ELECTRONIC WORKSHOPS IN COMPUTING

Series edited by Professor C.J. van Rijsbergen

Jonathan Furner, School of Information and Media Studies, and David Harper, School of Computer and Mathematical Studies, The Robert Gordon University, Aberdeen, Scotland. (Eds)

Information Retrieval Research

Proceedings of the 19th Annual BCS-IRSG Colloquium on IR Research,
Aberdeen, Scotland, 8-9 April 1997

Paper:

Feature Selection: A Useful Preprocessing Step

I. Moulinier

Published in collaboration with the
British Computer Society

Feature Selection: a Useful Preprocessing Step

Isabelle Moulinier

LIP6, Université P. et M. Curie

Paris, France

moulinie@laforia.ibp.fr

Abstract

Statistical classification techniques and machine learning methods have been applied to some Information Retrieval (IR) problems: routing, filtering and categorization. Most of these methods are usually awkward and sometimes intractable in highly dimensional feature spaces. In order to reduce dimensionality, feature selection has been introduced as a pre-processing step. In this paper, we assess to what extent feature selection can be used without causing a loss in effectiveness. This problem can be tackled since a couple of recent learners do not require a preprocessing step.

On a text categorization task, using the Reuters-22,173 collection, we give empirical evidence that feature selection is useful: first, the size of the collection index can be drastically reduced without causing a significant loss in categorization effectiveness. Then, we show that feature selection speeds up the time required to automatically build the categorization system.

1 Introduction

Recent research [10, 17, 19, 23] has shown that document routing, filtering and categorization could be modeled as classification problems. In this perspective, documents are to be assigned to one of two classes, relevant or non-relevant. Text classifiers can then be constructed automatically provided that a large sample of judged documents is available. We focus on binary classification, *e.g.* document filtering and text categorization, where we decide whether or not a document belongs to a class, and not how strongly a document belongs a class (as in a routing problem). In the remainder of this paper, we study text categorization as a particular text classification task.

A central problem in text classification using learning methods is the high dimensionality of the feature space, where there exists a potential feature for each unique term, word or phrase, found in the collection. Standard textual collections, like the Reuters-22,173 and the Ohsumed collection, include several tens or hundreds of thousands of features. Most learning methods, from either statistical classification or machine learning, are usually applied to small-sized problems which are described using at most a few hundreds of features. Scaling-up by a factor of one hundred or one thousand is hardly tractable for most learning techniques. A natural solution consists in reducing dimensionality. This includes selecting promising informative features, and constructing new features which semantically combine low-level features (latent semantics indexing is an example of feature construction used prior to classification techniques [23, 26]). In the following, we only address feature selection as a preprocessing step.

Several methods have been proposed to reduce dimensionality using feature selection. Most of those approaches rely on the evaluation of a “goodness” measure to filter promising features. In the following, we refer to this “goodness” measure as the quality criterion. Quality measures used in IR include the information gain criterion [16, 20], mutual information [25, 28], the χ^2 statistics [23, 26, 28] or simply frequency [2, 28]. Moulinier *et al.* [22] presented an alternative approach, when they used a “light” inductive rule learner SCAR to select informative features. These feature selection techniques have been investigated and compared in terms of effectiveness with different learners [23, 25, 22]: statistical classification and neural network learning when combined with dimensionality reduction were shown to perform significantly better than relevance feedback via Rocchio expansion [23, 25]. A recent study conducted a cross-method evaluation of feature selection using numerical learners (for instance a nearest neighbors method) [28]: one of Yang and Pedersen’s objectives was to assess to what extent the original feature set could be reduced without causing a loss of categorization effectiveness.

