Text and Hypertext Categorization

Houda Benbrahim[1] and Max Bramer[2]

Abstract Automatic categorization of text documents has become an important area of research in the last two decades, with features that make it significantly more difficult than the traditional classification tasks studied in machine learning. A more recent development is the need to classify hypertext documents, most notably web pages. These have features that add further complexity to the categorization task but also offer the possibility of using information that is not available in standard text classification, such as metadata and the content of the web pages that point to and are pointed at by a web page of interest. This chapter surveys the state of the art in text categorization and hypertext categorization, focussing particularly on issues of representation that differentiate them from 'conventional' classification tasks and from each other.

1 Introduction

Over the past two decades, automatic content-based document management tasks have received a great deal of attention, largely due to the increased availability of documents in digital form and the consequential need on the part of users to access them in flexible ways. Text categorization or text classification, which is the assignment of natural language texts into predefined thematic categories, is one such task.

Hypertext categorization is a variant of text categorization, where the documents concerned are web pages. The nature of web pages presents additional challenges when representing the information they contain, but also offers the opportunity to include information such as metadata that is unavailable with standard documents.

[1] Ernst and Young LLP, 1 More London Place, London SE1 2AF, United Kingdom
hbenbrahim@uk.ey.com

[2] School of Computing, University of Portsmouth, Portsmouth, Hants PO1 3HE, United Kingdom
max.bramer@port.ac.uk

This chapter begins by summarizing the state of the art in text categorization and then goes on to look at hypertext categorization, concentrating particularly on the differences from standard text that derive from the special nature of web pages. The emphasis is on illustrating how the information in web pages can best be represented including information that is not available with standard text, such as metadata and information about the web pages that point to or are pointed to by a page under consideration. Once a collection of text or hypertext documents have been converted to a dataset in standard attribute/value form they can be processed by standard classification algorithms such as decision trees (Quinlan, 1986), support vector machines (Vapnik, 1995) etc. Information about these is readily available and they will not be described here.

2 Text Categorization

Text categorization (TC) dates back to the early 1960s, but it was not until the early 1990s that it gained popularity in the information systems area, due to the exponential growth of online text and also advances in computer hardware. TC is used in many application contexts, ranging from automatic document indexing based on a controlled vocabulary (Borko and Bernick 1963; Gray and Harley 1971; Field 1975), to document filtering (Amati and Crestani 1999; Iyer, Lewis et al. 2000; Kim, Hahn et al. 2000), word sense disambiguation (Gale, Church et al. 1992; Escudero, Marquez et al. 2000), population of hierarchical catalogues of Web resources (Chakrabarti, Dom et al. 1998; Attardi, Gulli et al. 1999; Oh, Myaeng et al. 2000), and in general any application requiring document organization or selective and adaptive document dispatching.

Until the late 1980s the most popular approach to TC was the knowledge engineering (KE) one, where a set of rules appropriate to the problem under consideration was manually created by encoding expert knowledge. Normally this approach requires a great human effort in defining the right rules. It is not flexible and a small change in the domain implies a large effort to modify the whole system. A system designed for a given domain is not adaptable for another domain and it must be redesigned from scratch with high costs in time and human effort.

From the early 1990s this approach has increasingly lost popularity, especially in the research community, in favour of the machine learning (ML) approach. In this framework, a general inductive process automatically builds an automatic text classifier by learning, from a set of previously classified documents, the characteristics of the categories of interest. The advantages of this approach are (i) classification accuracy comparable to that achieved by human experts, and (ii) a considerable saving in terms of expert manpower compared with the KE approach, since no intervention from either knowledge engineers or domain experts is needed for the construction of the classifier or for its porting to a different category set.

Developing a generic text classifier system based on machine learning (ML) techniques requires important decisions to be made about *text representation*: the way in which a document is represented so that it can be analyzed. Text representation comprises several phases:

- indexing (defines and extracts index terms that will be considered as features representing a given document)
- feature reduction (removes non-informative features from documents to improve categorization accuracy and reduce computational complexity)
- feature vector generation (represents each document as a weighted *feature vector* that can later be used to generate a text classifier or as input to the resulting classifier).

Each of these is discussed in Section 3 below.

Once the set of training documents is represented by a number of feature vectors, a machine learning algorithm can be applied to analyze it. At this stage, a *training set* of documents whose correct classifications are known is needed. The output of the learning phase is a model (called a *classifier*) that can be used later to classify new unseen documents.

The accuracy of the classifier is estimated by using it to classify a set of pre-labelled documents (called a *test set*) which are not used in the learning phase. The classifications produced are compared with those previously assigned to them (for example by human classifiers) which are treated as a gold standard.

2.1 Machine Learning Approaches to Text Categorization

Text classifiers developed using ML algorithms have been found to be cheaper and faster to build than ones developed by knowledge engineering, as well as more accurate in some applications (Sj and Waltz 1992). Nevertheless, ML algorithms applied to TC are challenged by many properties of text documents: a high dimensionality feature set, intrinsic linguistic properties (such as synonymy and ambiguity) and classification of documents into categories with few or no training examples.

A wide variety of learning approaches have been applied to TC, to name a few, Bayesian classification (Lewis and Ringuette 1994; Domingo and Pazzani 1996; Larkey and Croft 1996; Koller and Sahami 1997; Lewis 1998), decision trees (Weiss, Apte et al. ; Fuhr and Buckley 1991; Cohen and Hirsh 1998; Li and Jain 1998), decision rule classifiers such as CHARADE (Moulinier and Ganascia 1996), or DL-ESC (Li and Yamanishi 1999), or RIPPER (Cohen and Hirsh 1998), or SCAR (Moulinier, Raskinis et al. 1996), or SCAP-1 (Apté, Damerau et al. 1994), multi-linear regression models (Yang and Chute 1994; Yang and Liu 1999), Rocchio method (Hull 1994; Ittner, Lewis et al. 1995; Sable and

Hatzivassiloglou 2000), Neural Networks (Schütze, Hull et al. 1995; Wiener, Pedersen et al. 1995; Dagan, Karov et al. 1997; Ng, Goh et al. 1997; Lam and Lee 1999; Ruiz and Srinivasan 1999), example based classifiers (Creecy 1991; Masand, Linoff et al. 1992; Larkey 1999), support vector machines (Joachims 1998), Bayesian inference networks (Tzeras and Hartmann 1993; Wai and Fan 1997; Dumais, Platt et al. 1998), genetic algorithms (Masand 1994; Clack, Farringdon et al. 1997), and maximum entropy modelling (Manning and Schütze 1999).

2.2 Benchmarks

To establish which classification model is the most accurate, several TC benchmarks have been developed. The best-known one is probably the Reuters corpus[3]. It consists of a set of newswire stories classified under categories related to economics. The Reuters collection has been used for most of the experimental work in TC so far.

Other collections have also frequently been used such as the OHSUMED collection (Hersh, Buckley et al. 1994) used in (Joachims 1996; Baker and McCallum 1998; Lam and Ho 1998; Ruiz and Srinivasan 1999), the 20-newsgroups (Lang 1995) and in (Baker and McCallum 1998; Joachims 1998; McCallum and Nigam 1998; Nigam and Ghani 2000; Schapire and Singer 2000).

2.3 Text Categorization Compared with Traditional Classification Tasks

Text categorization uses several of the successful classification algorithms that have been developed for other 'traditional' classification areas. However, those algorithms have to overcome some challenges caused by the key characteristics of the TC area:

- High dimensional feature space. In text categorization, the input to the learning algorithm consists of a set of all the words occurring in all the training documents. A few thousand training documents can lead to tens of thousands of features. The dimensionality can be reduced using methods such as those described in Section 3.2, yet the resulting feature set will generally still be very large.
- Sparse document vectors. Even if there is a large set of features, each document contains only a small number of distinct words. This implies

[3] http://www.daviddlewis.com/resources/testcollections/reuters21578/

that document vectors are very sparse, i.e. only a few words occur with non-zero frequency.

