Information Theoretic Feature Crediting in Multiclass Support Vector Machines

V.Sindhwani*[*], *P.Bhattacharya*[†], *and S.Rakshit [‡]

Abstract. Identifying relevant features for a classification task is an important issue in machine learning. In this paper, we present a feature crediting scheme for multiclass pattern recognition tasks, that utilizes the ability of Support Vector Machines to generalize well in high dimensional feature spaces. Support Vector learning identifies a small subset of training data relevant for the classification task. They primarily tackle the binary classification problem. This scheme uses relevant examples to identify relevant features for multi-class classification. We present, and employ for this purpose, an information-theoretic measure of classifier performance. This measure addresses the key issue of average rate of information being delivered by the classifier. It provides immunity to sampling bias in the data and sensitivity to pattern of errors made by the classifier. Empirical results on a number of datasets suggest efficient applicability to data with a very large number of features.

Key Words. performance evaluation, feature selection, support vector machines

1 Introduction

A supervised learning algorithm attempts to induce a decision rule from which to categorize examples of different concepts by generalizing from a set of training examples. In trying to construct a classifier for a given categorization problem, there arises a need to empirically evaluate the performance of decision rules induced by different learning algorithms. As we will describe later, commonly used methods

[*]Engineering Physics, Indian Institute of Technology, Bombay, India (vikas7up@ccs.iitb.ernet.in)

[†]Department of Computer Science and Engineering, Indian Institute of Technology, Bombay, India (pb@cse.iitb.ernet.in)

[‡]Center for Artificial Intelligence and Robotics, Bangalore, India (subrata@cair.res.in)

of comparing performance, like classification accuracy, suffer from a number of inadequacies. More elaborate methods have also been suggested [1]. However, most of these methods find applicability to only binary classification problems.

Another important issue concerns the description of the concept itself. Since one does not *a-priori* know what attributes are required for this description, a number of irrelevant features are recorded. Many learning algorithms suffer from the *curse of dimensionality, i.e*, the time requirements for induction may grow very fast as the number of features increases. Irrelevant features in such a case serve only to increase the learning period. They add unnecessary complexity to the underlying probability distribution of the concept label which the learning algorithm tries to capture.

The problem of identification of relevant features for machine learning has been looked into from two different lines of approach: filter and wrapper models [2], [3]. In the filter model [4], [5] feature selection is performed as a preprocessing step to learning. The selected features of the training examples are then presented to the learning algorithm. In the wrapper model [2], feature selection is wrapped around a learning algorithm relying on which the relevant features are selected. Many algorithms under these models employ search routines that makes them prohibitively slow for very high dimensional data.

The contribution of this paper is two-fold. We introduce an information theoretic method to evaluate a classifier so as to better characterize its performance, in general on a multiclass problem. This evaluation is obtained by measuring the average rate of information delivered by the classifier.

Our second contribution is to propose a scheme using this measure to efficiently identify relevant features for a given classification task. To avoid any search routines, our approach is as follows: Measure the information delivered by a classifier trained on the *full set of features* and credit each feature according to its contribution to this information flow by backpropagating this measure. This relevance estimate is therefore based on performance of the classifier on the full set of features as evaluated by our information measures. For such a scheme to be reliable, it is crucial for the classifier to offer resistance to the effect of irrelevant features. Some of its very useful properties make the Support Vector classifier [6] ideal for this purpose. An alternate approach for feature crediting with less tolerant classifiers can also formulated. This would involve an iterative procedure in which the classifier is trained initially with a randomly selected subset of inputs. Features can then be discarded and new features added based on estimates of informativeness provided by our feature crediting scheme. Such a procedure offers the advantage of training the classifier with lower input dimensionality and returning a trained classifier with best input subset. However, it does involve the more expensive task of searching the feature subset space and is a heuristic whose convergence properties cannot be proved. This approach is not explored in this paper.

SVM training identifies a small subset of the training data relevant for the construction of the classification boundary [6]. The information about the relevant features should also be available from the relevant examples; hence the use of SVMs provides the advantage of dealing with a much reduced subset of training data to examine feature relevance. SVMs are able to generalize relatively well in very

high dimensional feature spaces [6], and are less affected by unnecessary excess dimensionality than most other learning algorithms. This property has been used in many applications of SVM classification [7], [8]. However, a good feature selection for SVMs does lead to better performance [9]. In this paper, we explore if SVMs can indeed provide reliable crediting and whether feature selection done accordingly can be used to further improve learning in SVMs.

Our approach to feature selection therefore lies somewhere between the filter and the wrapper models. This scheme is indeed wrapped around the Support Vector learning algorithm but avoids a search routine. The reduced data in the form of relevant examples and features can be presented to any other induction algorithm. Therefore, this can also be considered as a preprocessing step similar to the filter model.

In the following sections, we first very briefly review SVMs. We then introduce an information theoretic method to evaluate performance of a classifier. We present a simple feature crediting scheme based on these concepts and provide experimental results suggesting applicability of the method to various domains.

2 Support Vector Machines

In this section we give a very brief introduction to Support Vector Machines [6], [10]. SVMs realize the following idea: Map the training data $\vec{x} \in \mathcal{X}^d$ into a high dimensional (possibly infinite) space and construct an *optimal separating hyperplane* (OSH) in this space separating examples of two classes. Different mappings $\vec{x} \in \mathcal{X}^d \mapsto \phi(\vec{x}) \in \mathcal{X}^p$ (with $\mathcal{X}$ as the space of real or binary numbers, and p typically much larger than d) construct different SVMs.