Our goal in this paper is similar. We use rule learning to automatically build categorization systems. Rule learners attempt to find general descriptions of the available judged documents and express these descriptions as rules of the form $C \leftarrow t_i \in d, t_j \in d$ (if terms t_i and t_j are present in document d , then d is relevant, *i.e.* assign class C to document d). A couple of recent rule learners (RIPPER [7, 8] and SCAR [22]) have been applied to text categorization and document filtering without prior feature selection. However, the construction process is time and storage consuming. With a series of experiments on the Reuters-22,173 corpus, we assess how drastically the collection index (*i.e.* the feature set) can be reduced without significantly decreasing the performance of the constructed categorization system. As a side effect, we show that learning time decreases when feature selection is performed.

Section 2 describes feature selection techniques from several domains and motivate the choices performed in IR. Section 3 gives an overview of the rule learners RIPPER and SCAR. We present our experimental set-up in Section 4. Experimental results are exposed and discussed in Section 5. Section 6 states our conclusions.

2 Feature Selection: What can be used from Machine Learning Research?

For the past few years, there has been an increase of interest for feature selection in machine learning. In these studies, feature selection is applied to find a subset which maximizes a quality criterion, such as accuracy or the size of the built classification rules, *i.e.* it aims at finding an optimal subset. It integrates techniques from earlier work in several domains, such as data analysis and pattern recognition. Thus, the purpose of feature selection in machine learning and in text classification differs. The latter often is a requirement, while the former corresponds to an optimization step.

We distinguish two trends in the approaches developed for feature selection in machine learning:

- The quality of features is assessed for each individual feature using a numerical criterion. A threshold which is usually user-defined, fixes the number of selected features. This approach is widely used in text classification, as we mention in the previous section.
- The quality of a subset is assessed globally. This approach chiefly relies on a strategy to search through the set of all possible subsets of the original collection index.

In the remainder of this section, we briefly present both trends and discuss their properties when feature selection is applied to text classification.

2.1 Selection of Individual Features

Selection of individual features is based on the evaluation of the quality of features and generally stops when the number of selected features rises above a user-defined threshold. This process is straightforward, and requires no extensive computation, apart from the quality criterion. The quality of a feature can be computed using different criteria:

- A measure based on corpus statistics, such as frequency, information gain, mutual information or the χ^2 statistics. These four criteria have been studied in [25, 28] where little difference was found between information gain and the χ^2 statistics using numerical learners.
- A weighting scheme such as Rocchio [26] or RELIEF+ [22]. All features are given a initial weight; the selection algorithm passes through the set of judged documents and modifies feature weights. Thresholding is then used either on weights or on the number of features.
- An inductive algorithm which builds a rule set [22] or decision tree [5]. Features appearing in the rules of the constructed system are selected. They are typical of documents that are assigned to the class of relevant documents and exclude documents that are irrelevant.

The two first methods are relatively efficient, even on very large data sets. The last approach may be intractable in text classification problems, when the inductive algorithm does not scale-up to tens of thousands features. Let us remark that all these selection methods are task-oriented: they use information on classes (relevant and irrelevant to a category or profile) and features during the selection process.

Figure 1: Illustration of the filter and wrapper models

2.2 Selection of an Optimal Subset

The previous approach to feature selection can be opposed to the selection of a subset of features, which is more frequent in machine learning. Selecting a subset not only involves choosing a criterion to assess the quality of each individual subset, but also relies on a search strategy among all candidate subsets. More precisely, selecting a subset of features depends on deciding:

- where the search algorithm starts,
- which search strategy is used,
- which quality criterion is chosen,
- when the search is ended (*e.g.* when no improvement in the quality criterion is found).

The subset used at start and the stopping criterion are linked to the search strategy and quality criterion. The starting subset is the empty set, when the adopted strategy adds one feature at a time (it is called a forward selection). It can also be the original feature set when the choice is to remove the less informative feature from candidate subsets. Finally, when the search strategy adds and removes features to candidate subsets, the initial subset is often chosen at random.

Most feature selection approaches in machine learning adopt heuristic search (*e.g.* forward selection, backwards elimination, and stepwise selection), because an exhaustive search of the space generated by the original feature set is not feasible on large feature set (the number of subsets is exponential). These strategies are inherited from statistics and pattern recognition [13].

