Integrating Feature and Instance Selection for Text Classification

Dimitris Fragoudis[1,3] dfragoud@ceid.upatras.gr
Dimitris Meretakis [2] meretaks@ceid.upatras.gr
Spiros Likothanassis [1,3] likothan@cti.gr

[1] Computer Engineering & Informatics Department, University of Patras, Rio GR-26500, Greece
[2] Zurich Financial Services, Switzerland
[3] Computer Technology Institute, Riga Fereou 613, Patras GR-26221, Greece

ABSTRACT

Instance selection and feature selection are two orthogonal methods for reducing the amount and complexity of data. Feature selection aims at the reduction of redundant features in a dataset whereas instance selection aims at the reduction of the number of instances. So far, these two methods have mostly been considered in isolation. In this paper, we present a new algorithm, which we call *FIS* (Feature and Instance Selection) that targets both problems simultaneously in the context of text classification

Our experiments on the Reuters and 20-Newsgroups datasets show that FIS considerably reduces both the number of features and the number of instances. The accuracy of a range of classifiers including Naïve Bayes, TAN and LB considerably improves when using the FIS preprocessed datasets, matching and exceeding that of Support Vector Machines, which is currently considered to be one of the best text classification methods. In all cases the results are much better compared to Mutual Information based feature selection. The training and classification speed of all classifiers is also greatly improved.

Categories and Subject Descriptors

I.5.2 **[Pattern Recognition]**: Design Methodology -Feature evaluation and selection. I.5.4 **[Pattern Recognition]**: Applications - Text processing.

1. INTRODUCTION

Feature selection is a major research area within IR since the reduction of the features used for the representation of documents is an absolute requirement for the use of any but the simplest machine learning algorithms [3]. Feature selection methods reduce the dimensionality of datasets by removing features that are considered irrelevant for the classification. This transformation procedure has been shown to present a number of advantages such as smaller dataset size, less computational requirements for the classification algorithms (especially those that do not scale well with the feature set size), reduction of the search space and improved classification accuracy

As Figure 1 illustrates, *instance selection* [3] works orthogonal to feature selection. The aim here is the reduction of the number of instances presented to the classifier during the training phase. The motives are similar to the ones considered in feature selection and include smaller dataset size, faster training and classification time for the classification algorithms (especially those that do not scale well with the dataset size) and improvement of the input quality by removing noise introduced by inconsistent examples and by examples that do not provide information useful for the classification.

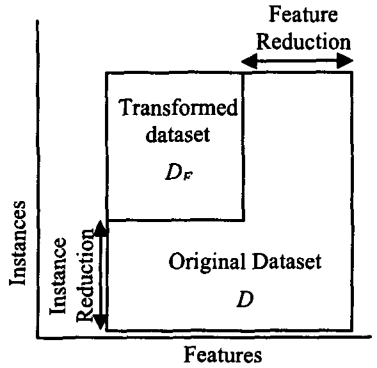


Figure 1: Feature and instance selection as orthogonal tasks.

In this paper, we present our approach that deals with both problems simultaneously. Our algorithm, which we call *FIS* (Feature and Instance Selection), targets the feature and instance selection problem in the context of text classification. FIS works in two steps. In the first step, it sequentially selects features that have high precision in predicting the target class. All documents that do not contain at least one such feature are dropped from the training set. In the second step, FIS searches within this subset of the initial dataset for a set of features that tend to predict the complement of the target class and these features are also selected. The sum of the features selected during these two steps is the new feature set and the documents selected from the first step comprise the training set.

Our experimental results with standard benchmark datasets show that FIS considerably reduces both the number of features and the number of instances. Equally important, the accuracy of a range of classifiers including Naïve Bayes [4], TAN [6] and LB [12] is considerably increased when the resulting training sets are used instead of the original ones. In some cases, these classifiers prove to be as accurate as the Support Vector Machines [7], which is currently considered one of the best text classification methods.