The computation of the OSH can be carried out in the original input space by using kernel functions, $K(\vec{x}, \vec{y}) = \phi(\vec{x}).\phi(\vec{y})$. The OSH separates the examples of the two classes by maximizing the *hyperregion* of separation. The OSH is therefore the hyperplane with maximum distance (called *margin* in ϕ space) to the closest image $\phi(\vec{x_i})$ from the training data. The margin can be seen as a measure of the generalization ability: the larger the margin, the better the generalization is expected to be [11]. The decision function given by an SVM is:

$$f(x) = sign(\vec{w}.\phi(\vec{x}) + b) = sign\left(\sum_{i \in support\ vectors} \alpha_i y_i K(\vec{x_i}, \vec{x}) + b\right) \quad (1)$$

where y_i is the class label for input vector $\vec{x_i}$; $y_i = +1$ if $\vec{x_i}$ belongs to the class and -1 otherwise; b is a hyperplane parameter; and α_i is the Lagrange multiplier corresponding to $\vec{x_i}$, arising in the quadratic optimization procedure that attempts to find the OSH. Only examples closest to the OSH contribute a non-zero α to the sum in (1). These examples are called *support vectors*. Thus index i in (1) runs over only the support vectors. In other words, SVMs identify a small subset of the training data relevant for the classification task. For the non-separable case, the optimization problem involves a tradeoff between minimization of classification errors and maximization of margin. The optimization problem is modified by introducing *slack variables* allowing for misclassified vectors [12].

A very useful property of support vector machines is the ability to generalize well in very high dimensional feature spaces [7], [8] making them suitable for our feature crediting scheme. Also, since SVMs do not directly attempt to minimize the error, but try to separate examples in high dimensional space, they are remarkably insensitive to the relative sizes of the training set for a class, as noted in [8]. Most learning algorithms are affected by unbalanced training data and tend to ignore classes with smaller training set sizes, in trying to reduce the error rate.

Multiclass Support Vector Machines

SVMs are designed for binary classification. For multiclass problems several approaches have been proposed [13]. One approach has been to incorporate multiple class labels directly into the quadratic solving algorithm. Another approach is to combine several binary classifiers: *one against one* applies pairwise comparison between classes and in *one against the rest*, one class is compared with the rest.

According to a comparison study in [13], the accuracies of these methods are roughly similar. We therefore chose to implement multiclass SVMs using the lowest complexity approach *i.e.* the *one against the rest*. In this algorithm, for a M-class classification problem, M binary SVMs are trained to separate a class (labelled +1) from the other classes (labelled -1). A test example is labelled according to maximum output among the M SVMs, before taking the sign in (1).

In the next section we introduce an information-theoretic method for evaluation of classifier performance.

3 Information-Theoretic Classifier Performance Evaluation

Consider the results of experiments on two medical domains, 'breast cancer' and 'primary tumor'. The 'naive' Bayes classifier achieved classification accuracies of 79% and 49% respectively [14]. At the first glance, the diagnosis of location of primary tumor is poor while that of recurrence of breast cancer is acceptable.

A closer examination would reveal a different picture. In 'primary tumor' there are 22 possible classes while 'breast cancer' is a binary classification problem. In addition, in 'breast cancer' the prior probability of one of the two classes is 80%, while the prior probability of the major class in 'primary tumor' is just 25%. Therefore, achieving a classification accuracy of 79% in 'breast cancer' is trivial since the induction algorithm needs to learn only the input bias, while 49% in 'primary tumor' is fairly hard to achieve.

A fair evaluation criterion should therefore exclude the effect of prior class probabilities. Multiple evaluation parameters [15] and more involved methods [1] can be used for a detailed picture of classifier performance. This can be cumbersome. Besides, these methods find applicability in only binary classification problems. There exists a need to have a single measure of performance of a classifier for multiclass problems capturing the details.

To be able to compare classifier performance across different domains, one

also needs to compare the difficulty of decision problems. A problem with multiple classes with equal prior probabilities can be expected to be harder than a problem with great differences between prior probabilities.

We now introduce an entropy-based measure of classifier performance that addresses these problems. In this paper we will not be concerned with other parameters like the amount of information available for classification (*i.e.* the training data), the time and computational cost requirements, for performance evaluation. Our model of a classifier is that of black-box with input lines corresponding to the features of an input vector and M output *class indicators*, one of which *fires* when the classifier is shown an input vector.

3.1 Average Rate of Information delivered by the Classifier

Consider an external observer attempting to classify an input vector with just the knowledge of the class distributions *i.e.* the prior probabilities of different classes given an input. When armed also with the knowledge of the performance on a test set, of a trained classifier which has learnt a decision rule, the observer has access to additional information which the classifier delivers. The performance of the classifier can be measured by considering: *By how much does the uncertainty about the class of an input reduce by observing the classification done by the classifier ?* Notice that this approach is significantly different from other criteria like the classification accuracy. For example, consider a trained classifier that classifies all instances of Class 1 as Class 2 and vice versa. Such a classifier has nil accuracy but delivers the very useful information that firing of Class 2 indicator implies an instance of Class 1 with little uncertainty. One therefore needs to measure the *utility* of the classifier in determining the true class of the input.

Given a trained classifier and its performance on a test set, this measure can be mathematically constructed by considering:

• Confusion Matrix **Q**: The performance of a *trained* pattern classifier on *test dataset* is recorded by the elements of this matrix. The element q_{ij} is the number of times a test-set input actually labelled C_i is labelled C_j by the machine.

• In the absence of the classifier, with just the knowledge of input distribution over the classes, an external user faces a certain amount of uncertainty in being able to label a new, unseen and unlabelled instance. This can be used as a measure of the difficulty of a decision problem. The probability that an incoming unseen input (I) has a true label C_i:

$$P(I \in C_i) = \frac{\sum_j q_{ij}}{\sum_{ij} q_{ij}}$$

The uncertainty associated therefore is [16]:

$$H_d(I) = \sum_i -P(I \in C_i) log P(I \in C_i)$$

.