- Heterogeneous use of terms. To cut down the size of the feature space, feature selection methods can be used to discard all the irrelevant features. However, in text classification this can lead to loss of information. Documents from the same category can consist of different words, since natural language allows the expression of related content with different formulation. There is generally not a small set of words that sufficiently describes all documents with respect to the classification task.
- High level of redundancy. In general, there are many different features relevant to the classification task and often several of those features occur in one document. This means that document vectors are redundant with respect to the classification task.
- Noise. Most natural language documents contain language mistakes. In machine learning, this can be interpreted as noise.
- Complex learning task. In text classification, the predefined classes are generally based on the semantic understanding of natural language by humans. Therefore, the learning algorithm needs to approximate such complex concepts.
- In 'conventional' classification problems studied by researchers in Machine Learning the classifications are usually mutually exclusive and exhaustive (very good, good, neutral, bad, very bad etc.). By contrast a text document can often belong to several categories, for example 'military history', 'crime' and 'second world war'. This is generally handled by converting a categorization problem with N categories into N separate binary yes/no categorization problems ('Is this document about military history yes/no?', 'Is it about crime yes/no?' etc.). Building N classifiers not just one obviously adds greatly to the processing time required.

3 Text Representation

3.1 Indexing

Text documents cannot be directly interpreted by a classifier. An indexing procedure needs to be applied to the dataset that maps each text document onto a compact representation of its content.

The choice of a representation for text depends on what one considers to be the meaningful textual units (the problem of lexical semantics) and the meaningful natural language rules for the combination of these units (the problem of compositional semantics).

A fundamental challenge in natural language processing and understanding is that information or meaning conveyed by language always depends on context. The same word might have different meanings. Also, a phrase may have different meanings depending on the context in which it is used. Consequently, trying to represent natural language documents by means of a set of index terms is a challenging task. Different linguistic approaches try to capture, or ignore, to varying degrees meaning with respect to context. These approaches can be divided into five levels:

- *Graphemic:* analysis on a sub-word level, commonly concerning letters.
- *Lexical:* analysis concerning individual words.
- *Syntactic:* analysis concerning the structure of sentences.
- *Semantic:* analysis related to the meaning of words and phrases.
- *Pragmatic:* analysis related to meaning regarding language-dependent and language-independent, e.g. application-specific, context.

The graphemic and lexical levels capture basically the frequencies of letter combinations or words in the documents. Text representation based on those approaches cannot deal with the meaning of documents, as there is a weak relationship between term occurrences and document content. On the other hand, the syntactic, semantic and pragmatic levels exploit more contextual information, such as structure of sentences, and lead to more complex text representation.

Choosing a suitable level of text analysis on which to base the definition of terms is a trade-off between semantic expressivity and representational complexity. A complex text representation leads to an increase in the dimension of the feature space, and with a limited number of training documents, inducing an accurate classifier is much harder. For this reason, in practice simple term definitions are dominant in text representation. The *n-grams* (overlapping and contiguous n letter subsequences of words) approach has often been used for indexing (De Heer 1982; Cavnar and Trenkle 1994; Tauritz, Kok et al. 2000). The advantage of n-grams is that the set of possible terms is fixed and known in advance (using only the 26 letters of the English alphabet and n=3, there are 26^3=17,576 distinct tri-grams). Furthermore, n-grams are language independent and are quite robust to both morphological and spelling variations and mistakes. N-grams are easy to calculate but the resulting representation is difficult to analyze by humans.

The most widely-used approach for indexing is the use of words, known as *tokens*. In this approach, the sequence in which the words appear in a document and any structure of the text are ignored. Effectively the document is treated as a *bag-of-words* (Lewis and Ringuette 1994). This term definition is language independent and computationally very efficient. However, a disadvantage is that each inflection of a word is a possible feature and thus the number of features can be unnecessarily very large. With a bag-of-words representation, we cannot tell (for instance) if the phrase “machine learning” exists within the document, or if the two words appeared unrelated to each other (Cohen and Singer 1998). Despite ignoring this information, classifiers using this representation perform well enough in comparison with ones that take word order into account.

Another approach is to use phrases, i.e. combining many words as one index, for example “artificial intelligence” or “data mining” (Fuhr and Buckley 1991; Tzeras and Hartmann 1993; Schütze, Hull et al. 1995). These indexes can be generated either manually or automatically. Because this process is highly domain dependent and considering all possible combinations of tokens is impossible, many algorithms exist to define phrasal indexes. Although some researchers have reported an improvement in classification accuracy when using such indexes (depending on the quality of the generated phrases), a number of experimental results (Lewis 1992; Apté, Damerau et al. 1994; Dumais, Platt et al. 1998) have not been uniformly encouraging, irrespective of whether the notion of “phrase” is motivated (i) *syntactically*, i.e. the phrase is such according to the grammar of the language (Lewis 1992); or (ii) *statistically*, i.e. the phrase is not grammatically such, but is composed of a set/sequence of words whose patterns of contiguous occurrence in the collection are statistically significant (Caropreso, Matwin et al. 2001).

(Lewis 1992) argues that the likely reason for those discouraging results is that, although indexing languages based on phrases have superior semantic qualities, they have inferior statistical qualities with respect to word-only indexing languages: a phrase-only indexing language has more terms, more synonymous or nearly synonymous terms, lower consistency of assignment (since synonymous terms are not assigned to the same documents), and lower document frequency for terms.

After tokens are extracted from files, an indexing phase follows. This consists of building a sequence of indexes based on those tokens. The way we use this set of indexes depends on whether the dictionary is already built or not, and therefore, whether the corresponding document is used for learning or for classification. When the dictionary is not yet constructed, the output of the tokenization step is merged to create a set of distinct indexes.

There are two ways in which the indexes can be chosen in order to build the dictionary. They can be selected to support classification under each category in turn, i.e. only those indexes that appear in documents in the specified category are used (the *local dictionary* approach). This means that the set of documents has a different feature representation (set of features) for each category. Alternatively, the indexes can be chosen to support classification under all categories, i.e. all indexes that appear in any of the documents in the training set are used (the *global dictionary* approach).

If the dictionary has already been created, then for each document under consideration the indexes are extracted and the ones that are not in the dictionary are omitted. This set of indexes is then used for document representation.

3.2 Feature Reduction

The main goal of the text classification task is to classify documents based on their contents. To do this, much of the auxiliary and collateral information inserted by writers to enrich the text exposition is not useful and on the contrary can make the decision more difficult. The system should analyze only the most informative words that can help to infer the topic of the document. Because of this, many auxiliary verbs, adverbs, conjunctions etc. are absolutely useless since they do not give a description of the topic. Also, they are often uniformly distributed over all topics and their informative contribution is uniformly spread over the classes.

A central problem in text classification using the machine learning paradigm is the high dimensionality of the feature space. There is one dimension for each unique index term found in the collection of documents and it is possible to have tens of thousands of different words occurring in a fairly small set of documents. Using all these words is time consuming and represents a serious obstacle for a learning algorithm. Standard classification techniques cannot deal with such a large feature set: not only does processing become extremely costly in computational terms, but the results become unreliable because of the lack of sufficient training data. However, many of the features are not really important for the learning task and their usage can degrade the system’s performance. There is an important need to be able to reduce the original feature set. There are two commonly used techniques of feature reduction: feature extraction and feature selection.

3.2.1 Feature Extraction

A frequent problem with using plain words as features is that the 'morphological variants' of a word will each be considered as a separate feature, for example computer, computers, computing and compute will be considered to be four different features, and this will frequently result in the number of features being unnecessarily large.

Feature Extraction is the process of extracting a set of new features from some of the original features (generally to replace them) through some functional mapping. Its drawback is that the generated new features may not have a clear physical meaning.

Stemming is among the widely used methods for feature extraction. It aims at conflating morphological variants of words by removing suffixes and mapping words to their base form. Mapping morphologically similar words into their stem can cut down the number of features substantially, for example ’compute’, ‘computing’, ‘computer’ and ‘computers’ might all be mapped to ‘comput’.

Many strategies for suffix stripping have been reported in the literature (Lovins 1968; Petrarca and Lay 1969; Andrews 1971; Dattola 1979; Porter 1980). The most widely used stemming tools for the English language are the Porter stemmer (Porter 1980) and the word-net morphological processor (Miller, Princeton et al. 1998). Other methods include term clustering techniques (Baker and McCallum 1998; Li and Jain 1998; Slonim and Tishby 2001) and latent semantic indexing (Deerwester, Dumais et al. 1990; Schütze, Hull et al. 1995; Wiener, Pedersen et al. 1995).

3.2.2 Feature Selection

Feature Selection is the process of choosing a subset from the original feature set according to some criteria. The selected feature retains original physical meaning and provides a better understanding for the data and learning process. However, in removing terms, the risk is of removing potentially useful information on the meaning of documents, so the reduction process needs to be performed with care.

Stop Words

One of the most frequently used methods of reducing the number of different features in the dictionary is to remove very common words known as *stop words*. These are high frequency words that are likely to exist within all or nearly all documents of the collection regardless of their classifications. In the case of phrasal indexes, stop words should be eliminated before dictionary building.

