually confident enough and does not need to check the rest of the candidates. Therefore, the number of categories being checked varies by input text, depending on where the correct category is in the ranked list.

Method (2) is more suitable for the situation where the end user checks a fixed number of the top-ranking objects for every request. Text retrieval is often such a case. We found that the number of related documents of a query varies from 1 to 81 in the MEDIR collection. This means that the end user does not know the number of related documents of a query until checking through the entire document collection, which is undesirable. A practical solution is to check only k top-ranking documents for each query. The categorization of MEDCL documents, as another example, is a similar case where the number of categories of a document varies from 3 to 50, and fixing the number of categories for an indexer to check through is a reasonable solution.

Despite the differences, either Method (1) or (2) can be informative for comparing the relative performance of different methods. We will use Method (1) for the evaluations in all the tests and Method (2) for additional information in some cases.

3.3 Methods for Comparison

We chose the following methods for comparison with LLSF:

(1) **STR** (STRing matching) is a word-based matching method using vectors for representing requests and objects. In the case of categorization, the requests are documents, and the objects are textual descriptions of categories. In the case of retrieval, the requests are queries, and the objects are documents. Binary weights are used in these vectors. We implemented this method for testing the effectiveness of word-based matching without using statistical word weighting.

(2) **SMART−** is a test of the SMART system without using the relevance feedback component. SMART, developed by Salton's group [Salton 1991], is one of the most representative retrieval systems. We use SMART− for testings the effectiveness of word-based matching with advanced statistical word-weighting techniques. We ran SMART with its default setting of parameters, and tested the word-weighting options of TF and TFIDF (Section 2.2). The better results of these two options will be used in the comparison of different methods.

SMART+ is a test of SMART using relevance information. We modified the relevance feedback scheme of SMART so that it does not require user feedback for identifying the relevant objects for each request. We used the MEDIR Training set, the same one we used for the LLSF mapping, to expand testing queries. That is, we expanded each query by adding the words of the related documents (if any) in the training set, and then ran the SMART system on the expanded queries for retrieving documents from the MEDIR Testing set. We did not apply the relevance feedback part of SMART because that would use different relevant documents for query expansion, and this would make the comparison with the LLSF method difficult. Since our focus is on the effectiveness of relevance information and not on the user interaction part of SMART, this modification would not be inappropriate.

All the above methods were tested on SURCL, MEDCL, and MEDIR, except SMART+ , which was tested on MEDIR only. SMART+ is not applicable to MEDCL because it is not helpful for documents that are different from the training set, and none of the MEDCL testing documents are included in the training set. SMART+ is not applicable to SURCL because it is not helpful for texts that have only one category, like 99.8% of the SURCL texts.

3.4 Results

We tested the LLSF on the SURCL, MEDCL, and MEDIR data sets. We applied a simple preprocessing that removes punctuation and numbers, and changes uppercase letters to lowercase. We did not apply the commonly used lexical techniques such as stemming of lexical variations, converting synonyms to canonical terms, or removing conjunctions, prepositions, and words in a "stop list." Our experimental system is implemented as a combination of C++, Perl, and UNIX shell programming. For singular value decomposition, currently we use a matrix library in C++ [DSC 1991] that implements the same algorithm as in LINPACK. The computation of the mapping function took 1.6 hours for SURCL, 2.3 hours for MEDCL, and 36 seconds for MEDIR on a SUN SPARCstation 10. The category ranking took 0.06 seconds per procedure text of SURCL, and 5.7 seconds per document of MEDCL; the document ranking in MEDIR retrieval took 1.8 seconds per query. Since the computation of the mapping function is only needed once until the data collection is renewed, a real-time response is not required. The categorization/retrieval response time is satisfactory for practical needs.

Figures 3, 4, and 5 compare the testing results of different methods on the data sets SURCL, MEDCL, and MEDIR. For each method, we computed the 10-point average precision, which is the average of the precision values at recalls of 10%, 20%, ..., 100%. For reference, we also included the probability of correct assignment by a random choice in each data set, i.e., the averaged precision of random assignment.

In all of these tests, LLSF outperformed SMART− significantly, which in turn was significantly better than STR. The improvement of SMART− over STR came from the use of statistical weights instead of the binary weighting of STR. The improvement of LLSF over SMART− and STR came from learning empirical associations between free words and categories. We found that only 56% (including duplicates) of the free words in SURCL Testing were covered by the vocabulary of categories. This means that 44% of the words were ignored in the surface-based matching of SMART− and STR. The poorer categorization results are not surprising because of the large loss of information. In contrast, LLSF is not a surface-based matching method. Since we used identifiers for representing categories, there are no shared words in the free texts and the category representations. The word-category associations were learned from the SURCL Training. The percentage of words covered by the training set is 99%; this means that 99% of the words in free texts had connections to categories, and therefore were effectively used in the categorization of texts. Figure 3 shows the results: LLSF had a 170% improvement over STR, and a 42% improvement over SMART−, when using the 10-point average precision value of each method for the comparison.

MEDCL is similar to SURCL in that the vocabulary gaps between MEDLINE documents and MeSH categories are large and significantly limit the performance of the word-based matching of SMART− and STR. LLSF, on the other hand, had a 215% improvement over STR, and a 107% improvement over SMART−. Figure 4(a) shows the 10-point average precision values of these methods; Figure 4(b) compares the recall precision curves obtained by computing the recall precision points for retrieval threshold $k = 1, \ldots, 50$ (Method (2) in Section 3.2). The significant difference between LLSF and the other methods is clearly shown in both graphs.