• Given that the incoming input has been labelled C_j by the machine(output vari-

able O), the probability that its actual label was C_i is :

$$P(I \in C_i | O \in C_j) = p_{ij} = \frac{q_{ij}}{\sum_i q_{ij}}$$

The uncertainty in the input classification after observing output C_j therefore is:

$$H_{O_j}(I|O \in C_j) = \sum_i -p_{ij} log(p_{ij})$$

• The probability of observing C_j is:

$$P(O \in C_j) = p_j^{out} = \frac{\sum_i q_{ij}}{\sum_{ij} q_{ij}}$$

The expected uncertainty in the class of an incoming input given the classification done by the machine over the test set, can be calculated by factoring in relative probability of firing of the various class indicators:

$$H_O(I|O) = \sum_j P(O \in C_j).H_{O_j}(I|O \in C_j)$$

• The amount of uncertainty in the input labelling resolved by observing the classification done by the machine therefore is:

$$H_{classifier} = H_d - H_O \tag{2}$$

A good classifier reduces the uncertainty about the class of an input by providing its output to the observer. Therefore, the performance of the classifier can be measured by the difference in the prior and posterior uncertainties. We will call $H_{classifier}$ as *classifier information.*

3.2 Some properties of the entropy based evaluation criterion

1. $H_{classifier}$, like *classification accuracy*, is sensitive to the number of misclassified vectors and not to the margin of error. Measures like root mean square error (RMSE) suffer from the disadvantage of becoming misleading in case of a few large errors.

2. As explained in the previous section, $H_{classifier}$ is able to exclude the effect of prior probabilities in estimating the classifier performance. The prior uncertainty term H_d takes into account the difficulty of the decision problem.

3. $H_{classifier}$ captures a more detailed picture of the classifier performance by providing sensitivity to pattern of errors. Consider the following confusion matrices. Note that a column represents the distribution of answers given by a particular class indicator to instances of different true classes, whereas a row represents how instances of a class are distributed over various indicators.

	(a)			(b)			(c)		
Class Indicators → Actual Class ↓	1	2	3	1	2	3	1	2	3
1	15	0	5	16	2	2	1	0	4
2	0	15	5	2	16	2	0	1	4
3	0	0	20	1	1	18	1	1	48

The classification accuracy for all cases is $50/60 = 83\%$. H_d for case (c) is 0.82 bits indicating little prior uncertainty as compared to $H_d = 1.58$ bits for cases (a) and (b) where the three classes have equal prior probability involving 20 examples in each. The observer can demonstrate good accuracy even without the classifier in (c) by labelling all instances as class 3. As the confusion matrix shows, even without constructing good decision boundaries for class 1 and class 2, the classifier achieves a high classification accuracy. However, $H_{classifier} = 0.06$ bits indicating that the underlying classification task has not been solved by the classifier, whereas $H_{classifier}(a) = 0.96$ bits and $H_{classifier}(b) = 0.78$ bits. The classifier performs better on (a) because with the information of firing of class 1 or 2, the observer can be confident about the class of the input. In (b), firing of class 1 or class 2 maintains slightly greater uncertainty than (a) by sometimes firing for other classes. Notice that even though the class 3 indicator in (b) is more reliable than that in (a), the classifier overall performs better in (a) on account of greater reliability of class 1 and class 2 indicators.

4. Performance of different classifiers on the same domain can be measured by comparing *relative classifier information* $= H_{classifier}/H_d * 100\%$ while classifier information $H_{classifier}$ can be used for comparison across different decision problems.

4 Information Backpropagation

4.1 Crediting Class Indicators

Having obtained the information being delivered by the classifier, we begin to propagate this information flow backward to examine how individual components of the classifying machine contribute to the flow. Consider the question : *How much of the resolution of the input uncertainty by the classification of the machine can be credited to the performance of a particular output class indicator ?* It is desirable that the information credited to each class indicator be such that (a) It reflect the difference in *a-priori* and *a-posteriori* uncertainties. (b) It reflect the frequency of firing of the indicator. (c) The sum across the indicators should add up to $H_{classifier}$.

The first condition ensures that class indicators that fire only for instances of a particular class are given more credit. Class indicators firing for multiple classes maintain observer uncertainty. The second condition ensures that rarely firing class indicators get less credit. Such indicators might be either specializing in rare classes or failing to fire even when the input is from their class. The third condition is a normalization constraint.

The usefulness of the knowledge of the firing of a particular indicator can be measured relative to a hypothetical worst case indicator whose firing only maintains maximum uncertainty about the actual label ,*i.e.*, it only suggests that all possible classifications are equally likely. The uncertainty associated with such an indicator is $logM$ where M is the number of output classes. The uncertainty of an input given that an indicator j has fired, as defined earlier, is H_{O_j}. Thus the usefulness of this indicator measured relative to the worst case indicator can be written as :

$$H_j = H_{classifier} \frac{p_j^{out}(log(M) - H_{O_j})}{\sum_i p_i^{out}(log(M) - H_{O_i})}$$

Note that this credit assignment uses the *worst case a-posteriori* uncertainty instead of *a-priori* uncertainty for individual class indicators to incorporate condition (a). *A-priori* uncertainties for a classification task are taken as the uncertainty in the absence of the classifier. A class indicator carries no meaning in the absence of the classifier. We deal with the issue of *a-priori* uncertainty of a class indicator to mean *uncertainty prior to training.* Intuitively, this would imply the worst-case class indicator as used in the crediting above. This is satisfying also because any crediting scheme with credits according to normalized weightage of the overall performance must have weights guaranteed to be >= 0 for all cases, with no credits (weight =0) for worst case performance.