The quality criterion can be a measure based on corpus statistics similar to criteria used for individual features. For instance, information gain is the criterion used in [3, 4]. Feature selection based on such measures refers to a filter model, since it filters out unimportant features prior to learning. However, recent studies in machine learning [12, 1, 14] have criticized this filter model, because it does not take into account the learning algorithm used after the feature selection stage.

The wrapper model is proposed as an alternative solution [12]. In this model, the quality criterion is computed using the learning algorithm which will be used to build the filtering or categorization system. Hence, the selected subset is said to be optimal for the given learner on the data set under consideration. Both models are illustrated in Figure 1. In most approaches using the wrapper model, the quality criterion corresponds to the predictive accuracy of the learner. This means that a learning process is called every time a subset has to be evaluated. To avoid selecting a feature set that is sensitive to a given training set, cross-validations are performed. This choice renders computation even more critical in large applications.

2.3 Discussion

While the wrapper model has the advantage of finding an optimal subset, it is intractable on large feature sets (including some thousands features). Let us consider, for instance, a greedy forward selection strategy, starting from the empty set. If there are n features, the wrapper model will run n k -fold cross-validations using a learning algorithm to select the optimal subset containing one feature. The selection will then run $(n - 1)$ k -fold cross-validations in order to add an additional feature to the optimal feature subset found so far. This process is repeated until the stopping criterion is met. Let p be the size of the selected feature set; the learner is run an order of $kn(p + 1)$ times during the feature selection process, which is computationally intensive when n is large. Moreover, in the worst case, the learner used as the quality function has to cope with the original feature set. This is intractable for most off-the-shelf learners.

A filter approach to the selection of an optimal subset also presents a few drawbacks when applied to textual collections. First, while computation is less intensive than with the wrapper model, since no cross-validation is required, it is more costly than selecting features on their individual quality. Next, quality criteria require the estimation of the joint probabilities of all features in a subset, given the class. This estimation may become unreliable when the size of candidate subsets grows: some of these joint probabilities rapidly tend towards zero. Indeed, irrelevant documents are far more numerous than relevant ones, and unique terms occur rather rarely compared to the number of terms in the collection index.

Therefore, it seems natural to prefer selecting individual features on the basis of corpus statistics. However, few IR problems are actually binary: there are more than two profiles in most filtering applications, and more than two categories in text categorization problems. Therefore, a commonly adopted solution binarizes the classification problems: for each profile, a binary classifier is built, where all documents which are not judged relevant for this profile are used as negative instances in the learning process. We take advantage of this binarization during the feature selection, as the selected subsets are local to each profile. This strengthens the task-oriented characteristics of feature selection.

Finally, it is worth noticing that the feature selection process described in this section is not performed on raw documents, but rather on indexed representations. This means that the number of unique terms has already been reduced using stemming, stop-words, frequency thresholding, and any other reduction method used in IR.

3 Classifiers

Previous research in text classification has used classification trees [16], Bayesian classifiers [16, 17], neural networks [26, 23], nearest neighbors methods [27, 15], discriminant analysis [11, 23], logistic regression [23] and rule induction [2, 8, 22]. The majority of these learners requires a preprocessing step to restrict the number of features, because computation in the original feature set is intractable.

We investigate two different rule learners which share the ability of learning without feature reduction. RIPPER was proven efficient on both standard machine learning problems [6] and text collections [8]. SCAR was designed as an inductive feature selection filter [22]; used as an inductive learner, it does not perform as well as RIPPER. However, it can handle several thousands features and enables us to assess the impact of feature selection. Both learners implement a *separate and conquer* method, *i.e.* they build a rule at a time, and remove all documents covered by this rule before growing a new rule. They also attempt to characterize a single class which corresponds to documents relevant to category C , and use a default rule that decides that documents are irrelevant. However, they differ in several respects:

- the strategy adopted to grow a rule, and
- the stopping criterion and handling noisy data: we consider that a data set is noisy when two documents which share a common representation are not labeled by the same class.