Articles, prepositions and common verbs (i.e. the, from, do etc.) that provide structure in the language rather than content, are usually considered to be stop words.

a
about
all
an
and
are
as
at
be
but
by
did
do
down
during
each
else
for
from
further
get
if
in
is
it
into
itself
just
last
late
like
many
much
more
must
near
no
of
often
on
once
or
that
the
they
to
with
which
whether

Figure 1. Some Frequently Used Stop Words

A list of 512-predefined English stop words can be found in (Lewis 1992). There is also the possibility of constructing a domain-dependent-list.

Using a Thesaurus

Another important objective in the context of feature reduction is to conflate synonyms, i.e. words of the same or similar meaning. A *thesaurus* is a collection of words that are grouped by likenesses in their meaning rather than in alphabetical order. Mapping all the index terms onto the equivalent class to which they belong can thus reduce the set of features.

Another common way of using a thesaurus is to expand rather than conflate the set of features, by adding semantically related words to the set. A thesaurus is domain- and language dependent and manually constructed thesauri are thus often used. However, there are also attempts to automatically construct thesauri based on document collections.

Document Frequency Thresholding

Another effective feature selection method is *document frequency thresholding*. Document frequency is defined as the number of documents in which a term occurs. The document frequency is computed for each unique term in the training dataset and those whose frequency is less than a predefined threshold are removed from the feature set. The basic assumption is that rare terms are either non-informative for category prediction or not influential in global performance. In either case, removal of rare terms reduces dimensionality of the feature space. Improvement in categorization accuracy is also possible if rare terms happen to be noise, e.g. spelling mistakes.

(Luhn 1958) suggested that the most relevant words belong to the intermediate interval of frequency and the irrelevant ones are out from this range.

Inverse Document Frequency and TFIDF

A significant problem with the Document Frequency Thresholding approach is that although it is true that rarely occurring terms are unlikely to be relevant for characterizing a particular class, terms that occur in a large portion of the document collection may not be discriminative either. The *term frequency/inverse document frequency (TFIDF)* method assumes that (common stop words having been removed) the importance of a term increases with its use frequency but is inversely proportional to the number of documents in which it appears. For a term *t*, the former is denoted by *tf(t)*, the frequency of term *t* in all documents. A possible measure for the latter is the *inverse document frequency* for term t, which is defined as

$$idf(t) = log\ (n / n(t))$$

where *n* is the number of documents in the training set and *n(t)* is the number of documents where term *t* occurs.

This leads to the *term frequency/inverse document frequency (tfidf)* measure:
*tfidf(t) = tf (t) * idf(t)*

The more important terms are those assigned higher *tfidf* values. Note that the evaluation of this measure does not depend on the class labels of the training documents.

Other more sophisticated information-theoretic functions, which use a term-goodness criterion threshold to decide about eliminating a feature, have been used in the literature. Among these are the DIA association factor (Fuhr and Buckley 1991), chi-square (Yang and Pedersen 1997; Sebastiani, Sperduti et al. 2000; Caropreso, Matwin et al. 2001), NGL coefficient (Ng, Goh et al. 1997; Ruiz and Srinivasan 1999), information gain (Lewis 1992; Lewis and Ringuette 1994; Moulinier, Raskinis et al. 1996; Yang and Pedersen 1997; Larkey 1998; Mladenic and Grobelnik 1998; Caropreso, Matwin et al. 2001), mutual information (Larkey and Croft 1996; Wai and Fan 1997; Dumais, Platt et al. 1998; Taira and Haruno 1999) odds ratio (Mladenic and Grobelnik 1998; Ruiz and Srinivasan 1999; Caropreso, Matwin et al. 2001), relevancy score (Wiener, Pedersen et al. 1995) and GSS coefficient (Galavotti, Sebastiani et al. 2000). Three of the most popular methods are descrivbed briefly below. They all make use of the class labels of the training documents.

Information gain (IG)

Information gain is widely used in the field of machine learning. It measures the number of bits of information obtained for category prediction by knowing the presence or absence of a term in a document. The information gain of term t is defined to be:

$$G(t) = -\sum_{i=1}^{m} \Pr(c_i)\log\Pr(c_i) + \Pr(t)\sum_{i=1}^{m} \Pr(c_i|t)\log\Pr(c_i|t) + \Pr(\bar{t})\sum_{i=1}^{m} \Pr(c_i|\bar{t})\log\Pr(c_i|\bar{t})$$

where:

Pr(c$_i$) is the probability of having class c_i

Pr(t) is the probability of having term *t*

Pr(c_i | t) is the probability of having class c_i given that the term *t* is observed in the document.

Pr(c_i | $\bar{t}$) is the probability of having class c_i given that term *t* is not observed in the document.

Mutual information (MI)

Mutual information is a criterion commonly used in statistical language modelling of word associations and related applications. If one considers the two-way contingency table of a term *t* and a category *c*, where *A* is the number of times *t* and *c* co-occur, *B* is the number of time *t* occurs without *c*, *C* is number of times *c* occurs without *t*, and *N* is the total number of documents, then mutual information criterion between *t* and *c* is defined to be:

$$I(t,c) = \log\frac{\Pr(t \wedge c)}{\Pr(t) * \Pr(c)}$$

and is estimated using:

$$I(c,t) \approx \log\frac{A * N}{(A+C)+(A+B)}$$

I(t,c) has a natural value of zero if *t* and *c* are independent. To measure the goodness of a term in a global feature selection, we combine the category specific scores of a term in two ways:

$$I_{avg}(t) = \sum_{i=1}^{m} \Pr(c_i) I(t,c_i)$$

$$I_{\max}(t) = \max_{i=1}^{m}\{I(t,c_i)$$

A weakness of mutual information is that the score is strongly influenced by the marginal probabilities of terms, as can be seen in this equivalent form:

$$I(c,t) = \log\Pr(t \mid c) - \log\Pr(t)$$

For terms with an equal conditional probability *Pr(t|c)*, rare terms will have a higher score than common terms. The scores, therefore, are not comparable across terms of widely differing frequency.

χ^2 statistic (CHI)

The χ^2 statistic measures the lack of independence between t and c and can be compared with the χ^2 distribution with one degree of freedom to judge extremeness. Using the two-way contingency table of a term *t* and a category *c*, where A is the number of times *t* and *c* co-occur, *B* is the number of times *t* occurs without *c*, *C* is the number of times *c* occurs without *t*, *D* is the number of times neither *c* nor *t* occurs, and *N* is the total number of documents, the term goodness measure is defined to be:

$$\chi^2(t,c) = \frac{N * (AD - CB)^2}{(A+C) * (B+D) * (A+B) * (C+D)}$$

The χ^2 statistic has a natural value of zero if *t* and *c* are independent. We compute for each category the χ^2 statistic between each unique term in a training dataset and that category, and then combine the category specific scores of each term into two scores:

$$\chi^2_{avg}(t) = \sum_{i=1}^{m} \Pr(c_i)\chi^2(t,c_i)$$

$$\chi^2_{\max}(t) = \max_{i=1}^{m}\{\chi^2(t,c_i)\}$$

A major difference between χ^2 and MI is that χ^2 is a normalized value, and hence χ^2 values are comparable across terms for the same category. However, this normalization breaks down if any cell in the contingency table is lightly populated, which is the case for low frequency terms. Therefore, the χ^2 statistic is known not to be reliable for low frequency terms.

Experimental Comparisons

Various experimental comparisons of feature selection functions applied to TC contexts have been carried out (Yang and Pedersen 1997; Mladenic and Grobelnik 1998; Galavotti, Sebastiani et al. 2000; Caropreso, Matwin et al. 2001). In these experiments most functions have improved on the results for basic document frequency thresholding. For instance, (Yang and Pedersen 1997) have shown that, with various classifiers and various initial corpora, sophisticated techniques such as information gain can reduce the dimensionality of the term space by a factor of 100 with no loss (or even with a small increase) of effectiveness.

3.2.3 Combined Methods

Techniques for reducing the size of the dictionary are usually used in combination. It is very common first to eliminate stop words, then to convert the remaining terms to their stem roots. After that, rarely occurring terms are removed, by thresholding either their term frequency or their document frequency. Finally, a more elaborate method such as information gain can be used further to reduce the number of features.

3.3 Feature Vector Generation

The most popular text representation form is the *Vector Space Model* (Salton, Wong et al. 1975). The task of the vector generation step is to create a weighted vector $d = \left(w(d,t_1),...,w(d,t_m)\right)^T$ for any document *d*, where each weight $w(d,t_i)$ expresses the importance of term t_i in document *d*.