Crediting Individual SVMs

The information flow through the multiclass SVM can be similarly defined as : $H_{svm} = H_d - H_O$, where the entropy terms can be calculated by observing the performance of the multiclass SVM for M classes, on the test set. Each SVM can therefore be credited for this flow according to the heuristic measure introduced above :

$$H_j = H_{svm} \frac{p_j^{out}(log(M) - H_{O_j})}{\sum_i p_i^{out}(log(M) - H_{O_i})} \tag{3}$$

where j is the j^{th} SVM separating class j from the remaining classes and p_j^{out}, as defined earlier, is its frequency of firing.

Crediting Features

To examine the informativeness of the features, we backpropagate the information further onto the input side. To credit the features with their contribution to the information flow there are two considerations to be guided by: (a) The credit must be proportional to the degree of influence the feature has on the performance of an SVM and (b) The total credit assigned to the features should add up to $H_{classifier}$.

The generalization performance of an SVM depends on the margin. The margin of a trained SVM is proportional to the inverse root of :

$$w^2 = \sum_{ij} \alpha_i \alpha_j y_i y_j K(\vec{x_i}, \vec{x_j}) \tag{4}$$

where i and j run over the set of support vectors; α_i is the Lagrange multiplier corresponding to i^{th} support vector; y_i is the class label on $\vec{x_i}$, *i.e*, it is $+1$ or -1 depending on whether the support vector belongs to the class for which the SVM is trained.

Intuitively, the absolute value of the *derivative* of inverse-margin-square with respect to k^{th} feature provides the degree of influence of the k^{th} feature on the generalization performance on the SVM. This can be written as :

$$\mathcal{D}_k(w^2) = \sum_i |\sum_j \alpha_i \alpha_j y_i y_j \frac{\partial K(\vec{x_i}, \vec{x_j})}{\partial x_j^k}| \tag{5}$$

Here x_j^k is the k^{th} component of $\vec{x_j}$ and the partial derivative is evaluated at $\vec{x_i}$. For example, if the kernel $K(\vec{x_i}, \vec{x_j}) = (\vec{x_i}.\vec{x_j})^2$, this derivative becomes $2 * (\vec{x_i}.\vec{x_j})x_i^k$. This is similar to several methods proposed [17] for feature extraction in neural networks to evaluate relevance of a variable by the derivative of the output with respect to this variable. The sum of the absolute value of derivatives has been used in [18]. These methods involve the sum over the entire training set. With SVMs, the sum in (5) is only over the set of support vectors.

For multiclass SVM , the features can therefore be credited for the overall information flow according :

$$Credit(k) = \sum_r H_r \frac{\mathcal{D}_k^r}{\sum_p \mathcal{D}_p^r} \tag{6}$$

where r runs over the output classes ; p runs over the features. $Credit(k)$ gives the credits assigned to feature k. $\mathcal{D}_p^r$ is the derivative of the margin of the r^{th} SVM (trained to separate class r from the rest) with respect to feature p. H_r can be calculated from (3).

5 A Feature Crediting Algorithm

Based on these concepts, the following feature crediting algorithm can be formulated. We will call this the SVM Information Backpropagation or *SVM-Infoprop* algorithm:

1. For an M-class classification problem , train M Support Vector Machines to build a multiclass classifier according to *one-against-rest* and using all the features. This will identify the set of support vectors and corresponding Lagrange multipliers for each SVM.
2. Pass the test data : Construct confusion matrix.
3. Calculate information flow through the classifier from (2).
4. Credit each SVM according to (3).
5. Use the set of support vectors and corresponding Lagrange multipliers to calculate derivatives of the margin with respect to each feature according to (5) for each SVM.

6. Assign credits to each feature according to (6).

SVM-Infoprop is a feature crediting algorithm that can be used for feature selection by employing a user-specified credit threshold. Features with credits below the threshold can be discarded. The threshold can be estimated from the credits *versus* features plot. It is fair to expect irrelevant features to contribute little to the information delivered by the SVMs which are resistant to their effect. Hence, such credit assignment should distinguish well between irrelevant and relevant features.

6 Experiments

To test the *SVM-Infoprop* algorithm, we tried feature crediting on several multiclass artificial and real-world datasets. These datasets include : a synthetic dataset that we constructed, LED-24, Waveform-40, DNA, Vehicle, and Satellite Images (SAT) drawn from the UCI repository [19]; and three datasets which are a subset of the Reuters document collection [20]. Table 5 lists the details of these datasets. We first examine the informativeness of features using *SVM-infoprop* in each of these datasets and then tabulate a comparison across different domains.

Synthetic Dataset

This 3 class and 9 attribute dataset was generated to test the algorithm. To introduce input bias, an example had the probabilities 0.5, 0.3 and 0.2 of belonging to Class 1, 2, or 3 respectively. 200 training and 100 test examples with this bias were generated. An example of Class i ($i \in 1, 2, 3$) had feature i drawn from a normal distribution of mean i and variance 0.1, and features j ($j \in 1, 2, 3$ and $j \neq i$) drawn from a normal distribution with mean 0 and variance 0.1. The remaining 6 features of all examples were just noise drawn from a Gaussian of variance 20 around the mean 0. The relevant features, therefore, in this problem are 1, 2, 3.