2. Related work

A very good survey on feature and instance selection as two independent problems in the context of machine learning is

SIGKDD '02, July 23-26, 2002, Edmonton, Alberta, Canada.

presented in [3]. In the context of information retrieval and text classification, several works have indicated that effective feature selection can enhance the performance of classifiers, In [5], [11] and [17] a few tens or hundreds of words maximize the performance of a range of classifiers. Similar results are reported in [9][13] as well. SVMs are a notable exception to this since they achieve the highest accuracy when almost all words are used, but they do not scale up in the dataset size

John et. al [8] identify three types of features; *Irrelevant features*, which can be ignored without degradation in the classifier performance, *strongly relevant features* that contain useful information such that if removed the classification accuracy will degrade and *weakly relevant features* that contain information useful for the classification, but are unnecessary given that some other words are present in an instance. In the design of FIS, we took into account this definition of relevance, which is very intuitive. Some words may contribute to the distinguishing of the class, but they might be redundant, as they tend to co-occur with other words, which are also good predictors. Therefore, we can reduce the redundancies hoping that we will not reduce the amount of information in the dataset.

There is not much research on Instance Selection for text classification. The issue is mostly addressed either with the traditional statistical approach of sampling [16] or by more elaborate, but sometimes heuristic, approaches. Most of the work refers to Instance-based or lazy algorithms [1]. In [15] the problem is addressed using a distance measure. In essence instances that are "closer" to each other tend to bear overlapping information; therefore, some of them can be discarded. Active Learning [10][14] is another approach to Instance Selection where the learner has access to a pool of unlabeled instances and can request the labels for some of them.

3. Algorithm *FIS* description

In this section we describe in detail our algorithm for feature and instance selection, called *FIS* (**F**eature and **I**nstance **S**election), and state the objectives behind our choices for the various steps of the algorithm.

Assume a document collection $D = \{d_1, d_2, \ldots d_{|D|}\}$, where each document d_k, k in $\{1, 2, \ldots, |D|\}$, contains one or more words from a vocabulary $W = \{W_1, W_2, \ldots W_{|W|}\}$. Each word W_i is associated with a binary variable w_i where $w_i = 1$, if W_i is present one or more times in d_k and $w_i = 0$, if W_i is not present in d_k. In addition, each document d_k is associated with a class label c, which indicates whether d_k belongs to the target class C ($c = 1$) or not ($c = 0$) in which case we will say that it belongs to class C'. For the sake of simplicity, we ignore the number of occurrences of a particular word W_i in a document d_k as well as their relative position in the document. Instead, we use the so-called "Bag of words" [11] document representation. Additionally, we consider only the binary class problem; multi-class problems have to be split into a sequence of binary class problems. Although in some cases this is not very convenient, it is the case in a variety of problems where a document may belong to more than one possible class.

The FIS algorithm operates in two steps. During the first step FIS searches for a subset F_P of the original vocabulary W that contains the words of W that are the best predictors of the given class C. We call such words *positive features*. As a convenient side effect of this selection, D is pruned and only the documents with non-empty projection on F_P are kept. The resulting dataset D_{F_P} contains the documents with at least one word from F_P. All other documents are ignored during training and assigned to C' during classification. Our goal is that the set D_{F_P} contains the majority of the C-labeled documents and only a small portion of the documents from C' while at the same time the documents outside D_{F_P} will mostly belong to C'. Taking this argument to the extreme, in the ideal case D_{F_P} would contain *all* documents of class C and *only* these documents. In reality this is not the case and the second step aims at refining the results of the first step by discovering words that are good predictors of C'.

In the second step *FIS* searches within D_{F_P} for the set of negative features F_N. This step is exactly symmetrical to the first one and the resulting set contains the words of W that are the best predictors of class C' *within* D_{F_P}. The output of *FIS* is the feature set $F = F_P \cup F_N$ and the instance set $T = D_{F_P}$ that contains the documents in D_{F_P} represented using the features in F.

In Figure 2 we list FIS0, a procedure that is called twice, the first time to discover F_P from the total set of documents D and the second time to discover F_N from the documents in D_{F_P}.