In addition to those two classifiers, we use the Rocchio algorithm, without prior feature selection, which represents the baseline in our experiments. Rocchio algorithm has been used in earlier studies, and its effectiveness was found lower than statistical classification and machine learning methods [23, 8]. In the following, we briefly describe the differences between the two rule learners.

3.1 Ripper

RIPPER is a general purpose rule learner and has been described elsewhere by Cohen [6]. It builds a rule set by repeatedly adding rules to an empty set, until all positive examples (i.e. all relevant documents in the training set) are covered. Rules are grown by greedily adding conditions to the antecedent of the rule (starting from the empty antecedent) until no negative instance is covered. Conditions are chosen using entropy as a quality measure. After growing a rule, the rule is pruned until some error threshold is reached. This enables the learner to cope with noisy data sets. The rule set is further post-processed as to reduce its size and to improve its generalization of training data.

RIPPER has been extended to handle set-valued attributes [7]. We use this property in our experiments, as we prefer a set-based representation $w_i \in \text{text}$ to representing documents in terms of Boolean vectors. Hence rule conditions are of the form $w_i \in \text{text}$ or $w_i \notin \text{text}$. We are thus able to compare RIPPER with and without feature selection using the same representation scheme. Moreover, rule sets generated by RIPPER can be easily understood by a human indexer.

3.2 Scar

SCAR was designed as a feature selection algorithm on text classification problems. As such, it is not a general purpose learner: it only handles sparse data sets described by Boolean features. Compared to RIPPER, it is a naive learning algorithm since it performs no pruning. It is thus unable to cope with noisy data, which explains its lower performance on categorization problems [21].

SCAR [22] builds a rule set by repeatedly adding rules to an empty rule set, until all positive examples are covered, or until no additional rule can be found given the chosen learning parameters. Rules are grown by searching the best rule (in terms of cover¹) in a local subspace; this subspace is generated using the features appearing in a seed document. This best-first search is made possible by a simple trick: while the number of unique terms reaches tens of thousands, documents hardly contain more than 500 unique terms. The dimensionality is automatically reduced by the learner from the entire feature space to a document-based feature space. However, the adopted strategy leads to only consider conditions of the form $w_i \in \text{text}$ in order to further reduce the search. Finally, the stopping criterion used in SCAR is consistency with the training data, which means that a rule can be generated only when it covers some relevant documents but no irrelevant ones.

4 Experimental Set-up

Binary text classification includes text filtering and text categorization. In our experiments, we study text categorization, *i.e.* the content-based assignment of one or more categories to documents. This task is transformed into several binary classification tasks: the problem of learning to assign all categories at a time is transformed into n sequential binary sub-problems (assign or not one given category).

We use the Reuters-22,173 collection, which has become a standard for the past few years [17, 2, 26, 8, 9, 22]. It consists in 22,173 news stories from the Reuters financial newswire. There are 135 predefined categories to which documents are assigned. We exclude from the collection all documents that are not assigned to a category, in an attempt to decrease noise in the data set. There remains 11,664 stories, split into 7,789 training and 3,875 test documents.

We use the collection preprocessing performed by Lewis [17, p. 99]: documents were tokenized, but no stemming was performed; a stop-word list was used and terms that appeared less than twice in the 22,173 documents were removed. This process resulted in circa 20,000 unique terms.

This original set of feature is further reduced using a filter model which selects individual features. The quality criterion is the information gain criterion. It was used in several other studies [17, 16, 22, 28]:

$$IG(W, C) = \sum_{w=0,1} \sum_{c=0,1} p(W = w, C = c) \log_2 \frac{p(W = w, C = c)}{p(W = w) \times p(C = c)}$$

Probabilities $p(W = w, C = c)$, $p(W = w)$ et $p(C = c)$ are estimated on the training set: let a be the number of documents in which word W and category C cooccur and n the total number of training documents, $p(W = 1, C =$

¹The cover represents the number of instances that verifies the conditions of a tentative rule.