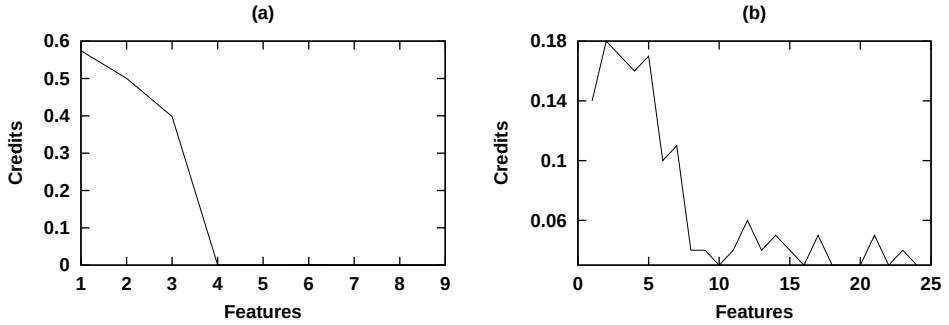
Linear Support Vector Machines (C=200) were trained for this problem. The classification accuracy obtained was 100% on the test set. The credits assigned by the *SVM-Infoprop* are shown in the Figure 1. The relevant features are very sharply identified. Feature 1 gets maximum credit on account of being relevant for the most informative SVM, in this case decided purely by the input bias.

LED Dataset

This 10-class dataset, drawn from the UCI repository, consists of 200 training and 500 test examples of 24 binary-valued features each. The first 7 features are relevant, and correspond to LEDs on a display which are either on or off. The remaining 17 features are randomly distributed. The problem becomes more difficult because of the introduction of noise. Each relevant attribute has 10% probability of having its value inverted.

Linear SVMs (C=200) were trained; Figure 1 shows the credits assigned to the features by *SVM-Infoprop*. The algorithm selects the 7 relevant features at a credit

Figure 1. *Credit Assignment for (a) Synthetic Dataset and (b) LED Dataset*



threshold of 0.09. With all features, a classification accuracy of 67% is obtained on the test set. On training with best 14 features, the accuracy improved to 72.4% and further to 73% with the best 7. The relative classifier information (Hsvm/Hd * 100) improved from 50% with full set of features, to 55% with 14 best features and 57% with best 7.

Waveform40 Dataset

The Waveform dataset has been applied to examine how well a feature selection algorithm works in the presence of high number of irrelevant features [21],[22]. The dataset contains 300 examples with every example assigned to one of three classes. Each example has 40 continuous feature values. The dataset is generated such that features $\{1, 2, ...21\}$ are necessary for class separation, whereas features $\{22, 23, ..., 40\}$ have random values. A training size of 200 and test size of 100 examples was used in the experiments.

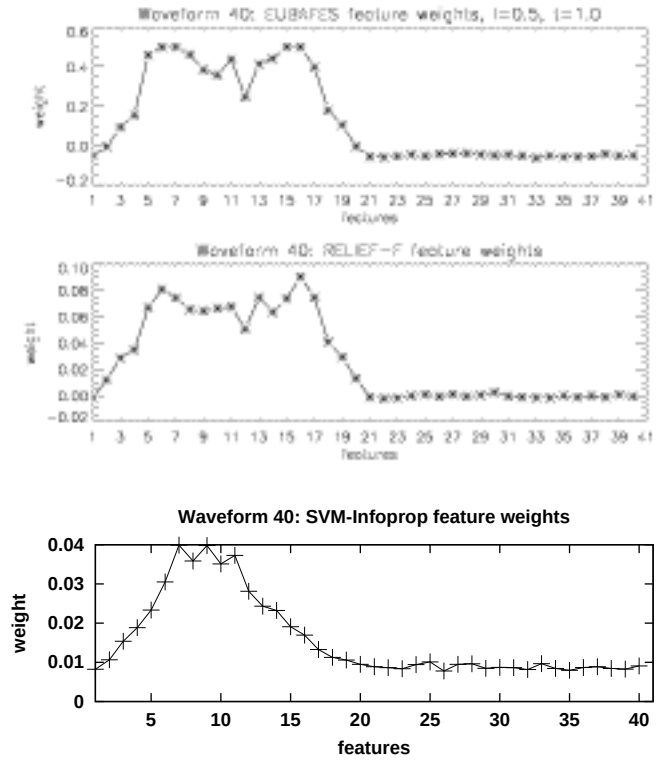
Figure 2 shows credits on the 40 features of the dataset by popular feature weighting algorithms *EUBAFES* [22] and *RELIEF-F* [23] and also the credits assigned by *SVM-Infoprop* (Linear SVMs; C = 200). There exists a strong correlation between the weights assigned by *RELIEF-F, EUBAFES* and *SVM-Infoprop.*

As mentioned in [22] , *EUBAFES* fails to select feature 1 and 21 or 1, 2, 20 and 21 depending on certain parameters of the algorithm. As explained in that work, this may happen, contrary to the design of the dataset, because of presence of noise in every feature. *SVM-Infoprop* shares the same difficulty and is able to select features 2 to 19 at a credit threshold of 0.0105.

DNA Dataset

The DNA dataset is a 3 class problem with 180 binary valued features encoding DNA sequences. The problem posed in this dataset is to recognize the boundaries between exons (the parts of the DNA sequence retained after splicing) and introns (the parts of the DNA sequence that are spliced out). The classification task therefore is : given a position in the middle of a window of DNA sequence elements (called nucleotides

Figure 2. *Feature Weighting in Waveform 40 dataset by EUBAFES, RELIEF-F, and SVM-Infoprop*



or base-pairs), decide if this is an intron-exon boundary, exon-intron boundary or neither.

The credits given to each feature by *SVM-Infoprop* implemented by training linear SVMs (C=2000) for the three classes, are shown in Figure 3.

Attributes closest to the junction towards the middle of the sequence are more relevant [24]. This implies greater relevance of features around 60 to 120. As can be seen from Figure 3, the feature crediting scheme is able to identify these attributes.