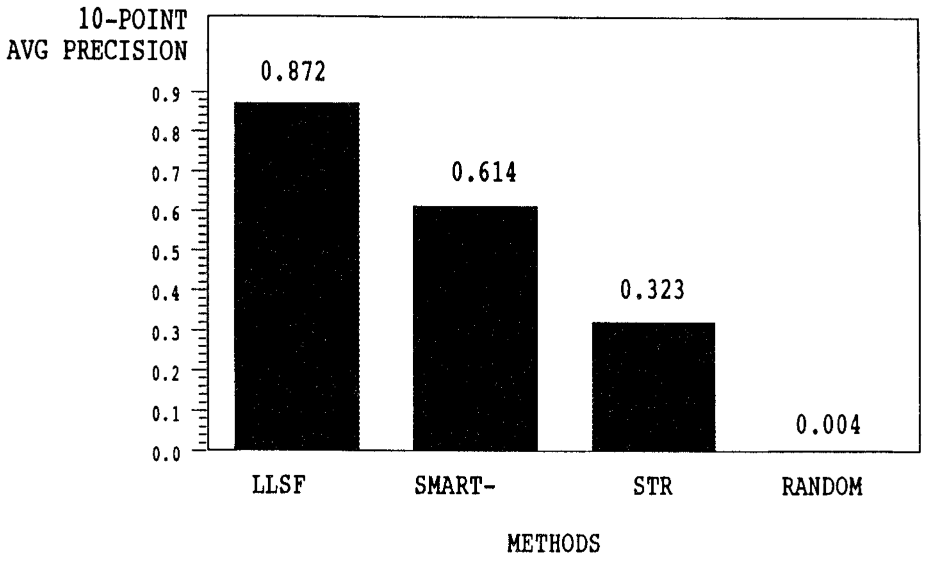


Fig. 3. The results of different methods on SURCL.

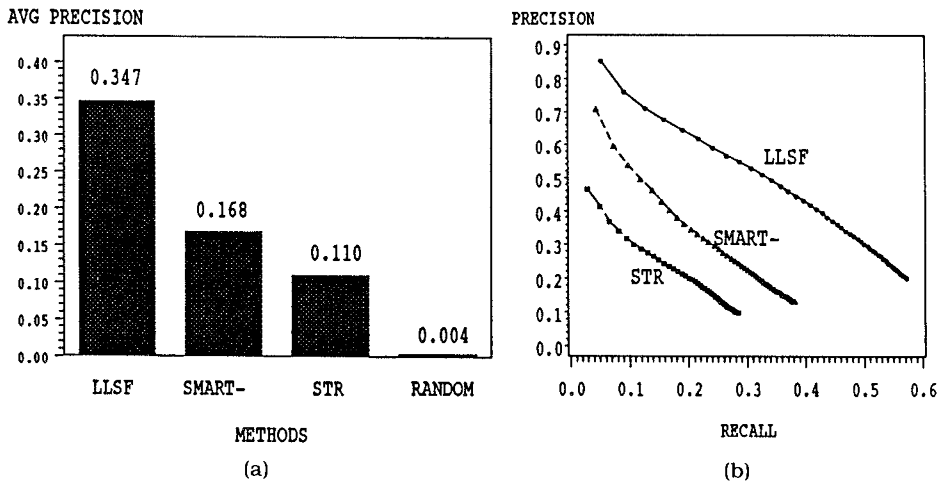


Fig. 4. The results of different methods on MEDCL.

MEDIR is different from SURCL and MEDCL. The task is to match queries and documents. We found that 99% of query words in MEDIR Testing are covered by the document vocabulary; on the other hand, only 39% of document words are covered by the query vocabulary. The documents that share words with a query would be found by the word-based matching of SMART− or STR, but there are possibly many documents that share concepts with a query but happen not to use the same words as those in the query. These

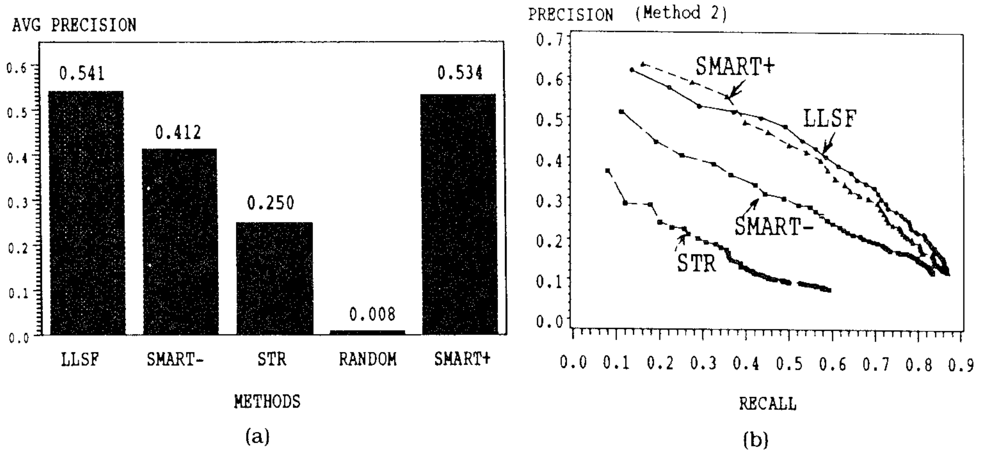


Fig. 5. The results of different methods on MEDIR.