Threshold	10	20	30	40	50	60	75	90
Index size	736	1340	1869	2321	2762	3168	3707	4244
Threshold	100	150	200	250	300	400	600	None
Index size	4541	5752	6771	7489	8160	9133	10596	19573

Table 1: The size of the collection index at several thresholds.

1) $= \frac{a}{n}$. In the experiments reported below, we use a threshold on the number of features selected per category. The threshold varies from 10 to 400. Indeed, earlier studies on feature selection for text classification reported the optimal number of features varied with the learner [16, 26]. We wish to assess whether we find similar results when learners do not require a feature selection preprocessing.

We perform 135 binary classifications using RIPPER, SCAR and the Rocchio algorithm. Both RIPPER and Rocchio belong to the RIPPER package provided by W. Cohen. Due to some of SCAR limitations, we adopt the following learning scheme: we represent documents as sets of terms; even though the selection is performed locally, we retain for each document the terms appearing in the union of all subsets. Hence even though the threshold is n , document length may be larger than n . We use RIPPER similarly, to perform a comparison. We call this configuration RIPPER/A. We also represent each document using only the features selected locally to a category. If the threshold is n , document length will remain smaller than n . This last configuration is equivalent to a vector-based representation using n Boolean features.

Finally, we choose the F_β measure to assess effectiveness [24, 18]. Let T be the category under study. Let a be the number of documents correctly assigned by the categorization system, b the number of documents wrongly labeled by T , c the number of labeled documents that were not found by the system, F_β is defined as $F_\beta = \frac{(\beta^2+1)a}{(\beta^2+1)a+b+\beta^2c}$. To summarize results obtained on all categories, we use micro- and macro-averaging: micro-averaging favors frequent categories while macro-average gives an equal weight to all categories.

5 Experimental results

Table 1 summarizes the reduction of the collection index achieved at several thresholds. A direct consequence is thus a noticeable gain in storage requirements. Besides, let us remark that the global size of the reduced index is far smaller than $135n$, where n is the number of features selected per category. Indeed, local feature subsets share common terms, particularly when categories are conjointly assigned to documents with a high frequency (for instance categories *wheat* and *grain*).

Figures 2 and 3 report performance curves for SCAR and RIPPER/A, and RIPPER respectively. These results are two-folded. First, we observe that a drastic reduction of the index does not cause a significant loss in effectiveness. Both learners achieve similar performances using the original index and a index reduced by nearly 80% (at a threshold of 75 features). RIPPER allows to reduce the collection index even more: it is competitive at a threshold of 50 features. Our results are similar to those reported in [28], where the Reuters-22,173 and the Ohsumed test collections are investigated using numerical learners.

Next, we see that RIPPER performs better than SCAR: at a given threshold in the feature selection process, the difference is in most cases significant. This behavior is not surprising since SCAR does not handle noisy data efficiently. Moreover, we observe that the difference between RIPPER and SCAR decreases when the size of the selected features increases. We find two explanations for that behavior: noise tolerance, which is inexistent in SCAR, and the influence of the quality criterion. While SCAR builds positive conditions (of the form $w_i \in \text{text}$), the information gain measure may select features because their absence, and their presence when the category is not assigned, is informative: the information gain measure is indeed symmetrical in W and C . Features selected on the basis of negative evidence are useless to the search SCAR performs. Category *acquisitions, mergers* is an example of that fact: 3 out of the first 10 selected features are chosen because, whenever they appear, category *acquisitions, mergers* is not assigned.

This last explanation holds for the following remark as well: SCAR ran on a reduced feature set using a threshold of 10 features shows an effectiveness which is significantly lower than SCAR other configurations. On the other hand,

Figure 2: Effectiveness (micro-averaged F_1) of SCAR and RIPPER/A vs. the number of unique terms

Figure 3: Effectiveness (micro-averaged F_1) of RIPPER vs. the number of unique terms

Learner	Micro F_1	Macro F_1
RIPPER	0.806	0.587
SCAR	0.756	0.567
Rocchio	0.735	0.485

Table 2: Summary of effectiveness. Feature selection used a threshold of 75 features for SCAR, and of 50 for RIPPER. No selection was used with Rocchio.