Based on these credits, the best subset of 80 and 30 features were selected. These numbers were taken so as to compare results with [25]. The training(2000 examples) and test(1186 examples) sets were also the same. A comparison of best classification accuracies for the filter method introduced in [25] and used with Naive-Bayes [27] and C4.5 [28] induction algorithms is shown in Table 1. The multiclass SVM trained with the selected features shows improvement in both the classifier accuracy and the information delivered as compared to performance on the full set of features.

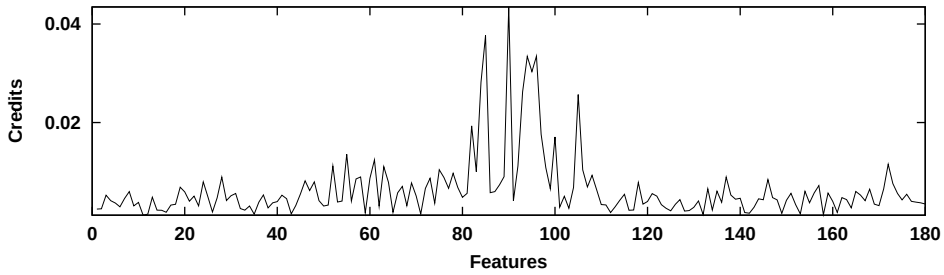
As a matter of comparison, the wrapper introduced in [26] selected 11 features with induction algorithms Naive-Bayes and C4.5. This took several hours.

Table 1. *Results for DNA dataset: Classification Accuracy with Naive-Bayes, C4.5 and SVMs. The SVM column also lists Relative Classifier Information (R.C.I)*

# Features	Naive-Bayes	C4.5	SVM (Classification Accuracy, R.C.I)
180	93.3%	92.3%	94.68%, 76.85%
Features	Naive-Bayes FS	C4.5 FS	SVM-Infoprop (Classification Accuracy, R.C.I)
80	94.9%	93.4%	96.12%, 81.95%
30	93.8%	93.8%	95.36%, 79.70%

Training SVMs with 180 features, crediting features, and retraining SVMs with best 11 features took a total time of about 70 min (P-III, 500MHz) with similar classification accuracy. The implementation of SVMs [29] we used, does not employ Platts *Sequential Minimal Optimization* (SMO) known to be an order of magnitude faster [30],[31] SVM training algorithm for problems such as this where input data is largely sparse and linear SVMs are employed.

Figure 3. *Credit Assignment for DNA dataset*



Vehicle and Satellite Datasets

The Vehicle dataset is a multiclass pattern recognition problem of classifying a given silhouette as one of four types of vehicle. There are 18 features. The Satellite dataset is a 6-class and 36-feature dataset containing Landsat satellite data. We tested *SVM-Infoprop* for non-linear SVMs with these datasets. Support Vector Machines with polynomial kernels of degree 2 were trained with 564 examples and tested with 282 examples for the Vehicle data. For the Satellite data, the training set consists of 4435 examples and the test set consists of 2000 examples. Class 4 was trained with the Anova Kernel (k(x,y)=$(\sum_i exp(-\gamma(x_i - y_i)))^d$) of degree $d = 1$ and $\gamma = 0.01$. The remaining classes were trained with polynomial kernels of degree 2. Since the outputs are in different feature spaces, the simple normalization technique of dividing the outputs by the margin [32] was done before making comparisons.

The performance of the SVMs on full set of features and on subsets of features ranked best by *SVM-Infoprop* is shown in Table 2 for the Vehicle dataset and Table 3 for the Satellite dataset. Also shown for comparison is the performance best among linear feature extractors like *PCA* (Principal Component Analysis) and *LDA* (Linear Discriminant Analysis), and Mutual Information based feature selectors like *MIFS* (Mutual Information Feature Selector), *SMIFE*(Separated Mutual Information Feature Extractor) and *MMIP* (Maximum Mutual Information Projection) [33]. The *SVM-Infoprop* column lists classification accuracy and relative classifier information (R.C.I). The accuracies for the best among *PCA, LDA, MIFS, MMIP* and *SMIFE* are approximate.

Table 2. *Results for Vehicle dataset: Classification Accuracy and Relative Classifier Information (R.C.I)*

# Features	SVM-Infoprop (Classification Accuracy, R.C.I)	Best classification accuracy among PCA, LDA, MIFS, MMIP, SMIFE
18	49.64%, 24.13%	50.00%
12	70.21%, 47.52%	59.00%

Table 3. *Results for Satellite dataset: Classification Accuracy and Relative Classifier Information (R.C.I)*

# Features	SVM-Infoprop (Classification Accuracy, R.C.I)	Best Classification Accuracy among PCA, LDA, MIFS, MMIP, SMIFE
36	78.10%, 63.80%	80.00%
19	79.00%, 65.05%	80.00%

Note that mutual information based methods like *MIFS, MMIP*, and *SMIFE* require quantization of continuous input variables. This demands computational complexity and data requirements. By dealing with discrete output variables, Information Backpropagation eliminates this need.

Text Classification : Reuters Dataset

The ability of SVMs to generalize well in very high dimensional spaces allows us to use *SVM-Infoprop* in the domain of information retrieval. These applications involve very large number of features.

We constructed three subsets of the Reuters collection. The first subset Reuters-1 comprises of articles on the topics *coffee, iron-steel* and *livestock*. These topics are not likely to have many meaningful overlapping words. The second subset Reuters-2 contains articles on *reserves, gold* and *gross national product*, likely to have similar words used in different contexts across these topics. Both these subsets were applied in [25]. Reuters-3 was constructed to examine the performance of *SVM-Infoprop* on even larger dimensionality. It contains articles on the five most frequent Reuters categories : *earn, acq, money-fx, grain* and *crude*. Each article was binary-encoded where each feature denoted whether a particular word occurred in

the article or not. As a preprocessing step, all articles with more than 30% content numeric were excluded from the dataset and words occurring less than 3 times in each dataset were eliminated to remove extremely rare words.