After a vocabulary (list of all index terms appearing in the training documents) *V* is built, it defines a $||V|$-dimensional vector space with the documents represented as vectors in this space. The value of the jth component of the ith vector is the weight of the jth index term in the ith document. The weights in the vector are determined by a weight function. Different weighting schemes exist, including:

- binary weighting has been used in (Apté, Damerau et al. 1994; Lewis and Ringuette 1994; Moulinier, Raskinis et al. 1996; Koller and Sahami 1997; Sebastiani, Sperduti et al. 2000) especially because of the symbolic, non-numeric nature of the learning systems
- term frequency
- TFIDF (McGill and Salton 1983)
- TFC (Salton and Buckley 1988)

- ITC (Buckley, Salton et al. 1995)
- and entropy weighting (Dumais 1991).

Some of the most commonly used methods are outlined below. Usually, the resulting vector is very sparse as most of the documents contain about 1 to 5% of the total number of terms in the vocabulary.

Binary Representation
One simple and common representation is the binary representation, where the appearance of an index is indicated with a 1 in the document vector representation. All non-present words have a weight of 0.

Term Frequency
Term frequency captures the number of occurrences of a term/index within a given document.

TFIDF Representation
TFIDF representation is an information-retrieval-style indexing technique that is widely used in text representation (a variant of it was used in Section 3.2.2).

$$(TF\_IDF)_{ij} = TF_{ij} * IDF_i \text{ where } IDF_i = \log_2 (n / DF_j)$$

TF_{ij} is the number of times term t_j occurs in document d_i
DF_j is the number of training documents in which word t_j occurs at least once.
n is the total number of training documents.

This weighting function encodes the intuitions that the more often a term occurs in a document, the more it is representative of the document's content, and that the more documents in which a term occurs, the less discriminating it is.

In order to make weights fall in the *[0,1]* interval and for documents to be represented by vectors of equal length, the weights resulting from the function $(TF\_IDF)_{ij}$ are normalized by 'cosine normalization', given by:

$$X_{ij} = \frac{(TF\_IDF)_{ij}}{\sqrt{\sum_{j=1}^{N} (TF\_IDF)_{ij}^2}}, \ 1 \leq i \leq n \text{ and } 1 \leq j \leq N$$

where N is the number of terms that occur at least once in the whole set of training documents.

4 Hypertext Categorization

It has been estimated that the World Wide Web comprises more than 9 billion pages (www.google.com). As long ago as 1998/9 it was estimated to be growing at a rate of 1.5 million pages a day (Bharat and Broder 1998; Lawrence and Giles 1999). Faced with such a huge volume of documents, search engines become limited: too much information to look at and too much information retrieved. The organization of web documents into categories will reduce the search space of search engines and improve their retrieval performance. A study by (Chen and Dumais 2000) showed that users prefer to navigate through directories of pre-classified content and that providing a categorized view of retrieved documents enables them to find more relevant information in a shorter time. The common use of the manually constructed category hierarchies for navigation support in Yahoo (www.yahoo.com) and other major web portals has also demonstrated the potential value of automating the process of hypertext categorization.

Text classification is a relatively mature area where many algorithms have been developed and many experiments conducted. For example, classification accuracy reached 87% (Chakrabarti, Dom et al. 1997) for some algorithms applied to known text categorization corpora (Reuters, 20-newsgroups etc.) where the vocabulary is coherent and the authorship is high. However, those same classifiers often perform badly on hypertext datasets. Hypertext classification poses new classification challenges in addition to those of text classification.

- Diversity of the web content: Web documents are diverse. They range from home pages, articles, tutorials etc. to portals.
- Diversity of authorship: the web is open to everybody. Millions of authors can put their scripts into the web. This leads to little consistency in the vocabulary.
- Sparse or non-existing text: Many web pages only contain a limited amount of text. In fact, many pages contain only images and no machine readable text at all. The proverb “one picture is worth a thousand words” is widely applied by web authors.

Several variations of the classical classification algorithms have been developed to meet the particularities of the hypertext/text classification task.

As well as the above, automated hypertext categorization poses new research challenges because of the extra information in a hypertext document. Hyperlinks, HTML tags, metadata and linked neighbourhood all provide rich information for classifying hypertext that is not available in traditional text categorization.

Researchers have only recently begun to explore the issues of exploiting rich hypertext information for automated categorization. There is now a growing volume of research in the area of learning over web text documents. Since most of the documents considered are in HTML format, researchers have taken advantage of the structure of those pages in the learning process. The systems generated differ in performance because of the quantity and nature of the additional information considered.

(Benbrahim and Bramer 2004a) used the BankSearch dataset (Sinka, M. P. and D. W. Corne, 2002) to study the impact on classification of the use of metadata (page keywords and description), page title and link anchors in a web page. They concluded that the use of basic text content enhanced with weighted extra information (metadata + title + link anchors) improves the performance of three different classifiers. In (Benbrahim and Bramer 2004b), they used the same dataset to investigate the influence of the neighbourhood pages (incoming and outgoing pages of the target document) on classification accuracy. It was concluded that the intelligent use of this information helps improve the accuracy of the different classifiers used.

(Oh, Myaeng et al. 2000) reported some observations on a collection of online Korean encyclopaedia articles. They used system-predicted categories of the linked neighbours of a test document to reinforce the classification decision on that document and they obtained a 13% improvement over the baseline performance when using local text alone.

(Furnkranz 1999) used a set of web pages from the Web->KB corpus, created by the Carnegie-Mellon University World Wide Knowledge Base Project[4], to study the use of anchor text and the words near the anchor text in a web page to predict the class of the target page pointed to by the links. By representing the target page using the anchor words on all the links that point to it, plus the headlines that structurally precede the sections where links occur, the classification accuracy of a rule-learning system improved by 20%, compared with the baseline performance of the same system when using the local words in the target page instead.

(Slattery and Mitchell 2000) used the Web->KB university corpus, but studied alternative learning paradigms, namely, a First Order Inductive Learner which exploits the relational structure among web pages, and a Hubs and Authorities style algorithm exploiting the hyperlink topology. They found that a combined use of these two algorithms performed better than using each alone.

(Yang, Slattery et al. 2002) have defined five hypertext regularities which may hold in a particular application domain, and whose presence may significantly influence the optimal design of a classifier. The experiments were carried out on 3 datasets and 3 learning algorithms. The results showed that the naïve use of the linked pages can be more harmful than helpful when the neighbourhood is noisy, and that the use of metadata when available improves classification accuracy.

(Attardi, Gulli et al. 1999) described an approach that exploits contextual information extracted from an analysis of the HTML structure of Web documents as well as the topology of the web. The results of the experiments with a categorization prototype tool were quite encouraging.

[4] http://www.cs.cmu.edu/~webkb/

(Chakrabarti, Dom et al. 2002) studied the use of citations in the classification of IBM patents where the citations between documents were considered as hyperlinks, and the categories were defined on a topical hierarchy. Similar experiments on a small set of pages with real hyperlinks were also conducted. By using the system-predicted category labels for the linked neighbours of a test document to reinforce the category decision on that document, they obtained a 31% error reduction, compared with the baseline performance when using the linked documents, treating the words in the linked documents as if they were local.

5 Hypertext Representation

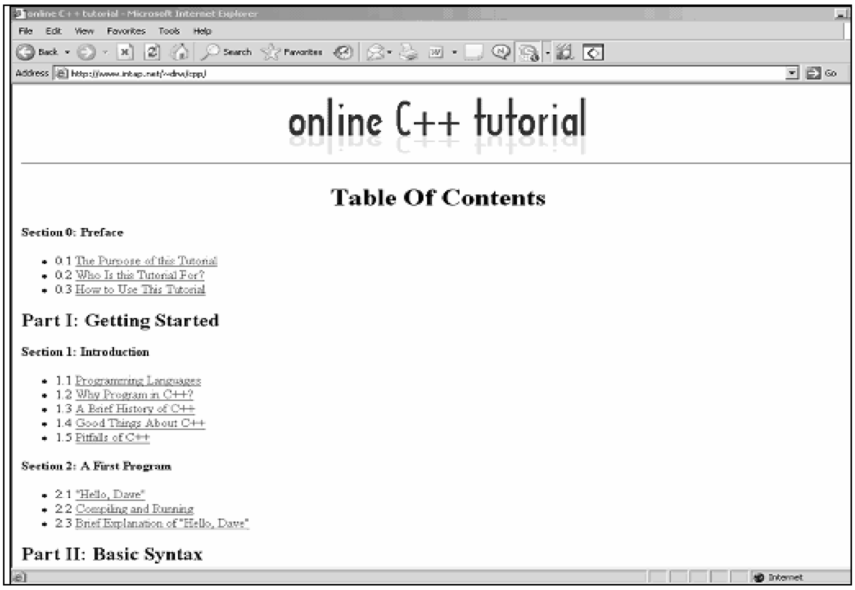
Before machine-learning methods can be applied to hypertext categorization, a decision first needs to be made about a suitable representation of HTML pages. Hypertext representation inherits all the basics and steps of the traditional text representation task described in Section 3. In addition, it takes advantage of the hidden information in HTML pages. Hyperlinks, HTML tags, metadata and information in a page's neighbours (documents pointing to and pointed to by the target page) are used to enrich the HTML pages' representation.