documents cannot be found by word-based matching. This is our interpretation of the unsatisfactory performance of SMART− and STR on MEDIR. The LLSF solves this problem by learning a query-to-categories transformation from a training set of query/document pairs and document/category-set pairs. Once the query is transformed, the broad linkages (concept-based and assigned by humans) between categories and documents can be used for finding the related documents. Figure 5 shows the testing results on MEDIR, where LLSF had a 116% improvement over STR and a 31% improvement over SMART− . SMART+ had a performance similar to LLSF, because it also used human-assigned relevances to relate queries and documents. Unfortunately, it is not applicable to MEDCL or SURCL due to the limitations discussed in Sections 1 and 3.3.

Combining the testing results and these observations, we can conclude that the LLSF mapping is effective in solving the vocabulary gap problem in both categorization and retrieval, and achieves significant improvements over the conventional word-based matching methods.

4. OTHER ASPECTS

4.1 Context Sensitivity of the Mapping

A distinguishing feature of our LLSF model is the context sensitivity of the mapping. By "context," we mean the words that cooccur in a text. By "sensitivity," we mean that the category assignment to a text is dependent on the words in the entire text. Our model does not assume independence among words, and uses a training set of word-combination/category-combination pairs, not word/category pairs. The LLSF mapping function guarantees the optimal category assignments (in the sense of the globally minimized squares error) to texts **contained** in the training set, and relatively accurate category assignments to texts **close** to the training texts. For texts that are very different from the training texts, such category assignment may not be as accurate. For each individual word, no optimal assignment is guaranteed, unless the word happens to be a single-word text in the training set.

In contrast, the LSP approach (Section 1) to text categorization uses term/category pairs for training, where "term" means a word or a phrase predefined in their dictionary. The Least Squares Fit technique is used to minimize the error in category prediction for each individual term, but not to minimize the categorization error for a text as a combination of terms. The LSP approach to retrieval is similar, in that a shared term of a query and a document is used as an independent predictor, and the relevance estimate of a document is optimized with respect to each individual term, not optimized with respect to the term combination in a query.

The crucial question is whether the context sensitivity leads to any difference in text categorization and text retrieval. Intuitively, word combinations are much more informative than individual words, and assumption of word independence is far from the truth of human language usage. Theoretically, a context-sensitive framework is more powerful than a context-free framework. Ideally, we would like to have experimental results of LLSF and LSP on the same data sets for a comparison of effectiveness; however, such results are not currently available. An indirect comparison might be informative, that is, testing each method on both the training samples and the testing samples, and observing the differences.

Figure 6 shows the results of LLSF on the SURCL, MEDCL, and MEDIR training and testing samples, respectively. LLSF exhibits precision between 97.3% and 100% on the training samples, while the results on a testing sample depend on how "close" the testing sample is to the training sample. In SURCL, the results of the training sample and the testing sample are relatively close, partly because 58% of the testing texts are covered by the training sample (see Table I). That is, doctors tend to repeat their phrases, so the overlap between a training sample and a testing sample is relatively large. In MEDCL, the overlap between training documents and testing documents is only 0% (Table I); that is, articles are unlikely to be identical to each other; therefore, the results of LLSF on the training sample and the testing sample differ greatly, compared to the case of SURCL. In MEDIR, 88% (Table I) of the testing queries are covered by the training sample; however, none of the testing documents (target objects) are covered. As a result, the performance difference of LLSF on the training sample and testing sample is smaller than in the case of MEDCL and larger than in the case of SURCL.

Figure 7 shows the testing results of LSP on different retrieval data sets, according to published data [Fuhr and Buckley 1991]. Testing results on five data collections were given: CACM, CISI, CRAN, INSPEC, and NPL. We excluded the results of CISI and NPL because the authors claimed that CISI had "some strange results" and that NPL was "not appropriate" for the evaluation. The results of the remaining three sets are included in Figure 7. Interestingly, the results on the training samples are in a range of 22.1%–36.3%, which is significantly lower than the 97.3%–100% range of