Mutual Information is among the most popular techniques (used in [34]) for feature reduction in text classification. Features that have the highest average mutual information with the class variable are selected. In Table 3 we tabulate the performance of SVMs on the full set of features and features selected by *Mutual Information* and *SVM-Infoprop* (Linear SVMs with C = training set size). In each case, we performed drastic feature selection selecting roughly the top 20% features as ranked by these two different techniques.

As Table 4 shows, with both *Mutual Information* and *SVM-Infoprop*, the classifier maintains acceptable levels of performance on drastic feature reduction. As for resource consumption, the major component of *SVM-Infoprop* runtime is the training time of the SVMs with all the features. With Platts SMO, SVMs have been reported to very impressive training times on the text classification task [34]. For the largest dataset, Reuters-3, the training time had an average of 11 min per SVM. The feature crediting module is very quick since it processes only the support vectors, which in Reuters-3, average 475 per class. As in the DNA dataset which involves sparse input and linear SVMs, we expect this to be up to an order of magnitude faster with SMO indicating that *SVM-Infoprop* is a useful feature crediting technique for text categorization. By comparison, wrappers as in [2] would roughly take the order of thousands of hours for similar feature selection. In real-valued feature representation, this also eliminates the need for quantization of inputs variables, as required in mutual information based techniques.

Table 4. *Performance on the Reuters dataset: Classification Accuracy and Relative Classifier Information (R.C.I)*

Dataset Training Set Size, Test Set Size	# Features	Mutual Information Classification Accuracy, R.C.I	SVM-Infoprop Classification Accuracy, R.C.I
Reuters-1 (199, 113)	2225 (full set)	96.46%, 87.42%	96.46%, 87.42%
	431	99.12%, 96.18%	98.23%, 93.45%
Reuters-2 (193, 162)	2344 (full set)	94.44%, 77.95%	94.44%, 77.95%
	458	95.67%, 83.72%	93.83%, 76.52%
Reuters-3 (3257, 2912)	8167 (full set)	93.00%, 78.80%	93.00%, 78.80%
	1167	92.00%, 76.77%	93.10%, 79.00%

7 Comparison across different domains

Table 5 lists the details of the datasets used for experimentation and the performance of the multiclass Support Vector Machine with all features in terms of classifier information and classification accuracy.

The multiclass SVM performs unimpressively on the LED and Satellite (SAT)

datasets according to classification accuracy. However, this does not take into account the difficulty of the decision problem and the prior-probabilities. LED and SAT involve maximum number of classes and the least probabilities for the major class in the collection of datasets used, which makes them the most difficult problems. Achieving even 67% and 78.1% accuracy respectively on these problems is not an easy task. This can be seen by the highest H_d values for these datasets. In fact, as classifier information H_{svm} brings out, the multiclass SVM performs best in these datasets. By comparison, high classification accuracy for the Synthetic and the DNA datasets overlooks the fact that the major class in these 3-class datasets occurs half the time.

Table 5. *Datasets used for Experiments: A comparison of performance on full set of features*

Dataset	# Classes	# Features	Probability of major class	H_d	H_{svm}	Classification Accuracy
Artificial						
Synthetic	3	9	50%	1.48	1.48	100%
LED	10	24	10%	3.31	1.67	67%
Waveform40	3	40	43%	1.54	0.62	79%
Real World						
DNA	3	180	51%	1.49	1.15	94.68%
Vehicle	4	18	25.77%	1.99	0.48	49.66%
SAT	6	36	23.5%	2.5	1.60	78.1%
Text						
Reuters-1	3	2225	36.28%	1.58	1.38	96.46%
Reuters-2	3	2344	43.21%	1.52	1.18	94.44%
Reuters-3	5	8167	38.15%	2.00	1.58	93%

8 Conclusion

Classifier Information, as a performance evaluation criterion, effectively addresses many problems with measures like classification accuracy. It takes into account prior-probabilities of classes and captures performance details in a single measure for multiclass problems.

SVM Information Backpropagation is an output-side heuristic for feature crediting. It eliminates the need for quantization of continuous features as required in many feature selection techniques. SVMs provide the relevant examples for classification in a dataset. Information Backpropagation explores feature relevance from these relevant examples. This scheme is able to identify relevant features without expensive search routines in a variety of classification tasks, including those that involve very high dimensionality and unbalanced data. As many of our experiments have indicated, feature selection using this technique can improve learning in Support Vector Machines.

Acknowledgements

We are grateful to Matthias Scherf for permitting us to include the plots (Figure 5 in [22]) in Figure 2 for *EUBAFES* and *RELIEF-F*.
For text processing, we used *Rainbow :* McCallum, Andrew Kachites, (1996), *Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering*, http://www.cs.cmu.edu/~mccallum/bow

References

[1] F. Provost and T. Fawcett, *Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions*, in Proceedings of the Third International. Conference on Knowledge Discovery and Data Mining, AAAI Press, 1997, pp. 43-48.

[2] G. H. 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, pp. 121-129.

[3] M. Dash and H. Liu, *Feature Selection for Classification,* Intelligent Data Analysis, 1(3), 1997, http://www-east.elsevier.com/ida/browse/0103/ida00013/article.htm

[4] K. Kira and L.A. Rendell, *The feature selection problem: traditional methods and a new algorithm*, Proceedings of the Tenth National Conference on Artificial Intelligence, Menlo Park, CA: AAAI Press/Cambridge, MA: MIT Press, 1992, pp. 129-134.