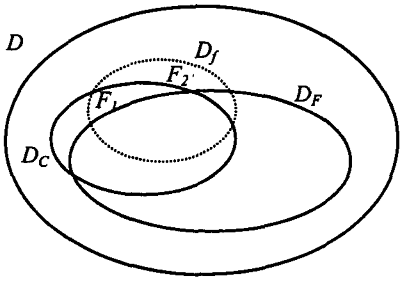
```
FIS0(C, W, D, min_score, c_support, F, D_F)
Input:
  A target class C
  A vocabulary W
  A dataset D of labeled documents containing words
  from W
  User defined thresholds: min_score, c_support.
Output:
  A subset F of W that contains best features for
  predicting class C
  A subset D_F of D with the documents that have non-
  empty projection on F
Algorithm:
  $D_F = \{\}$ ,   $F = \{\}$ ,  $D'_C = D - D_C$
  repeat{
    for each feature $f_j$ in $W - F$ calculate the
    following{
      $D_{f_j}$ = {All documents that contain $f_j$}
      $F_1 = (D_{f_j} \cap D_C) - D_F$
      $F_2 = (D_{f_j} \cap D'_C) - D_F$
      $S_{f_j} = score(f_j) = |F_1| / (|F_2| + 1/2)$
    }
    if there is a feature f that meets the
    following three criteria:
      a. $S_f$ > min_score
      b. $S_f \geq S_{f_k}$ for each other feature $f_k$ in
      $W - F$
      c. $|D_{f_j}| \geq c\_support \cdot |D_C|$
    then{
      $F = F \cup \{f\}$
      $D_F = D_F \cup F_1 \cup F_2$
    }
  }until no new feature is added to F
```

Figure 2: Procedure FIS0

The procedure FIS0 starts with an empty set of relevant features F and an empty set of relevant documents D_F. At each iteration it selects the best new feature f to be added to the set of relevant features, where "best" is measured with the function *score(f)* in line 7 of Figure 2. This feature is added to F and all documents that contain it are added to D_F if they are not already there. This procedure repeats until some stopping criteria are met.

Figure 3 illustrates the sets involved in the operation of FIS0. At some point in its operation FIS0 will have created a set D_F of selected relevant documents. The aim of FIS0 is to ensure that D_F will contain most, and ideally all, C-labeled documents without containing many (and ideally containing none) C'-labeled documents. In other words, the objective of FIS0 is to maximize $|D_F \cap D_C|$ while keeping $|D_F - D_C|$ as small as possible. Note that in the ideal case the set D_F would contain all and only the documents that belong to class C, i.e., the feature selection would have built a perfect classifier.

Let us now examine the criteria for the selection of the next feature to be inserted in the set of relevant features F. For this, assume that the feature f is currently evaluated and D_f, shown in Figure 3, is the set of all documents that contain f. The word f will be inserted into F if its score, measured in line 7 of FIS0, is the highest among all other candidate words. The sets F_1 and F_2, shown in Figure 3, are used to derive a value for the function *score(f)*.



D: The full set of documents
D_C: The documents in D that belong to class C
D_F: The documents in D selected by FIS0
D_f: Documents containing word f.
D'_C: The documents in D that do not belong to class C

$F_1 = (D_f \cap D_C) - D_F$

$F_2 = (D_f \cap D'_C) - D_F$

Figure 3: The document sets involved in evaluating the relevancy of a word f

The values of $|F_1|$ and $|F_2|$ used in line 7 of FIS0 represent the increase in the number of positive and negative examples in D_F respectively, if all the documents that contain the word f are added to D_F. By dividing nominator and denominator by (F_1+F_2) we can show that $score(f)$ essentially represents the ratio of the precision of the word f as a predictor of C in the set $D_f - D_F$ versus the precision of the word f as a predictor of C' in the same set. The set $D_f - D_F$ is the set of documents that contain word f but do not contain any of the previously selected words.

The set F generated by FIS0 contains words that also exist in negative examples and consequently D_F contains negative examples. As new features are added to F, D_F grows and along with D_F, $|D_F \cap D_C|$ and $|D_F - D_C|$ grow as well. As the *score* of the new features inserted into F in later iterations decreases, the $|D_F \cap D_C|$ growth rate decreases while the $|D_F - D_C|$ growth rate increases. This means that each newly added feature adds fewer positive and more negative documents in D_F than its predecessors do. In the extreme case, if all features are added to F D_F will contain all documents in D. The final size of D_F, however, as well as the proportions of $|D_F \cap D_C|$ and $|D_F - D_C|$, are controlled by the two stopping criteria in lines 10 and 12 of FIS0:

1. $S_f >$ min_score, where *min_score* is a user defined constant.
2. $|D_f| \geq c\_support \cdot |Dc|$, where *c_support* is also a user defined constant.

For example a value of *min_score=1* means that FIS0 will stop if all candidate words introduce more C'-labeled than C-labeled documents to D_F. We fixed the second constant to a value of *c_support=0.01*. This requires the frequency of f in the dataset to be higher than a minimum threshold of $0.01 \cdot |Dc|$ to prevent overfitting.