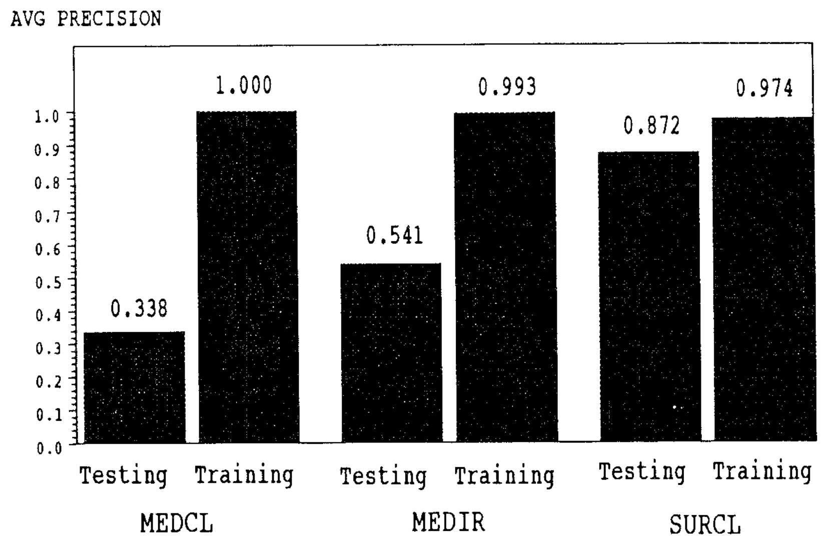


Fig. 6. The results of LLSF on training samples and testing samples

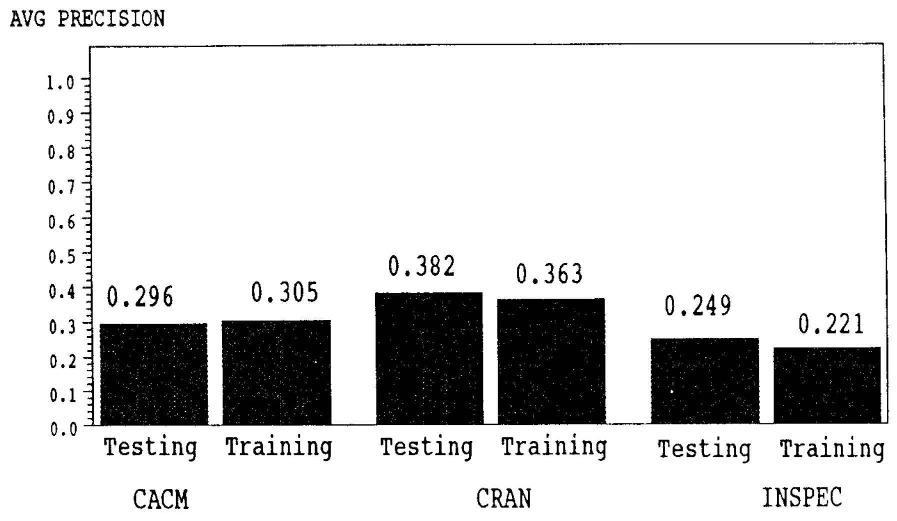


Fig. 7. The results of LSP on training samples and testing samples (based on data from Fuhr and Buckley [1991])

LLSF. Obviously, the optimal document assignments to training queries were not guaranteed, although the relevance estimates from each term in these queries were optimized. This means that the word independence assumption used in LSP sacrifices a large degree of retrieval effectiveness. The results of LSP on each testing sample matched its performance level on the corresponding training sample, which is not surprising.

Combining Figures 6 and 7, we conclude that the context sensitivity plays an important role in categorization and retrieval, and is a fundamental difference between LLSF and LSP.

4.2 Use of Human-Developed Synonyms for Training

We have discussed the belief that the power of the LLSF mapping comes from the learning of human knowledge. Naturally, questions can be asked about whether human-developed synonyms of categories can be used for the training, and whether that would improve the mapping. To answer these questions, we tested the expert-developed synonyms of the clinical categories in the SURCL data set. There are 819 indexing phrases in ICD-9-CM that are defined as synonyms of the 281 cardiovascular categories in SURCL. We used the 281 category definitions and the 819 indexing phrases as training texts, and the 281 categories as training categories. Together they formed a training set (ICD + IND Training) of 1100 text/category pairs. We used this training set and tested the 3067 free texts of SURCL Testing for an evaluation.

Figure 8 compares the results of using ICD + IND Training and SURCL Training on the SURCL Testing texts. The averaged precision is 36.6% when using ICD + IND Training, and 87.2% when using SURCL Training. To analyze possible reasons for the different performances, we checked the source vocabulary coverage of each training set. The ICD + IND Training vocabulary covered 64% of the words in the testing set, whereas SURCL Training covered 99% of these words. This means that 36% of the words in the texts were ignored using ICD + IND Training, while only 1% of the words were ignored in the case of using SURCL Training. The difference in the vocabulary coverage is obviously an influential factor in the different performances. It is not surprising that the people who defined the ICD-9-CM categories and their synonyms did not cover 99% of the words used by Mayo physicians in their daily practice. It is generally true that a general-purpose terminology thesaurus often does not have sufficient vocabulary coverage for all potential applications or specific user groups. It is not difficult, on the other hand, to reach a sufficient vocabulary coverage by sampling from application-specific and user-group-specific data, such as the texts written by Mayo doctors.

For further analysis, we constructed a subset of the SURCL Training that had a vocabulary coverage similar to the ICD + IND set. Our purpose was to compare the effectiveness of SURCL training pairs with ICD + IND under the condition of a similar vocabulary coverage. We split the SURCL Training arbitrarily into odd-even halves, and then split the odd half several times, until the word coverage with respect to the testing set came close to the coverage of ICD + IND. This process resulted in a subset (SURCL−) with 24 training pairs that covered 65% of the testing words. The result of using SURCL− for training is shown in Figure 8; the precision value is 49.0%, which is 12.4% higher than the result of ICD + IND. It is surprising that the 24 pairs of SURCL− have a word coverage even better than the 1100 training pairs of human-developed category descriptions and synonyms.