Figure 4: Learning time for each learning configuration vs. the number of unique terms

the circa 5% difference between RIPPER (and RIPPER/A) lowest and highest performances is not found significant². However, RIPPER seems to perform, in all cases, a little better with negative conditions ($w_i \notin \text{text}$) than without.

The lack of noise tolerance in SCAR has another impact: the learner builds larger rule sets. More specifically, RIPPER not only constructs less rules (474 for RIPPER, as opposed to 4099 for SCAR), but also uses less features (399 versus 1452) from the reduced feature set. The effect of noise tolerance has been discussed in details elsewhere [21].

Table 2 gives a synthesis of our discussion on effectiveness. They recall SCAR and RIPPER performances and show the effectiveness of the Rocchio algorithm, with the parameters used in [8]. Both SCAR and RIPPER perform significantly better than the Rocchio algorithm. This confirms results obtained in earlier studies [23, 8]. The difference between micro-averaged F_1 and macro-averaged F_1 show that all approaches perform significantly better on frequent categories than on rare categories.

Finally, we show that feature selection speeds up learning. Figure 4 displays curves related to learning times (time spent reading data files is excluded). Experiments were performed on an UltraSparc-1. Learning time is summed up for all 135 categories. We observe that SCAR achieves similar performances and decreases learning time by 50%, when 75 features per category are selected (an overall 3707 features). Similar results are achieved by Ripper/A, while Ripper drastically cuts down learning times: using 50 features per category reduces learning time by a factor of 8, while performance remains unchanged.

The slope of the curves displayed in Figure 4 loses its sharpness as the number of features grows. We believe this derives from a lack of evidence in training data: both learners build a rule set until all positive examples are covered

²We use the Friedman test for multiple comparison to test for significant differences [10].

and remove instances when they are covered by a rule; hence they do not consider conditions ($w_i \in \text{text}$) that cover no instance, a case that may arise when w_i appears in one of the removed instances.

Reducing learning time may become of great utility when rule learners are applied to text filtering. The learners we studied are off-line learners, while filtering is by essence a dynamic process. A significant cut in the time required to construct a filtering tool will enable us to build anew the filtering tool more frequently and to integrate relevance feedback to some extent.

6 Conclusion

In this paper, we investigated to what extent the original feature set could be reduced without causing a loss of classification accuracy, with an emphasis on rule learning. We first examined recent approaches to feature selection in machine learning. While the selection of an optimal subset is more attractive in standard machine learning applications, we rejected that approach, since computation was either hardly tractable for (very) large feature sets, or not reliable enough. Our experiments on the Reuters-22,173 collection led to results which were coherent with other studies [23, 8, 28]:

- both learners, with and without feature selection, performed better than the Rocchio algorithm, and
- a reduction of 90% (resp. 80%) of the original feature set caused no loss in accuracy when RIPPER (resp. SCAR) was used. This difference between these two rule learners is explained by the SCAR stopping criterion, consistency with training data. This aspect is discussed elsewhere [21].

Moreover, we showed that feature selection allowed learning to dramatically speed up; in the best case, a factor of 8 was observed, and feature selection caused no loss of accuracy. This property is important, since it allows to build filtering or categorization tools anew when feedback is given, or new training data available.

The work presented here can be extended in several ways: we reported results with the information gain criterion. Other existing quality measures have to be tried. The observations we made may depend on the Reuters-22,173 collection. Experimenting with another collection such as the Ohsumed collection used in [19, 28] would help generalizing our results.