Vector Space Model (VSM) representations identify each document by a feature vector in a space in which each dimension corresponds to a distinct index term (feature). The set of features can be generated either manually or automatically based on a specific document collection, usually the set of training documents used as input for the learning algorithm. A given document vector has, in each component, a numerical value to indicate its importance. This value is expressed as a function of the frequency of the index term in the particular document. By varying this function, we can produce different term weightings. The resulting representation of the text is equivalent to the attribute-value representation, which is commonly used in machine learning.

There are several steps involved in transforming HTML-pages into feature vectors. Tokenization transforms a web-page into a sequence of word tokens that are used to build indexes during the indexation phase. There is an important difference between Hypertext Categorization and Text Categorization at the tokenization stage. In general, a hypertext dataset consists of a set of HTML source files, not plain text files. The aim of the tokenization step is the extraction of plain tokens (each a sequence of characters separated by one or more spaces) from each HTML page. After this, further normalization is applied to the set of tokens. Usually all letters are converted to lower-case as it is expected that case gives no information about the subject matter. Then HTML tags, scripts, punctuation, special characters and digits are removed. Tokens that contain numeric or non-alphanumeric characters are often omitted, even though sometimes they should not be. For example the token "c++" might indicate a programming language. Such a normalization step would ideally make use of both domain- and application-specific knowledge.

Following indexation, if training documents are being processed all distinct index terms are merged to generate a set of potential features that might be used in the dictionary. Usually, the large size of the dictionary is reduced at the dimensionality reduction step. If instead a new document is to be processed with an existing classifier and thus the dictionary is already built, the indexation task outputs only index terms which exist in the dictionary. Finally, a vector generation step calculates weights for all the index terms of any given HTML page.

Figures 2, 3 and 4 show an HTML document as we see it in an internet browser, part of the corresponding HTML source code and an example of what it might look like after pre-processing, respectively.


online C++ tutorial - Microsoft Internet Explorer
File
Edit
View
Favorites
Tools
Help
Back
Search
Favorites
Address
http://www.intap.net/~drw/cpp/
Go
Internet

online C++ tutorial

Table Of Contents

Section 0: Preface

- 0.1 The Purpose of this Tutorial
- 0.2 Who Is this Tutorial For?
- 0.3 How to Use This Tutorial

Part I: Getting Started

Section 1: Introduction

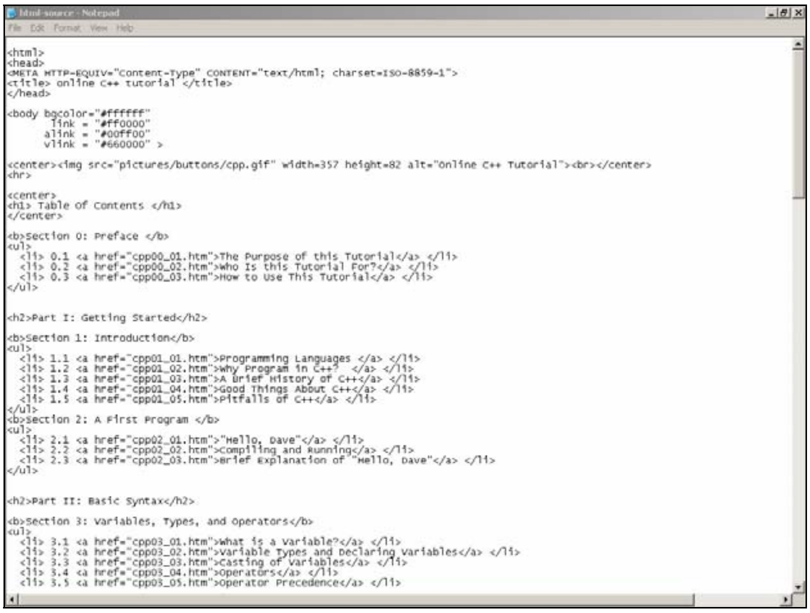
- 1.1 Programming Languages
- 1.2 Why Program in C++?
- 1.3 A Brief History of C++
- 1.4 Good Things About C++
- 1.5 Pitfalls of C++

Section 2: A First Program

- 2.1 "Hello, Dave"
- 2.2 Compiling and Running
- 2.3 Brief Explanation of "Hello, Dave"

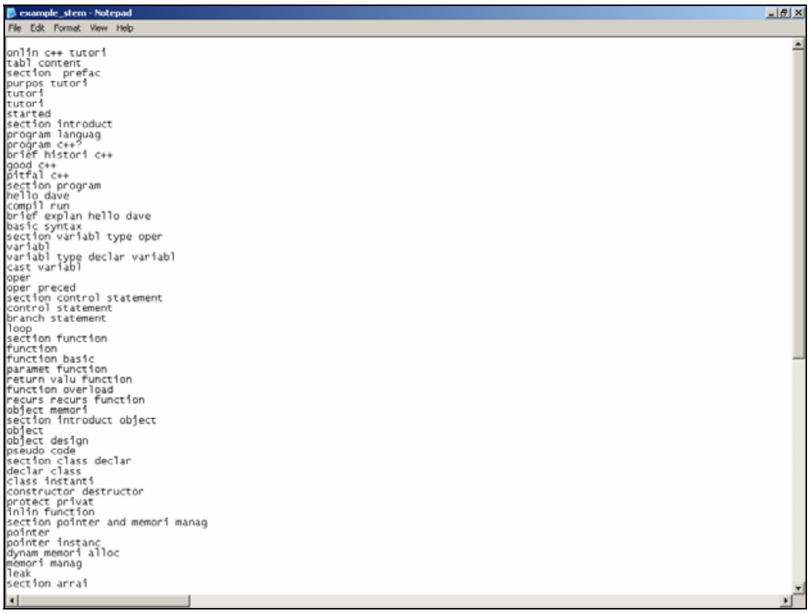
Part II: Basic Syntax

Figure 2. An HTML Document As We See It In An Internet Browser



```
<html>
<head>
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=ISO-8859-1">
<title> online C++ tutorial </title>
</head>

<body bgcolor="#ffffff"
        link = "#ff0000"
      alink = "#00ff00"
      vlink = "#660000" >

<center><img src="pictures/buttons/cpp.gif" width=357 height=82 alt="Online C++ Tutorial"><br></center>
<hr>

<center>
<h1> Table of Contents </h1>
</center>

<b>Section 0: Preface </b>
<ul>
  <li> 0.1 <a href="cpp00_01.htm">The Purpose of this Tutorial</a> </li>
  <li> 0.2 <a href="cpp00_02.htm">Who Is this Tutorial For?</a> </li>
  <li> 0.3 <a href="cpp00_03.htm">How to Use This Tutorial</a> </li>
</ul>


<h2>Part I: Getting Started</h2>

<b>Section 1: Introduction</b>
<ul>
  <li> 1.1 <a href="cpp01_01.htm">Programming Languages </a> </li>
  <li> 1.2 <a href="cpp01_02.htm">Why Program in C++?  </a> </li>
  <li> 1.3 <a href="cpp01_03.htm">A Brief History of C++</a> </li>
  <li> 1.4 <a href="cpp01_04.htm">Good Things About C++</a> </li>
  <li> 1.5 <a href="cpp01_05.htm">Pitfalls of C++</a> </li>
</ul>
<b>Section 2: A First Program </b>
<ul>
  <li> 2.1 <a href="cpp02_01.htm">"Hello, Dave"</a> </li>
  <li> 2.2 <a href="cpp02_02.htm">Compiling and Running</a> </li>
  <li> 2.3 <a href="cpp02_03.htm">Brief Explanation of "Hello, Dave"</a> </li>
</ul>


<h2>Part II: Basic Syntax</h2>

<b>Section 3: Variables, Types, and Operators</b>
<ul>
  <li> 3.1 <a href="cpp03_01.htm">What is a Variable?</a> </li>
  <li> 3.2 <a href="cpp03_02.htm">Variable Types and Declaring Variables</a> </li>
  <li> 3.3 <a href="cpp03_03.htm">Casting of Variables</a> </li>
  <li> 3.4 <a href="cpp03_04.htm">Operators</a> </li>
  <li> 3.5 <a href="cpp03_05.htm">Operator Precedence</a> </li>
```