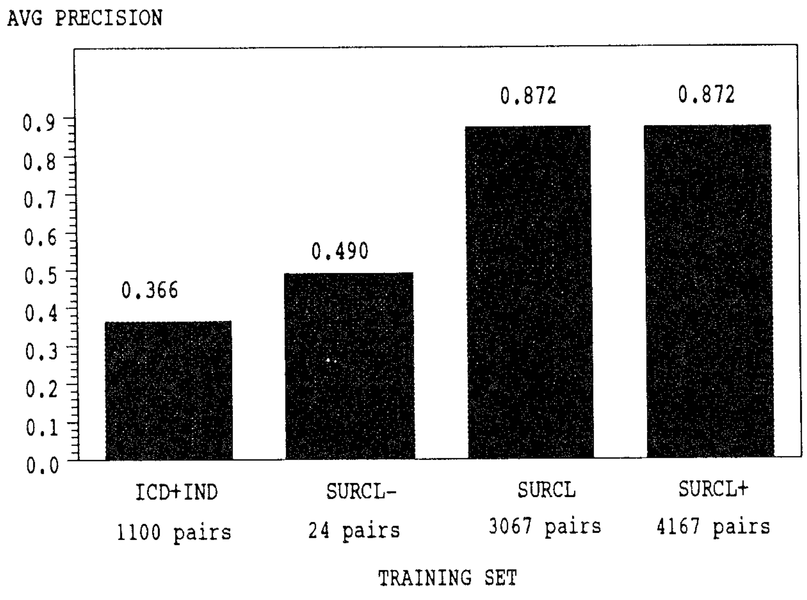


Fig. 8. Results of LLSF using different training sets.

Why did the similar vocabulary coverage of the two training sets lead to different results? Our answer is that SURCL− is more "representative" of SURCL Testing than ICD + IND, in the sense that the word combinations in SURCL− better reflect how doctors phrase their words, and the phrase/category cooccurrences in the training pairs better reflect what doctors mean by their phrases. ICD + IND, on the other hand, contains some phrases which are commonly used by Mayo doctors, and many more phrases that are not. Since each phrase occurs once and only once in the ICD + IND training pairs, the LLSF mapping function is set to satisfy all the phrases evenly, without favoring the commonly used ones. In other words, ICD + IND does not statistically reflect how phrases are likely to be used in practice, and therefore is not a suitable training set for the statistical learning of LLSF.

A further test was to add the ICD + IND Training pairs to the SURCL Training pairs. The resulting training set, SURCL + containing 4167 pairs, had a precision value of 87.2%, which is the same as the result of using SURCL Training alone. This result is also shown in Figure 8.

To summarize our observations, there are three factors that influence the effectiveness of LLSF: the coverage of words, the coverage of word combinations, and the distribution of these word combinations over training pairs. A reasonable sampling from real-world applications would satisfy these criteria; however, human-developed thesauri often do not. They are generally insufficient in vocabulary coverage, even less sufficient in the coverage of word combinations, and particularly insufficient in providing statistical information about how categorization/retrieval decisions depend on word combinations. Because of the lack of statistical information about word usage and word meanings in real-world applications, human-developed thesauri are not suitable training samples for example-based learning methods where such information counts.

4.3 Effect of Morphological Preprocessing

As mentioned in Section 3.4, we did not apply conventional morphological canonicalization techniques such as word stemming and removing words in a "stop list." In order to test how much our results can be improved by using such a morphological canonicalization, we employed a lexical parser (MORPH) developed by the Evans group at Carnegie Mellon [Evans et al. 1991]. MORPH includes regular rules for stemming and a dictionary of 95,000 entires for converting irregular lexical variations, abbreviations, and synonyms to canonical stems. It also removes "noise" words such as conjunctions and prepositions. We tested LLSF on SURCL after it was preprocessed by MORPH. The resulting 10-point average precision was 87.2%, which is identical to the result of LLSF without using MORPH. It is evident from this testing result that most of the lexical variations, synonyms, and abbreviations occurring in the texts of SURCL Testing were covered by SURCL Training, and that therefore, the lexical canonicalization by MORPH could contribute nothing more. It is also evident that since the effect of "noise" words was minimized by the LLSF, having them removed by MORPH did not make a difference.

We also tested the simple word-based matching method STR on SURCL after it was preprocessed by MORPH. The resulting precision was 51.3%, which is much lower than LLSF (87.2%). This test indicates that MORPH partly reduced the vocabulary gap between the patient records and the category definitions, but is not sufficient for filling the gap. LLSF, on the other hand, solved this problem by using a different kind of information, i.e., human-assigned relevances between free texts and categories. Figure 9 compares the testing results of LLSF and STR on SURCL, with or without using MORPH as a preprocessing.

4.4 Effects of Target Languages

We have shown (Section 3.4) the testing results of LLSF, where category identifiers were used for representing categories of SURCL and MEDCL, and for representing documents of the MEDIR set. The question is whether using the original words of category descriptions or using the original texts of documents instead of category identifiers would make a difference. In other words, does the choice of target language for object representation affect the mapping result? Our answer is yes and no. We say "yes" in the sense that the result would be different if the choice were made between target languages that are not equally powerful in representing the contents of objects. We say "no" in the sense that the resulting difference, if any, is not from the mapping process. Our experiments on MEDIR will explain these points.