The FIS algorithm described in Figure 4 is a wrapper algorithm that calls FIS0 twice. In the first step, it uses FIS0 to extract F_P and D_{F_P}. F_P is the set of words that are best predictors for class C in D, and D_{F_P} contains all documents from D that have at least one word from F_P. D_{F_P} may be regarded as the union of two sets: $D_{F_P} \cap D_C$ and $D_{F_P} - D_C$. This means that it contains both documents that belong to C and documents that belong to C'. The relative size of $D_{F_P} \cap D_C$ and $D_{F_P} - D_C$ depends on the quality of the words in F_P and the threshold *min_score_pos*. It is obvious that the higher the precision of the words in F_P as predictors of C, the smaller the size of $D_{F_P} - D_C$. In the ideal

```
FIS(W,D,min_score_pos,min_score_neg,c_support,F,D_F)
Input:
  A vocabulary W
  A dataset D of class-labeled documents
  containing words from W
  User defined thresholds: min_score_pos,
  min_score_neg, c_support
Output:
  $F = F_P \cup F_N$
  $D_F = D_{F_P}$
Algorithm:
1.FIS0(C, W, D,  min_score_pos,c_support, $F_P$, $D_{F_P}$).
2.FIS0(C',W, $D_{F_P}$ ,min_score_neg, c_support,$F_N$ , $D_{F_N}$ ).
```

Figure 4: Algorithm FIS

case, $D_{F_P} \cap D_C = Dc$ and $D_{F_P} - D_C = \{\}$. In the real world, however, $D_{F_P} - D_C$ usually contains many documents that might drive a classification system to make wrong decisions. This is the reason why FIS calls FIS0 again to perform a second step of feature extraction.

In the second step FIS0 looks only within D_{F_P} for a set of words that may accurately discriminate documents that belong to C' from documents that belong to C. Our experimental results showed that if we represent the new dataset with the features of F_P alone and feed them to a classifier, the classifier will not be able to accurately learn class *C*. This is why the second iteration that returns F_N, is absolutely necessary.

As a result of the feature selection, FIS produces the transformed data set D_{F_P} where documents are represented with the features from $F_P \cup F_N$. We set ad-hoc values for the parameters *min_score_pos* and *min_score_neg* as 0.01 and 1, respectively. Intuitively *min_score_pos* should be minimal but non-zero to guarantee that even a word that rarely appears in positive documents will be included in F_P. A value of *min_score_neg=1* ensures that a word will not be characterized as negative unless it contributes more new negative cases than positive ones. In our experiments we show that this transformed dataset leads to superior results by a variety of classification algorithms.

4. Experimental evaluation

For our experiments, we used two standard benchmark document collections, the Reuters-21578 and the 20 Newsgroups [2] collection. The classification algorithms we used for the evaluation were NB, LB, TAN and SVM. In the following paragraph, we briefly present the algorithms used in the experiments. Then, we describe the datasets and the experimental setup. We next provide an overview of the experimental results and finally we drill down discussing the performance of the algorithms in more detail.

4.1 Algorithms used

Our experiments evaluated the competitiveness of FIS as a feature and instance selection method. We chose Mutual Information (MI) based feature selection as the alternative for the comparison with FIS. Note that within the Information Retrieval literature Mutual Information is sometimes confusingly called Information Gain. Here we use the standard Information Theory definition. The MI-based feature selection algorithm selects for each class *C* a local dictionary consisting of the *K* words w_t with the highest average mutual information with *C*:

$$MI(C;W_t) = H(C) - H(C \mid W_t) = \sum_{c \in C} \sum_{w_t \in \{0,1\}} P(c,w_t) \log \frac{P(c,w_t)}{P(c) \cdot P(w_t)}$$

The classification task requires that a document may be assigned in none, one or more than one categories.

Feature selection methods influence the performance of classification algorithms in a different way. We used four algorithms: Naïve Bayes (NB) [4], Tree augmented Naïve Bayes Classifier (TAN) [6], Large (or Local) Bayes Classifier (LB) [12] and Support Vector Machines (SVM) [7].

4.2 Dataset description and preprocessing

The 20 Newsgroups contains about 20,000 newsgroup postings from 20 different UseNet groups. Each document belongs to one or more groups. The documents are evenly divided among the classes. We extracted word tokens from the data and removed words that occur only once in the whole collection. All headers from the postings (including the newsgroup header of course) were removed and only the body was used for training or testing. No stemming was used. To reduce the vocabulary size we kept only features that occurred in at least five documents and removed stop-words. The resulting vocabulary contained 12,357 words. For each class we created a training set consisting of the first 80% of the documents and a testing set containing the last 20% of the data set.