Figure 3. An Example Source Code Of An HTML Document



```
onlin c++ tutori
tabl content
section  prefac
purpos tutori
tutori
tutori
started
section introduct
program languag
program c++?
brief histori c++
good c++
pitfal c++
section program
hello dave
compil run
brief explan hello dave
basic syntax
section variabl type oper
variabl
variabl type declar variabl
cast variabl
oper
oper preced
section control statement
control statement
branch statement
loop
section function
function
function basic
paramet function
return valu function
function overload
recurs recurs function
object memori
section introduct object
object
object design
pseudo code
section class declar
declar class
class instanti
constructor destructor
protect privat
inlin function
section pointer and memori manag
pointer
pointer instanc
dynam memori alloc
memori manag
leak
section arrai
```

Figure 4.An Example Document After Pre-processing

5.1 Enriched Hypertext Representation

Web pages contain additional information such as metadata which it seems reasonable to suppose might be usable to improve categorization accuracy. This section addresses the open question of which extra information hidden in the HTML pages should be included, and how to include it. This affects both the tokenization and the vector generation phases.

An HTML document is much more than a simple text file. It is structured and connected with other HTML pages. The simplest way to represent a web page is to extract the text found in the BODY tag, but this representation does not exploit the rich information hidden in the hypertext document.

This extra information can be modelled with varying levels of detail, and might be divided into two major subsets. One subset includes information that is local to the HTML web document, mainly the information enclosed in the structure of the page itself such as the data in the META tag and TITLE tag, and the other contains the extra information present in the hyperlinks, i.e. information included in the page’s neighbours (documents pointing to, and pointed to by the target page).

5.1.1 Which Extra Information?

HTML structure data

The basic structure of HTML pages has changed with HTML versions. At the beginning, HTML pages were quite simple, but as HTML has been extended, pages have taken a more subtle but still simple structure.

An ordinary HTML page has two main parts: the first line and the rest of the document (delimited by the **<HTML>** and **</HTML>** tags), which is also split into two other parts: the heading and the body of the page.

The very first line of an HTML page (<!DOCTYPE>) indicates to the browser which HTML version is used for this document.

The HTML page's heading, which follows immediately the <!DOCTYPE...> tag, is delimited by the <HEAD> and </HEAD> tags. This part contains the so called ***meta*** information, i.e. information about the content of the HTML page. For instance, one indicates information about the author of the page, the date of creation, keywords or a description of the content of the page, the page's title, and possibly other elements, like style sheets or JavaScript code. This part contains information that is not directly displayed in the browser.