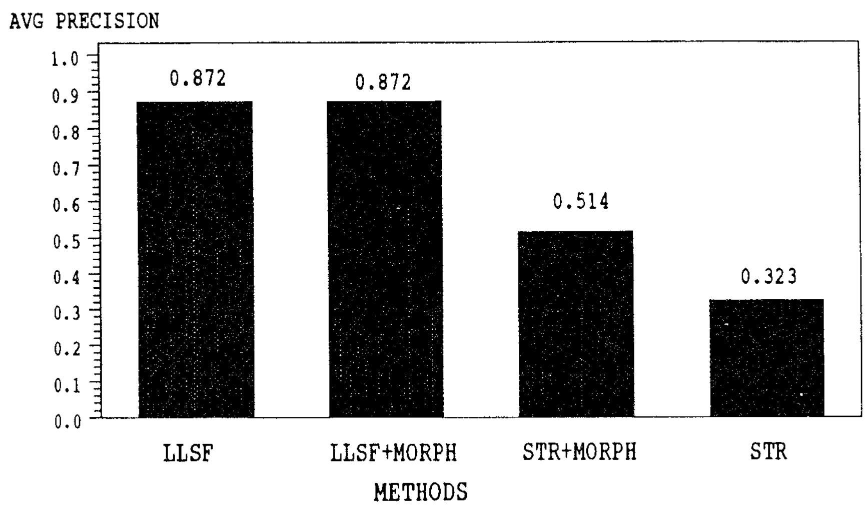


Fig. 9. Effect of MORPH preprocessing on LLSF and STR

Figure 10 shows the results of LLSF on the MEDIR data set using three choices for document representation. A document is represented by either a set of MeSH categories (Case 3 in Section 2.1), the full text (Case 5 in Section 2.1) of the document, or the title text (Case 6 in Section 2.1) of the document. The results of using MeSH categories and of using full texts are similar; we say that the two target representations are almost equally good in representing the documents. The result of using title texts is not as good as the other choices, because a certain amount of information is lost when documents are represented by short texts (the titles) instead of by full texts or categories. This relatively poor result of using titles, however, does not imply that the LLSF mapping is less effective in this case than in the other cases, because before the mapping is learned or applied, the information is already lost in the document representation.

Generally speaking, we can say that LLSF technique guarantees an optimal mapping to the target language, but does not change the power or the quality of the target language in representing the semantics of the target space. Therefore, the choice of target language is important for the overall performance of our approach to classification and retrieval. Whether a particular target language is a good choice is dependent on applications. In our tests of LLSF on SURCL, using ICD-9-CM category identifiers (Case 1 in Section 2.1) was as good as using category words (Case 2); in the tests on MEDCL document categorization, we observed significant improvement from the use of category identifiers rather than category words (Case 2); in the case of MEDIR document retrieval, as mentioned above, using categories for document representation was as good as using the original documents.

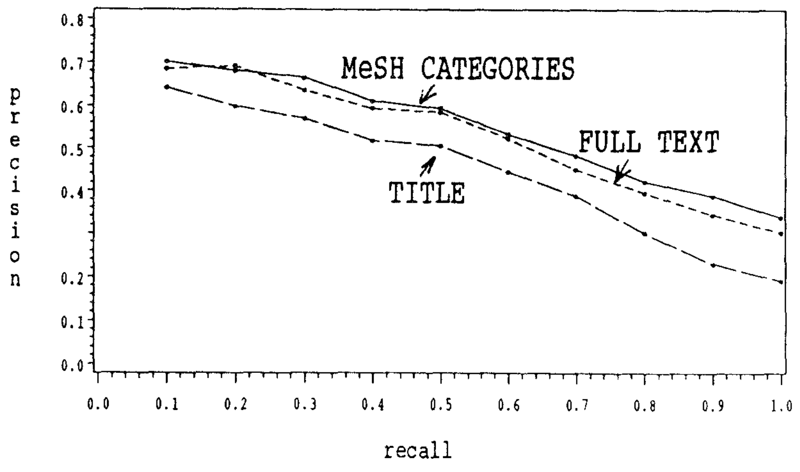


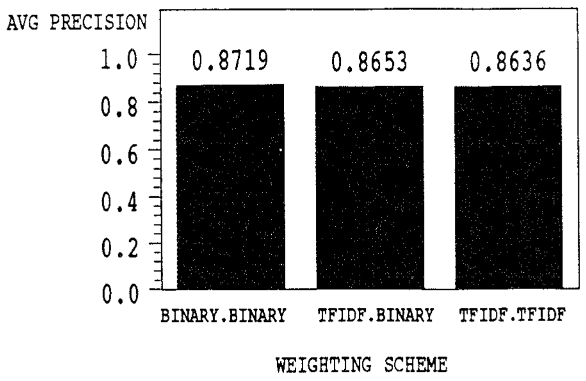
Fig. 10. Effects of indexing languages for document representation.

4.5 Effects of Weighting Schemes

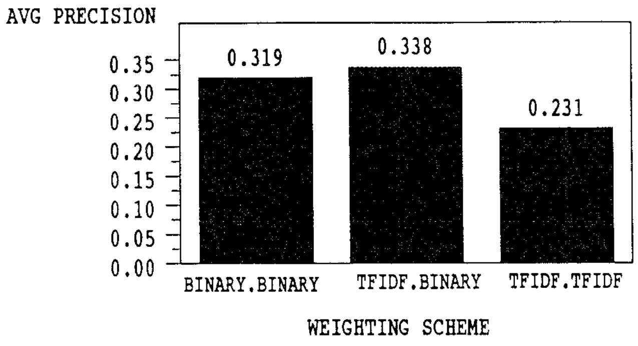
In Section 2.2, we described the options of word weighting and category weighting: the binary weight, the within-request term frequency (TF), the Inverse Document Frequency (IDF), or the combination TF $\times$ IDF (TFIDF). The TFIDF weighting scheme is commonly used in retrieval systems in order to emphasize informative words in a text and reduce the effect of "noise" words. It assumes that a word that occurs more frequently in a text is more important, and that a word that is widely distributed over the entire text collection is less meaningful. The underlying assumption of TFIDF in category weighting is similar. Whether such an assumption is generally true is an open question.