The Reuters-21578 Distribution 1.0 [2] consists of 21,578 stories from the 1987 Reuters newswire, each one pre-assigned to one or more of a list of 135 topics. We used the '*Mod Apte*' training-test split that contains 9,603 training and 3,299 test examples. All words were converted to lower case, punctuation marks were removed, numbers were ignored and stop-words were removed. Within Reuters the frequency of the classes is highly skewed. Following a practice popular in the literature, we only used the 10 most popular classes for our experiments.

In both cases, the classification task requires that a document may be assigned in none, one or more than one categories. We followed standard practice and treated the problem as a series of binary classification problems.

4.2.1 Performance measures

We evaluated performance using the standard Information Retrieval measures *recall*, *precision* and *accuracy*. To combine recall and precision with a single-value metric that can be used to derive a total order on the classifiers we use the F_1 measure, defined as follows: $F_1(recall, precision) = \frac{2 \cdot recall \cdot precision}{recall + precision}$

The algorithms were optimized to yield maximum F_1 scores using a validation test consisting of the last 25 per cent of the training stories. This is done with an additional pass over the data to adjust the decision thresholds of the algorithms to maximize F_1.

4.2.2 Results discussion

The positive influence of FIS in the performance of classification algorithms is summarized in Table 1 and Table 2 that list the F1-measure, accuracy and train/test time of the four classifiers described above in the Reuters (Table 1) and 20-newsgroups (Table 2) datasets. The two tables compare FIS as a feature and instance selection method with Mutual Information (MI) based feature selection. MI is a commonly used method for feature selection and it has been shown to yield very competitive results compared to other feature selection methods [17].

In both datasets, there is a significant improvement in the classification quality of all four algorithms as measured by the F1 measure when FIS is used as the feature selection method compared to MI-based feature selection. Interestingly, the performance improvement with FIS is higher for the methods that have the lowest performance when MI is used. We believe that this is attributed to the following two reasons:

- There are fewer complex relationships between the features returned by FIS compared to the features returned by MI, and
- FIS significantly reduces the number of examples retained, compared to the total dataset size.

	NB		TAN		LB		SVM	
	MI	FIS	MI	FIS	MI	FIS	MI	FIS
F1	83.02	89.21	86.75	89.34	86.6	89.29	88.88	89.72
Accuracy	97.14	98.14	97.79	98.21	97.72	98.20	98.16	98.25
Train Time	1.25	0.07	138.7	8.96	126.75	6.3	16.9	4.9
Test Time	0.92	0.1	1.64	0.16	165.9	4.69	1.25	0.8

Table 1: Micro-averaged classification accuracy and time (in sec) for "Reuters 21578"

	NB		TAN		LB		SVM	
	MI	FIS	MI	FIS	MI	FIS	MI	FIS
F1	57.5	69.52	62.6	67.98	59.2	68.63	64.99	66.23
Accuracy	95.5	96.98	96.4	96.89	95.5	97.02	97.01	96.89
Train Time	2.5	0.66	226.2	87.75	1.25	2.78	56	10.1
Test Time	1.36	0.36	2.74	0.71	4.9	0.54	1.65	0.62

Table 2: Micro-averaged classification accuracy and time (in sec) for "20-newsgroups"

NB is known to reach its peak performance under such conditions, whereas classifiers that build more complex models such as TAN, LB and SVM tend to under-perform when the complexity of the dataset is lower.

The simplicity and small size of the datasets after the application of FIS compared to MI is also apparent from the training time of the algorithms. In most cases, there is a significant decrease in the running time, which can reach to an order of magnitude (e.g., for TAN and LB in the Reuters dataset).

In the results of Table 1 and Table 2, we also include the accuracy next to the F1 measure as a measure of classification quality mainly because several papers use it as a measure in algorithm comparisons involving the 20-Newsgroups dataset. Our results, however, show that accuracy is not as appropriate as the F1 measure even in the 20-Newsgroups dataset because of the small number of documents that belong to each class. In this case a default classifier that always classifies to the negative class can achieve 99% accuracy without returning even a single positive document. In what follows, we will only refer to F1 as a measure of classification quality.