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Home | University of Portsmouth</title>
```

<meta name="dc.keywords" content="University of Portsmouth, University, Portsmouth, degree, courses, research, postgraduate, undergraduate, study, studying" />
<meta name="dc.description" content="Welcome to the University of Portsmouth, UK. Find out about studying in Portsmouth, our courses and research activities. " />
</head>

The second part contains the content of the web page, i.e. what should be displayed by the web browser. This part is delimited by the <BODY> and </BODY> tags. HTML tags (other than <HEAD>, <BODY>, <HTML> and <META>) are exclusively used in this part to format the document.

Hyperlink Data

Another source of extra information in HTML pages is the use of data in the linked neighbourhood (incoming and outgoing documents of the target page). This data can be put under two type levels, either local or remote hyperlink data.

The local hyperlink data (i.e. anchor data) can be captured from the content of the <A> tag in the target page. An example of its use:

<A href = "./VB.html"> Visual Basic tutorial </A>.

In this example, the A tag is used to link the current page to another one. Users can have an idea about the content of the linked page by means of the content of the A tag, that is "Visual Basic tutorial". Usually, when a developer makes a link in his page, he tries to explain with few words the content of the linked page in the A tag. Therefore, it is assumed that the words used in this description are close to the subject of the linked page, and hence can be given special interest as they can represent the content of the linked page.

The remote hyperlink data might also be split into two subsets, namely, the set of links that the target page points to, and the set of links that point to the target page. To have access to this data, one should download the content of the linked pages.

5.1.2 How Much Extra Information?

Data in Meta-Description, Meta-Keywords and Title tags is generally considered more important and representative of a document's content than a word present in the BODY tag. Unfortunately, in a quantitative study conducted on a sample of web pages in (Pierre 2000), it was stated that although TITLE tags are common in HTML pages, their corresponding amount of text is relatively small with 89% of the titles containing only 1-10 words. Also, the titles often contain only names or terms such as ``home page", which are not particularly helpful for subject classification. On the other hand, Metatags for keywords and descriptions are found in only about a third of web sites. Most of the time these metatags contain between 11 and 50 words, with a smaller percentage containing more than 50 words (in contrast to the number of words in the body text which tended to contain more than 50 words).

There are other structured data in the BODY of HTML pages that might be taken into special consideration as well, such as the data in the heading tag, i.e. data enclosed in <Hi> and </Hi>, or boldfaced data i.e. data between <b> and </b>.

The use of links' content in hypertext representation might be straightforward, i.e. concatenate it to the target page. However, this blind use may harm the classification task, due to the fact that many pages point to or are pointed to by pages from different subjects, e.g., web pages of extremely diverse topics link to Yahoo! or BBC news web pages. Many techniques have been used to filter out the noisy link information. There are many different ways to measure how similar two documents are, including Euclidian distance, Cosine measure and Jaccard measure. The cosine measure is a very common similarity measure for textual datasets. It is given by:

$$\cos\_sim(D_A, D_B) = \frac{D_A . D_B}{\|D_A\|_2 . \|D_B\|_2}$$

$$= \frac{\sum_{i=1}^{n} w_{iA} . w_{iB}}{\sqrt{\sum_{i=1}^{n} w_{iA}^2} \sqrt{\sum_{i=1}^{n} w_{iB}^2}}$$

where w_{iA} represents the weight of index i in document A.

5.1.3 How to Include This Extra Information

Structure Term Weighting

To exploit the structural information in HTML pages, one should consider not only the frequency of the term in the document, but also the HTML tag the term is present in. The idea is to assign greater importance to terms that belong to tags that are believed to be more representative for web pages (META and TITLE tags). The term frequency of words might then be recomputed as an enhanced term frequency as follows:

$$ETF(t_i, D_j) = \sum_{tag_k} \left(w(tag_k).TF(t_i, tag_k, D_j) \right)$$

Where tag_k is an HTML tag, $w(tag_k)$ is the weight we assign to tag_k and $TF(t_i, tag_k, D_j)$ is the number of times the term t_i is present in the tag tag_k of the HTML document D_j.

w(tag) can be defined as a function as:

$$w(tag) = \begin{cases} \alpha \ \text{if tag} = \text{META} \\ \beta \ \text{if tag} = \text{TITLE} \\ 1 \ \text{if somewhere else} \end{cases}$$

Linked Term Weighting

There are many ways to incorporate the ‘similar’ linked neighbourhood in the target web document representation. The most straightforward method is to concatenate the content of the similar linked pages to the target web page. A new dictionary is then built to take into consideration the new indexes present in the neighbourhood, and that do not exist in the target pages.

A new enhanced term frequency scheme might be defined as:

$$ETF(t_i, D_j) = \sum_{source_l} \sum_{tag_k} \left(w_s(source_l) w_t(tag_k) . TF(t_i, tag_k, source_l, D_j) \right)$$

Where tag_k is an HTML tag, $w_t(tag_k)$ is the weight we assign to tag_k , $source_l$ is the source of the term, i.e. term exists in target page, or a neighbour page, $w_s(source_l)$ is the weight we assign to $source_l$, and $TF(t_i, tag_k, source_l, D_j)$ is the number of times term t_i is present in the tag tag_k of the $source_l$ related to HTML document D_j.

$$w_t(tag) = \begin{cases} \alpha \ \text{if tag} = \text{META} \\ \beta \ \text{if tag} = \text{TITLE} \\ 1 \ \text{if somewhere else} \end{cases}$$

$$w_s(source) = \begin{cases} \delta \ \text{if source} = \text{incoming link} \\ \lambda \ \text{if source} = \text{outgoing link} \\ 1 \ \text{if source} = \text{target web page} \end{cases}$$

The drawback of this approach is that the resulting set of total features (size of the hypertext dictionary) is inflated.

The other approach is a variation of the previous one. In this case, no new dictionary is built, i.e. just terms already existing in the original HTML page are considered, and no extra terms from the neighbourhood are added. The new term frequencies are recomputed using the same above formula.

Another approach is to distinguish the terms absorbed from the neighbours from the original local terms by adding a prefix, so that the classifier can distinguish between the local and non-local terms. The disadvantage of this method is the term is split into many forms which can make it relatively rare, even if the corresponding original term is not that rare. It also explodes the size of the new dictionary, and hence, the classifier faces a challenge as the number of features increases while the number of documents remains the same.

5.2 Experiments with FSS-SVM

With the techniques described above, it is possible to map any HTML document into its vector representation, which is then suitable as input for any of a range of standard learning algorithms, such as decision trees (Quinlan, 1986) and support vector machines (Vapnik, 1995). A recent example of the use of a new form of Fuzzy Semi-supervised Support Vector Machine (FSS-SVM) to classify web pages with "enriched' representation is given in (Benbrahim and Bramer, 2008).

In all experiments, the fuzzy semi-supervised support machine performed better than its supervised version when the number of labelled training documents is small, i.e. FSS-SVM can achieve a specific level of classification accuracy with much less labelled training data. For example, with only 550 labelled training examples for the BankSearch dataset (50 documents per class), FSS-SVM reached 65% classification accuracy, using the F1 measure of accuracy (Bramer, 2007), while the traditional SVM classifier achieved only 50%. To reach 0.65 classification accuracy, SVM required about 1100 labelled training documents and FSS-SVM only 550.

For the Web->KB dataset, the performance increase was also substantial. For 80 labelled training examples (20 documents per class), SVM obtained 29% accuracy (F1 measure) and FSS-SVM 59%, reducing classification error by 30%. (Benbrahim and Bramer, 2008) also reports other experimental findings and gives full details of the fuzzy semi-supervised support vector machine algorithm and the enriched representation of hypertext documents used.

References

Amati, G. and F. Crestani (1999). "Probabilistic learning for selective dissemination of information." Information Processing and Management 35(5): 633-654.

Andrews, K. (1971). The development of a fast conflation algorithm for English. Dissertation submitted for the Diploma in Computer Science, University of Cambridge (unpublished).

Apté, C., F. Damerau, et al. (1994). "Automated learning of decision rules for text categorization". ACM Transactions on Information Systems (TOIS) 12(3): 233-251.

Attardi, G., A. Gulli, et al. (1999). "Automatic Web Page Categorization by Link and Context Analysis." Proceedings of THAI'99 99: 105-119.

Baker, L. D. and A. K. McCallum (1998). "Distributional clustering of words for text classification." Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval: 96-103.

Benbrahim, H. and M. Bramer (2004a). "An empirical study for hypertext categorization." Systems, Man and Cybernetics, 2004 IEEE International Conference on 6.

Benbrahim, H. and M. Bramer (2004b). "Neighbourhood Exploitation in Hypertext Categorization." In Proceedings of the Twenty-fourth SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence, Cambridge, December 2004, pp. 258-268. ISBN 1-85233-907-1

Benbrahim, H. and M. Bramer (2008). A Fuzzy Semi-Supervised Support Vector Machines Approach to Hypertext Categorization In: Artificial Intelligence in Theory and Practice II, Springer, pp. 97-106.

Bharat, K. and A. Z. Broder (1998). "A Technique for Measuring the Relative Size and Overlap of Public Web Search Engines." WWW7 / Computer Networks 30(1-7): 379-388.

Borko, H. and M. Bernick (1963). "Automatic Document Classification." Journal of the ACM (JACM) 10(2): 151-162.

Bramer, M.A. (2007). Principles of Data Mining. Springer-Verlag.

Buckley, C., G. Salton, et al. (1995). "Automatic query expansion using SMART: TREC 3." Overview of the Third Text Retrieval Conference (TREC-3): 500-225.

Caropreso, M. F., S. Matwin, et al. (2001). "A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization." Text Databases and Document Management: Theory and Practice: 78–102.

Cavnar, W. B. and J. M. Trenkle (1994). "N-Gram based document categorization." See: Proceedings of the Third Symposium on Document Analysis and Information Retrieval. LasVegas: 161-176.

Chakrabarti, S., B. Dom, et al. (1997). "Using taxonomy, discriminants, and signatures for navigating in text databases." Proceedings of the 23rd VLDB Conference: 446-455.

Chakrabarti, S., B. Dom, et al. (1998). "Scalable feature selection, classification and signature generation for organizing large text databases into hierarchical topic taxonomies." The VLDB Journal The International Journal on Very Large Data Bases 7(3): 163-178.

Chakrabarti, S., B. E. Dom, et al. (2002). Enhanced hypertext categorization using hyperlinks, Google Patents.

Chen, H. and S. Dumais (2000). "Bringing order to the web: automatically categorizing search results." Proceedings of the SIGCHI conference on Human factors in computing systems: 145-152.

Clack, C., J. Farringdon, et al. (1997). "Autonomous document classification for business." Proceedings of the 1st International Conference on Autonomous Agents: 201–208.

Cohen, W. W. and H. Hirsh (1998). "Joins that generalize: text classification using Whirl." Proceedings of KDD-98, 4th International Conference on Knowledge Discovery and Data Mining: 169–173.