We tested all the weighting schemes on SURCL, MEDCL, and MEDIR, and will compare the two most distinguishable here, the TFDIF weight and the binary weight. Figure 11 shows the results. We use BINARY.BINARY to mean that both source words and target categories are assigned binary weights, TFIDF.BINARY to mean (1) that source words have TFIDF weights and (2) target categories have binary weights, and TFIDF.TFIDF to mean that both source words and target categories are assigned statistical weights. By comparing the results of BINARY.BINARY and TFIDF.BINARY, we can see the effect of weights on source words; by comparing the results of TFIDF.BINARY and TFIDF.TFIDF, we see the effect of weights on categories.

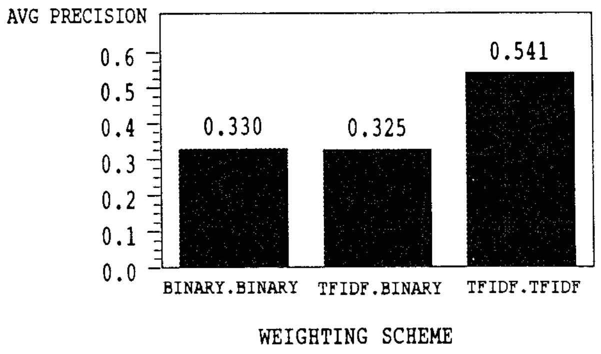
Figure 11(a) shows the results on SURCL. No significant difference is observed between BINARY and TFIDF. This means that the words are relatively evenly distributed over texts, and that the categories are relatively evenly distributed over the category collection. In such a case, neither scheme makes a difference in the result.



(a) The results of different weighting schemes on SURCL



(b) The results of different weighting schemes on MEDCL



(c) The results of different weighting schemes on MEDIR

Fig. 11. Effects of the weighting schemes.

Figure 11(b) shows the results on MEDCL. Using TFIDF for source words has a small improvement over BINARY, but using TFIDF for categories made the result much worse than using BINARY. This implies that a MeSH category used frequently as a subject of documents can still be important or informative in identifying the content of documents. The basic assumption of TFIDF does not apply to this case.

Figure 11(c) shows the results on MEDIR. Using TFIDF or BINARY for source words did not make any difference, while using TFIDF for category weighting made a significant improvement over BINARY. This implies that the category distribution agrees with the basic assumption of TFIDF, from a retrieval point of view. In MEDIR, each query is related to a large set of categories via relevant documents. That is, there are 17 related documents per query on average, and about 14 categories per document. A query is, therefore, transitively related to $14 \times 17 = 238$ categories, including duplicates. It is reasonable to consider a category that occurs more frequently in such a category collection to be more relevant to the query. This agrees with the assumption of TFIDF.

Combining the observations of the above tests, we see that the basic assumption of the conventional statistical-weighting scheme TFIDF is not generally true. Often it is not obvious that one weighting scheme is better than another unless a gross difference in the resulting performance is observed. Our experience suggests leaving the decision on weighting schemes to application and experiment.

5. DISCUSSION

We have studied the nature of the LLSF mapping as a solution for classification and retrieval of natural language texts. Our approach is characterized by

(1) the example-based nature that learns from categorization or retrieval decisions made by humans and obtains empirical associations between free words and indexing terms, and
(2) the optimization algorithm that minimizes the errors of mapping free texts to their representations using indexing terms.

The use of human-assigned relevance makes our approach fundamentally different from surface-based matching methods, methods that are based on morphological canonicalization, and methods that are based on human-developed synonyms or terminology thesauri. There is often confusion when comparing different methods at a knowledge level and at a representation level. For example, "vector based" is a commonly used phrase to describe a group of retrieval or classification methods. Such a grouping is based on *how* information is represented and not on *what* information is used in making a retrieval or categorization decision. SMART, STR + MORPH (Section 4.3), and the LLSF mapping, for example, are all "vector based", but the matching criteria used in these methods are different. SMART without relevance feedback is no more than a surface-based matching, and is powerless to solve the problem of vocabulary differences between queries and documents, or between documents and categories. STR + MORPH is a method based on linguistic knowledge about lexical variations and human-developed terminology thesauri. This method is not a reliable solution for the vocabulary gap problem, because it relies on its own vocabulary, and because the vocabulary does not necessarily cover the needs of various applications. LLSF is a method of solving vocabulary differences based on human decisions in categorization and retrieval. It learns from training samples of particular applications, and captures the word usage in these applications. SMART with relevance feedback (and its variation, SMART+ , which we introduced in this article) is similar to LLSF in that it also uses relevance information from human assignments; it is different in that it limits the use of the relevance information of a particular query to that query only.