References

- [1] D. Aha and R. L. Bankert. A comparative evaluation of sequential feature selection algorithms. In *Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics*, pages 1–7, Ft. Lauderdale, 1995.
- [2] C. Apté, F. Damerau, and S. Weiss. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, July 1994.
- [3] P. Battiti. Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on Neural Networks*, 5(4):537–550, July 1994.
- [4] B. V. Bonnländer and A. S. Weigend. Selecting input variables using mutual information and nonparametric density evaluation. In *Proceedings of ISANN'94*, 1994.
- [5] C. Cardie. Using decision trees to improve case-based learning. In *Proceedings of the Tenth International Conference on Machine Learning*, Amherst, MA, 1993. Morgan Kaufman.
- [6] W. Cohen. Fast effective rule induction. In *Proceedings of the Twelveth International Conference on Machine Learning*, Lake Tahoe, California, 1995.
- [7] W. Cohen. Learning trees and rules with set-valued features. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI'96)*, Portland, Oregon, 1996.

- [8] W. Cohen and Y. Singer. Context-sensitive learning methods for text categorization. In *Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Zürich, Switzerland, 1996.
- [9] J. Goldberg. CDM: an approach to learning in text categorization. *International Journal of Artificial Intelligence Tools*, 1996.
- [10] D. Hull. Using statistical testing in the evaluation of retrieval performance. In *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 329–338, 1993.
- [11] D. Hull. *Information Retrieval using statistical classification*. PhD thesis, Department of Statistics, Stanford University, November 1994.
- [12] G. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In *Machine Learning: Proceedings of the Eleventh International Conference*. Morgan Kaufmann, 1994.
- [13] J. Kittler. Feature selection and extraction. In T. Y. Young and K.-S. Fu, editors, *Handbook of Pattern Recognition and Image Processing*. Academic Press, 1986.
- [14] P. Langley. Selection of relevant features in machine learning. In *Proceedings of the AAAI Fall Symposium on Relevance*, pages 1–5, New Orleans, LA, 1994. AAAI Press.
- [15] L. S. Larkey and W. B. Croft. Combining classifiers in text categorization. In *Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Zürich, 1996.
- [16] D. Lewis and M. Ringuette. A comparison of two learning algorithms for text categorization. In *Proceedings of the Symposium on Document Analysis and Information Retrieval*, 1994.
- [17] D. D. Lewis. *Representation and Learning in Information Retrieval*. PhD thesis, Graduate School of the University of Massachusetts, 1992.
- [18] D. D. Lewis. Evaluating and optimizing autonomous text classification systems. In *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1995.
- [19] D. D. Lewis, R. E. Shapire, J. P. Callan, and R. Papka. Training algorithms for linear text classifiers. In *Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Zürich, 1996.
- [20] I. Moulinier. A framework for comparing text categorization systems. In *Working notes of the AAAI Spring symposium on Machine Learning and Information Access*, 1996.
- [21] I. Moulinier. Is learning bias an issue on the text categorization problem? Submitted to SIGIR’97, 1997.
- [22] I. Moulinier, G. Raškinis, and J.-G. Ganascia. Text categorization: a symbolic approach. In *Proceedings of the 5th Symposium on Document Analysis and Information Retrieval*, Las Vegas, NV, 1996. ISRI.
- [23] H. Schütze, D. A. Hull, and J. O. Pedersen. A comparison of classifiers and document representations for the routing problem. In *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 229–238, Seattle, WA, 1995.
- [24] G. J. van Rijsbergen. *Information Retrieval*. Butterworths, 1979.
- [25] E. Wiener. A neural network approach to topic spotting in text. Master’s thesis, Department of Computer Science, University of Colorado, 1995.

- [26] E. Wiener, J. O. Pedersen, and A. S. Weigend. A neural network approach to topic spotting. In *Proceedings of the Fourth Symposium on Document Analysis and Information Retrieval (SDAIR'95)*, 1995.
- [27] Y. Yang. Expert network: Effective and efficient learning from human decisions in text categorization and retrieval. In *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1994.
- [28] Y. Yang and J. Pedersen. A study on feature selection methods in text categorization. Submitted to ICML'97, 1997.