Cohen, W. W. and Y. Singer (1998). "Context-Sensitive Learning Methods for Text Categorization." Conference on Research and Development in Information Retrieval (SIGIR) pp 307: 315.

Creecy, R. H. (1991). Trading MIPS and Memory for Knowledge Engineering: Automatic Classification of Census Returns on a Massively Parallel Supercomputer, Thinking Machines Corp.

Dagan, I., Y. Karov, et al. (1997). "Mistake-driven learning in text categorization." Proceedings of the Second Conference on Empirical Methods in NLP: 55-63.

Dattola, R. T. (1979). "FIRST: Flexible Information Retrieval System for Text." J. Am. Soc. Inf. Sci 30(1).

De Heer, T. (1982). "The application of the concept of homeosemy to natural language information retrieval." Information Processing & Management 18(5): 229–236.

Deerwester, S., S. T. Dumais, et al. (1990). "Indexing by latent semantic analysis." Journal of the American Society for Information Science 41(6): 391-407.

Domingos, P. and M. Pazzani (1997). "On the Optimality of the Simple Bayesian Classifier under Zero-One Loss." Machine Learning 29(2): 103-130.

Dumais, S., J. Platt, et al. (1998). "Inductive learning algorithms and representations for text categorization." Proceedings of the seventh international conference on Information and knowledge management: 148-155.

Dumais, S. T. (1991). "Improving the retrieval of information from external sources." Behavior Research Methods, Instruments and Computers 23(2): 229-236.

Escudero, G., L. Marquez, et al. (2000). "Boosting Applied to Word Sense Disambiguation." Arxiv preprint cs.CL/0007010.

Field, B. (1975). "Towards automatic indexing: automatic assignment of controlled-language indexing and classification from free indexing." Journal of Documentation 31(4): 246-265.

Fuhr, N. and C. Buckley (1991). "A probabilistic learning approach for document indexing." ACM Transactions on Information Systems (TOIS) 9(3): 223-248.

Furnkranz, J. (1999). "Exploiting structural information for text classification on the WWW." Intelligent Data Analysis: 487–498.

Galavotti, L., F. Sebastiani, et al. (2000). "Experiments on the use of feature selection and negative evidence in automated text categorization." Proceedings of ECDL-00, 4th European Conference on Research and Advanced Technology for Digital Libraries: 59–68.

Gale, W. A., K. W. Church, et al. (1992). "A method for disambiguating word senses in a large corpus." Computers and the Humanities 26(5): 415-439.

Gray, W. A. and A. J. Harley (1971). "Computer-assisted indexing." Inform. Storage Retrieval 7(4): 167–174.

Hersh, W., C. Buckley, et al. (1994). "OHSUMED: An interactive retrieval evaluation and new large test collection for research." Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval: 192-201.

Hull, D. (1994). "Improving text retrieval for the routing problem using latent semantic indexing." Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval: 282-291.

Ittner, D. J., D. D. Lewis, et al. (1995). "Text categorization of low quality images." Symposium on Document Analysis and Information Retrieval: 301–315.

Iyer, R. D., D. D. Lewis, et al. (2000). "Boosting for document routing." Proceedings of the ninth international conference on Information and knowledge management: 70-77.

Joachims, T. (1996). A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization, School of Computer Science, Carnegie Mellon University.

Joachims, T. (1998). Text Categorization with Suport Vector Machines: Learning with Many Relevant Features, Springer-Verlag London, UK.

Kim, Y. H., S. Y. Hahn, et al. (2000). "Text filtering by boosting naive Bayes classifiers." Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval: 168-175.

Koller, D. and M. Sahami (1997). "Hierarchically classifying documents using very few words." Proceedings of the Fourteenth International Conference on Machine Learning: 170-178.

Lam, S. L. Y. and D. L. Lee (1999). "Feature reduction for neural network based text categorization." Database Systems for Advanced Applications, 1999. Proceedings., 6th International Conference on: 195-202.

Lam, W. and C. Y. Ho (1998). "Using a generalized instance set for automatic text categorization." Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval: 81-89.

Lang, K. (1995). "Newsweeder: Learning to filter netnews." Proceedings of the Twelfth International Conference on Machine Learning 331339.

Larkey, L. S. (1998). "Automatic essay grading using text categorization techniques." Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval: 90-95.

Larkey, L. S. (1999). "A patent search and classification system." Proceedings of the fourth ACM conference on Digital libraries: 179-187.

Larkey, L. S. and W. B. Croft (1996). "Combining classifiers in text categorization." Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval: 289-297.

Lawrence, S. and C. L. Giles (1999). "Accessibility of information on the web." Nature 400: 107.

Lewis, D. D. (1992). "An evaluation of phrasal and clustered representations on a text categorization task." Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval: 37-50.

Lewis, D. D. (1992). "Feature selection and feature extraction for text categorization." Proceedings of the workshop on Speech and Natural Language: 212-217.

Lewis, D. D. (1992). "Representation and learning in information retrieval." PhD Thesis, Department of Computer and Information Science, University of Massachusetts.

Lewis, D. D. (1998). "Naive (Bayes) at forty: The independence assumption in information retrieval." Proceedings of ECML-98, 10th European Conference on Machine Learning 1398: 4–15.

Lewis, D. D. and M. Ringuette (1994). "A comparison of two learning algorithms for text categorization." Third Annual Symposium on Document Analysis and Information Retrieval: 81-93.

Li, H. and K. Yamanishi (1999). "Text classification using ESC-based stochastic decision lists." Proceedings of the eighth international conference on Information and knowledge management: 122-130.

Li, Y. H. and A. K. Jain (1998). "Classification of Text Documents." The Computer Journal 41(8): 537.

Lovins, J. B. (1968). Development of a Stemming Algorithm, MIT Information Processing Group, Electronic Systems Laboratory.

Luhn, H. P. (1958). "The automatic creation of literature abstracts." IBM Journal of Research and Development 2(2): 159-165.

Manning, C. D. and H. Schütze (1999). Foundations of Statistical Natural Language Processing, The MIT Press.

Masand, B. (1994). "Optimizing confidence of text classification by evolution of symbolic expressions." Mit Press In Series In Complex Adaptive Systems: 445-458.

Masand, B., G. Linoff, et al. (1992). "Classifying news stories using memory based reasoning." Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval: 59-65.

McCallum, A. and K. Nigam (1998). "Employing EM in pool-based active learning for text classification." Proceedings of ICML-98, 15th International Conference on Machine Learning: 350–358.

McGill, M. J. and G. Salton (1983). Introduction to modern information retrieval, McGraw-Hill.

Miller, G., U. Princeton, et al. (1998). WordNet, MIT Press.

Mladenic, D. and M. Grobelnik (1998). "Word sequences as features in text-learning." Proceedings of ERK-98, the Seventh Electrotechnical and Computer Science Conference: 145–148.

Moulinier, I. and J. G. Ganascia (1996). "Applying an existing machine learning algorithm to text categorization." Lecture Notes In Computer Science; Vol. 1040: 343-354.

Moulinier, I., G. Raskinis, et al. (1996). "Text categorization: a symbolic approach." Proceedings of the Fifth Annual Symposium on Document Analysis and Information Retrieval.

Ng, H. T., W. B. Goh, et al. (1997). "Feature selection, perception learning, and a usability case study for text categorization." Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval: 67-73.

Nigam, K. and R. Ghani (2000). "Analyzing the effectiveness and applicability of co-training." Proceedings of the ninth international conference on Information and knowledge management: 86-93.

Oh, H. J., S. H. Myaeng, et al. (2000). "A practical hypertext catergorization method using links and incrementally available class information." Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval: 264-271.

Petrarca, A. E. and W. M. Lay (1969). "Use of an automatically generated authority list to eliminate scattering caused by some singular and plural main index terms." Proceedings of the American Society for Information Science 6: 277-82.

Pierre, J. M. (2000). "Practical Issues for Automated Categorization of Web Sites." Electronic Proc. ECDL 2000 Workshop on Semantic Web.

Porter, M. (1980). "An Algorithm for Suffix Stripping Program." Program 14(3): 130-137.

Quinlan, J. R. (1986). "Induction of decision trees." Machine Learning 1(1): 81-106.

Ruiz, M. E. and P. Srinivasan (1999). "Hierarchical neural networks for text categorization (poster abstract)." Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval: 281-282.

Sable, C. L. and V. Hatzivassiloglou (2000). "Text-based approaches for non-topical image categorization." International Journal on Digital Libraries 3(3): 261-275.

Salton, G. and C. Buckley (1988). "Term-weighting approaches in automatic text retrieval." Information Processing and Management: an International Journal 24(5): 513-523.

Salton, G., A. Wong, et al. (1975). "A vector space model for information retrieval." Communications of the ACM 18(11): 613-620.

Schapire, R. E. and Y. Singer (2000). "BoosTexter: A Boosting-based System for Text Categorization." Machine Learning 39(2): 135-168.

Schütze, H., D. A. Hull, et al. (1995). "A comparison of classifiers and document representations for the routing problem." Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval: 229-237.

Sebastiani, F., A. Sperduti, et al. (2000). "An improved boosting algorithm and its application to automated text categorization."

Sinka, M. P. and D. W. Corne (2002). "A large benchmark dataset for web document clustering." Soft Computing Systems: Design, Management and Applications 87: 881-890

Sj, C. and D. J. Waltz (1992). "Trading mips and memory for knowledge engeneering." Communications of the ACM 35: 48-64.

Slattery, S. and T. Mitchell (2000). "Discovering test set regularities in relational domains." Proc. ICML.

Slonim, N. and N. Tishby (2001). "The power of word clusters for text classification." Proceedings of ECIR-01, 23rd European Colloquium on Information Retrieval Research.

Taira, H. and M. Haruno (1999). "Feature selection in SVM text categorization." Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence table of contents: 480-486.

Tauritz, D. R., J. N. Kok, et al. (2000). "Adaptive Information Filtering using evolutionary computation." Information Sciences 122(2-4): 121-140.

Tzeras, K. and S. Hartmann (1993). "Automatic indexing based on Bayesian inference networks." Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval: 22-35.

Vapnik, V. N. (1995). "The Nature of Statistical Learning Theory [M]." NY: Springer-Verlag.

Wai, L. A. M. and L. Fan (1997). "Using a Bayesian Network Induction Approach for Text Categorization." Proceedings of the 15th International Joint Conference on Artificial Intelligence: 745-750.

Weiss, S. M., C. Apte, et al. "Maximizing text-mining performance." Intelligent Systems and Their Applications, IEEE [see also IEEE Intelligent Systems] 14(4): 63-69.

Wiener, E., J. O. Pedersen, et al. (1995). "A neural network approach to topic spotting." Proceedings of the Fourth Annual Symposium on Document Analysis and Information Retrieval (SDAIR'95): 317-332.

Yang, Y. and C. G. Chute (1994). "An example-based mapping method for text categorization and retrieval." ACM Transactions on Information Systems (TOIS) 12(3): 252-277.

Yang, Y. and X. Liu (1999). "A re-examination of text categorization methods." Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval: 42-49.

Yang, Y. and J. O. Pedersen (1997). "A comparative study on feature selection in text categorization." Proceedings of the Fourteenth International Conference on Machine Learning 97.

Yang, Y., S. Slattery, et al. (2002). "A Study of Approaches to Hypertext Categorization." Journal of Intelligent Information Systems 18(2): 219-241.