Another confusion might arise between LLSF and the Latent Semantic Indexing (LSI) method, which has been studied for both text retrieval [Deerwester et al. 1990] and text categorization [Evans et al. 1992; Chute and Yang 1992]. Since LLSF uses SVD as a part of the computation, one might think that LLSF is similar to LSI, which is also based on a SVD computation. However, the fundamental difference between LSI and LLSF can be clearly seen at the knowledge level: LSI is not a learning method based on human decisions in categorization and retrieval, and does not use two vector spaces to learn a mapping of texts from one vocabulary to another. Besides, while SVD is essential to LSI, it is not the only numerical method for solving LLSF. We could use a QR decomposition instead of SVD to solve LLSF, for example, and the "similarity" between LSI and LLSF would disappear in that case. We mention SVD because it is the computationally intensive part of the current algorithm we use, and would be informative to readers who are interested in the computational aspects.

To summarize our study, we claim that human knowledge is crucial for the effectiveness of text categorization and text retrieval, and we showed that the knowledge can be statistically learned from human decisions in these tasks. The LLSF approach, using an existing fitting technique and a vector-matrix representation, captured effectively a concept-based mapping of free texts to their representations in an indexing language. The significant improvement over alternative approaches is evident in our tests.

ACKNOWLEDGMENT

We would like to thank David Wills, David Lewis, and Craig Stanfill for suggesting many valuable improvements, and Geoffrey Atkin for programming assistance.

REFERENCES

CHUTE, C. G., AND YANG, Y. 1992. An evaluation of concept based Latent Semantic Indexing for clinical information retrieval. In *Proceedings of the 16th Annual Symposium on Computer Applications in Medical Care, vol. 16*. McGraw-Hill, New York, 639–643.

CPHA. 1986. *International Classification of Diseases*. 9th Rev. *Clinical Modifications*. Commission on Professional and Hospital Activities, Ann Arbor, Mich.

DEERWESTER, S., DUMAIS, S. T., FURNAS, G.W., LANDAUER, T. K., AND HARSHMAN, R. 1990. Indexing by Latent Semantic analysis. *J. Am. Soc. Inf. Sci. 41*, 6, 391–407.

DONGARRA, J. J., MOLER, C. B., BUNCH, J. R., AND STEWART, C. W. 1979. *LINPACK Users' Guide*. SIAM, Philadelphia, Pa.

DSC. 1991. *M++ Class Library, User Guide. Rel.* 3. Dyad Software Corporation, Bellevue, Wash.

EVANS, D. A., CHUTE, C. G., HANDERSON, S. K., YANG, Y., MONARCH, I. A., AND HERSH, W. R. 1992. Mapping vocabularies using "Latent Semantics." In *MEDINFO 92*. 1462–1468.

EVANS, D. A., HERSH, W. R., MONARCH, I. A., LEFFERTS, R. G., AND HANDERSON, S. K. 1991. Automatic indexing of abstracts via natural-language processing using a simple thesaurus. *Medical Decision Makıng 11*, 4, 108–115.

FUHR, N., AND BUCKLEY, C. 1991. A probabilistic learning approach for document indexing. *ACM Trans. Inf. Syst. 9*, 3, 223–248.

FUHR, N., ET AL. 1991. AIR/X—a rule-based multistage indexing systems for large subject fields. In *Proceedings of the RIAO'91*. 606–623.

GOLUB, G. B., AND VAN LOAN, C. E. 1989. *Matrix Computatıons*. 2nd ed. The John Hopkins University Press, Baltimore, Md.

HAYNES, R., MCKIBBON, K., WALKER, C., RYAN, N., FITZGERALD, D., AND RAMSDEN, M. 1990. Online access to MEDLINE in clinical settings. *Ann. Int. Med. 112*, 1, 78–84.

HERSH, W. R., HICKAM, D. H., AND LEONE, T. J. 1992. Words, concepts, or both: Optimal indexing units for automated information retrieval. In *Proceedings of the 16th Annual Symposıum on Computer Applıcations ın Medical Care, vol. 16*. McGraw-Hill, New York, 644–648.

LAWSON, C. L., AND HANSON, R. J. 1974. *Solving Least Squares Problems*. Prentice-Hall, Englewood Cliffs, N.J.

LEWIS, D. D. 1991. Evaluating text categorization. In *Proceedings of the Speech and Natural Language Workshop*. Morgan Kaufmann, San Mateo, Calif., 312–318.

NLM. 1993. *Medıcal Subȷect Headıngs (MeSH)*. National Library of Medicine, Bethesda, Md.

SALTON, G. 1991. Development in automatic text retrieval. *Science 253*, 974–980.

SALTON, G. 1989. *Automatic Text Processıng: The Transformatıon, Analysis, and Retrieval of Information by Computer*. Addison-Wesley, Reading, Mass.

SALTON, G., AND BUCKLEY, C. 1990. Improving retrieval performance by relevance feedback. *J. Am. Soc. Inf. Sci. 41*, 4, 288–297.

YANG, Y., AND CHUTE, C. G. 1993a. An Application of least squares fit mapping to text information retrieval. In *Proc. of the 16th Annual Internatıonal ACM SIGIR Conference on Research and Development in Information Retrıeval*. ACM, New York, 281–290.

YANG, Y., AND CHUTE, C. G. 1993b. Words or concepts: The features of indexing units and theır optimal use in information retrieval. In *Proceedings of the 17th Annual Symposium on Computer Applıcations ın Medical Care, vol. 17*. McGraw-Hill, New York, 685–689.

YANG, Y., AND CHUTE, C. G. 1992. A linear least squares fit mapping method for information retrieval from natural language texts. In *Proceedings of the 14th International Conference on Computational Linguistıcs (COLING 92)*. McGraw-Hill, New York, 447–453.

Received June 1993; accepted March 1994