The effect of FIS on the four classifiers varies considerably. Overall, NB achieves the highest average F_1 score of 79.37 on the two datasets, followed by LB with 78.96 and TAN with 78.66. SVM is last with an F_1 score of 77.98. Furthermore, training and testing times for NB remain become even lower, making NB the best choice. It is interesting that SVM and TAN, two methods that build complex classification models, do not benefit from the use of FIS. While their performance improves when FIS is used, overall NB and LB enjoy stronger performance improvements from the use of FIS. We attribute this to the fact that the feature/instance selection step with the use of FIS produces a feature and instance set that do not contain complex feature relationships. Naïve Bayes is known to perform best in such situations, whereas other more complex methods perform better when there are more complexities inherent in the dataset. LB has the ability to adapt its model to the complexities of the dataset and it reduces to NB in the extreme case. This may explain the fact that its performance is close to the performance of NB.

Table 3 provides a comparison of FIS and MI in terms of their direct effects on the dataset rather than on the performance of the algorithms that use the dataset. Both methods reduce the feature set size to such an extent, that only a handful of words are used to represent an average document. The main benefit of FIS, however, is its ability to reduce the dataset size by around a significant 80% in these cases. This significantly improves the performance of classifiers applied on the datasets.

	20-Newsgroups		Reuters	
	MI	FIS	MI	FIS
Avg. # of features per document	4.13	5.1	7.31	3.92
Average # of examples per class	11110	2515	8913	1471
Dataset size in KBs	305.53	79.57	390.1	34.3
Feature selection time per class in sec	4.87	7.78	1.76	1.95

Table 3: Effect of FIS and MI on "20 Newsgroups" and "Reuters-21578"

A unique characteristic of FIS is that to the best of our knowledge it is the first algorithm that performs simultaneous feature and instance selection. In contrast, MI and other feature selection methods reduce the number of features, but not the number of instances. The joint instance and feature selection by FIS results in both reduced number of examples in the training dataset (2,515 instead of 11,110 in 20-Newsgroups and 1,471 instead of 8,913 in Reuters) and significantly lower size for the dataset (four-fold reduction in 20-Newsgroups and more than ten-fold reduction in Reuters). Notably, all these improvements come at a very low cost; the running time of FIS is actually only slightly higher that the time of MI.

Table 4 illustrates the per class performance of FIS and MI for the Reuters-21578 collection. It is clear that in the majority of cases the classification accuracy increased significantly. Notice also that the improvement in accuracy is significant in categories with few documents only. Although the cause of this behavior is not clear to us yet, it makes FIS especially useful when small categories exist and the task is to accurately distinguish the very few documents that belong to these categories.

Table 5 lists several characteristics of the datasets that FIS and MI built for the 20-Newsgroups and the Reuters-21578 collections. In Reuters-21578 the F_P feature set consists of an average of only 27 features per class. These 27 features are enough to cover 99.6 per cent of the positive training examples while they are present in only 8.5 per cent of the negative training examples. It is really impressive that just 39 features cover 2,869 of 2,877 positive training examples of the "earn" class. FIS also extracted an average of 62 additional features per class for the F_N feature set. Thus, just 99 features led all the four-classification systems to superior classification accuracy. On the other hand, 20-

Newsgroups is a much tougher domain. The set F_P contains an average of 118 features per class; more than four times larger than in Reuters.

Fl	NB		TAN		LB		SVM	
Class	MI	FIS	MI	FIS	MI	FIS	MI	FIS
Earn	96.90	96.60	96.71	96.75	97.4	97.21	97.66	97.67
Acq	87.33	92.01	90.99	91.93	89.66	91.94	91.47	92.67
money-fx	57.45	73.25	63.70	69.36	68.28	72.77	65.06	75.86
Grain	75.08	92.31	84.21	92.26	83.44	91.56	91.75	90.55
Crude	80.11	83.82	82.89	83.06	82.48	81.84	80.87	82.04
Trade	54.98	65.44	57.73	66.67	61.33	66.37	70.20	66.67
Interest	50.90	70.08	61.03	69.32	62.91	67.91	62.50	68.33
Wheat	69.66	89.61	78.20	88.89	71.19	89.47	84.29	86.45
Ship	81.08	78.82	82.96	79.76	83.16	73.20	77.22	79.04
Corn	52.48	90.32	78	90.16	72.44	90.16	87.72	88.52