[5] H. Almuallim and T. G. Dietterich, *Learning with many irrelevant features*, in Proceedings of the Ninth National Conference on Artificial Intelligence, MIT Press, 1991, pp. 547-552.

[6] V.N. Vapnik, *The Nature of Statistical Learning Theory.* Springer-Verlag, New York, 1995.

[7] T. Joachims, *Text Categorization with Support Vector Machines: Learning with Many Relevant Features*, Proceedings of the Tenth European Conference on Machine Learning ECML98, 1998, pp. 137-142.

[8] H. Drucker, D. Wu and V. Vapnik, *Support Vector Machines for Spam Categorization*, IEEE Transactions on Neural Networks , vol. 10, no. 5, 1999, pp. 1048-1054.

[9] J. Weston , S.Mukherjee, O. Chapelle, M. Pontil, T. Poggio and V. Vapnik, *Feature Selection for SVMs*, in Sara A Solla, Todd K Leen, and Klaus-Robert Muller, editors, Advances in Neural Information Processing Systems 13, MIT Press, 2001.

[10] C.J.C. Burges, *A tutorial on support vector machines for pattern recognition*, Data Mining and Knowledge Discovery,2(2), 1998, pp. 955-974.

[11] P. Bartlett and J. Shawe-Taylor, *Generalization Performance of Support Vector Machines and Other Pattern Classifiers*, in 'Advances in Kernel Methods - Support Vector Learning', Bernhard Scholkopf, Christopher J. C. Burges, and Alexander J. Smola (eds.), MIT Press, Cambridge, USA, 1998.

[12] C. Cortes and V. Vapnik, *Support vector networks*, Machine Learning, 20 (1995), pp.1-25.

[13] J. Weston and C. Watkins, *Support vector machines for multiclass pattern recognition*, in Proceedings of the Seventh European Symposium On Artificial Neural Networks, April 1999.

[14] I. Bratko and I. Kononenko, *Learning diagnostic rules from incomplete and noisy data*, in B. Phelps, editor, Interactions in Artificial Intelligence and Statistical Methods, Gower Technical Press, Aldershot, UK, 1987, pp. 142-153.

[15] S. Weiss, R. Galen, and P.Tadepalli, *Optimizing the Predictive Value of Diagnostic Decision Rules*, in Proceedings of the National Conference of AAAI-87, Seattle, 1987, pp.521-526.

[16] C.E Shannon and W. Weaver, *The mathematical theory of communications*, Urbana, IL: The University of Illinois Press, 1949.

[17] P. Leray and P. Gallinari, *Feature selection with neural networks*, Technical report LIP6 1998/012, LIP6, 1998, Behaviormetrika.

[18] D.W. Ruck, S.K. Rogers, and M. Kabrisky, *Feature Selection Using a Multi-Layer Perceptron*, In J. Neural Network Comput., 2 (2), 1990, pp. 40-48.

[19] P.M. Murphy and D.W. Aha, *UCI repository of machine learning databases*, 1995, http://www.ics.uci.edu/~mlearn/MLRepository.html

[20] Reuters Collection, 1995, http://www.research.att.com/~lewis/reuters21578.html

[21] D. Wettschereck, D.W. Aha and T. Mohri *A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms*, Artificial Intelligence Review, 11 (1997), pp.273-314.

[22] M Scherf and W. Brauer, *Feature selection by means of a feature weighting approach*, Technical Report FKI-221-97, Institut fur Informatik, Technische Universitat Munchen, 1997.

[23] I. Kononenko, *Estimating attributes: analysis and extensions of relief*, in Proceedings of the Seventh European Conference on Machine Learning. Springer Verlag, 1994, pp. 171-182.

[24] D. Michie, D. Spiegelhalter, C.C Taylor *Machine Learning, Neural and Statistical Classification*, Ellis Horwood, 1994 pp. 158-161.http://www.amsta.leeds.ac.uk/~charles/statlog/chap9.ps.gz

[25] D.Koller and M.Sahami *Toward optimal feature selection*, in Proceedings of the Thirteenth International Conference on Machine Learning, Morgan Kaufmann Publishers, Inc., 1996, pp 284-292.

[26] R. Kohavi, and G.H. John, *Wrappers for feature subset selection*, Artificial Intelligence, 97(1-2), 1997, pp.273-324.

[27] R.Duda and P.Hart, *Pattern Classification and Scence Analysis*, Wiley, 1973.

[28] J.R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, Inc., Los Altos, California, 1993

[29] mySVM available at http://www-ai.cs.uni-dortmund.de/SOFTWARE/MYSVM

[30] John C. Platt, *Using Analytic QP and Sparseness to Speed Training of Support Vector Machines*, Neural Information Processing Systems 11, MIT Press, 1999

[31] O.L. Mangasarian and D.R. Musicant, *Successive overrelaxation for support vector machines*, IEEE Transactions on Neural Networks, 10(5), 1999, pp. 1032-1037.

[32] E. Mayoraz and E. Alpaydin, *Support Vector Machines for Multiclass Classification*, Proceedings of the International Workshop on Artifical Neural Networks (IWANN99), IDIAP Technical Report 98-06, 1999

[33] K.Bollacker and J.Ghosh, *Linear feature extractors based on mutual information*, in Proceedings of the Thirteenth International Conference on Pattern Recognition, August, 1996, pp. IV:720-24.

[34] S. T. Dumais, J. Platt, D. Heckerman, and M. Sahami, *Inductive learning algorithms and representations for text categorization*, in Proceedings of ACM-CIKM98, Nov. 1998, pp. 148-155.