Table 4: Per class classification quality for Reuters-21578

class	FIS							MI	
	$\|F_p\|$	$\|F_N\|$	$\|D_c\|$	$\|D_c \cap D_{FP}\|$	$\|D_{Fp}\|$	$\|Dc \cap D_{Fn}\|$	$\|D_{Fn}\|$	$\|D_c \cap D_F\|$	$\|D_F\|$
Avg. (N)	118	159	792	733	2525	20	1453	761	11110
Avg. (R)	27	62	719	716	1471	72	764	716	8913

Table 5: Average characteristics of the datasets generated by FIS for 20-Newsgroups (N) and Reuters-21578 (R).

5. Conclusion

Mutual Information has long been considered the core methodology for feature selection in automated text categorization. In this paper we introduced a new algorithm, named FIS, which combines feature and instance selection. The FIS algorithm is easy to implement, very fast, while it greatly decreases the number of features and training instances required to train accurate text classifiers. Training and testing times for text classification are also decreased, in some cases to an order of magnitude. Furthermore, the accuracy of the well-known Naive Bayes classifier increases to such a degree, that, in many cases, it proves to be equally or more accurate than Support Vector Machines, one of the most accurate classification systems today. A key task that remains to be done is to examine the performance of individual components of FIS; initial experiments have shown that if only the set of positive feature is used the classification performance suffers. We intend to measure the contribution of both the positive and the negative features to the classification accuracy. Similarly we intend to measure the contribution of instance pruning and of feature pruning to the classification accuracy. This will provide a deeper insight to the operation of FIS and may lead to enhancements of the algorithm. We also intend to extend FIS for dealing with multiclass problems and to apply it to structured data in addition to text.

6. References

[1] D. W. Aha, Lazy Learning, (Reprinted from Artificial Intelligence Review 11), Kluwer Academic Publishers, 1997.

[2] S. D. Bay, *The UCI KDD Archive* [http://kdd.ics.uci.edu]. Irvine, CA: University of California, Department of Information and Computer Science, 1999.

[3] L. Blum and P. Langley. "Selection of relevant features and examples in machine learning", *Artificial Intelligence, 97:245--271*, 1997

[4] R. Duda, P. Hart, *Pattern Classification and Scene Analysis*, New York: John Wiley & Sons, 1973.

[5] S. Dumais, J. Platt, D. Heckerman and M. Sahami, "Inductive Learning Algorithms and Representations for Text Categorization", *7th ACM CIKM Conference*, 1998.

[6] N. Friedman, D. Geiger, M. Goldszmidt, "Bayesian Network Classifiers", *Machine Learning*, 29, 131-163, 1997.

[7] T. Joachims, "Text Categorization with Support Vector Machines: Learning with many Relevant Features", *ECML'98*, 1998.

[8] G. John, R. Kohavi and K. Pfleger, "Irrelevant Features and the Subset Selection Problem", *11th ICML*, 1997.

[9] D. Lewis, "Feature Selection and Feature Extraction for Text Categorization", *Speech and Natural Language: in proceedings of a workshop held at Harriman, NY,* Morgan Kaufmann, San Mateo, CA, pp. 212-217, 1992.

[10] D.Lewis and W.Gale. "A sequential algorithm for training text classifiers". *17th ACM-SIGIR Conference*, 1994.

[11] A. McCallum and K. Nigam. "A Comparison of Event Models for Naive Bayes Text Classification", *AAAI/ICML-98 Workshop on Learning for Text Categorization*, 1998.

[12] D. Meretakis, D. Fragoudis, H. Lu and S. Likothanassis, "Scalable Association Based Text Classification", 9th *CIKM*, 2000.

[13] E. Riloff, "Little words can make a big difference for Text Classification", *18th ACM SIGIR Conference*, 1995.

[14] G.Schohn and D. Cohn. "Less is more: Active learning with support vector machines". *17th ICML*, 2000.

[15] D. R. Wilson and T. R. Martinez, "Instance Pruning Techniques", *14th ICML-97*, 1997.

[16] Yang, Y. "Sampling strategies and learning efficiency in text categorization". *AAAI Spring Symposium on Machine Learning in Information Access*, pp. 88-95 1996.

[17] Y. Yang and J. O. Pedersen, "A Comparative Study on Feature Selection in Text Categorization", *14th ICML*, 1997.