Machine Learning, xx, 1–24 (xxxx)
© xxxx Kluwer Academic Publishers, Boston. Manufactured in The Netherlands.

Machine Learning for the Detection of Oil Spills in Satellite Radar Images

MIROSLAV KUBAT, ROBERT C. HOLTE, STAN MATWIN

{mkubat,holte,stan}@csi.uottawa.ca

Department of Computer Science
University of Ottawa
150 Louis Pasteur, Ottawa
Ontario, K1N 6N5 Canada

Editor:

Abstract. During a project examining the use of machine learning techniques for oil spill detection, we encountered several essential questions that we believe deserve the attention of the research community. We use our particular case study to illustrate such issues as problem formulation, selection of evaluation measures, and data preparation. We relate these issues to properties of the oil spill application, such as its imbalanced class distribution, that are shown to be common to many applications. Our solutions to these issues are implemented in the Canadian Environmental Hazards Detection System (CEHDS), which is about to undergo field testing.

Keywords: Inductive learning, classification, radar images, methodology

1. Introduction

In this paper we describe an application of machine learning to an important environmental problem: detection of oil spills from radar images of the sea surface. We cover the application cycle from the problem formulation phase to the delivery of a system for field testing. The company that sponsored this work, Macdonald Dettwiler Associates, has just begun the final phases of the cycle — field testing, marketing, and deployment. This paper focuses on the research issues that have arisen during the development of the Canadian Environmental Hazards Detection System (CEHDS). These issues cover the entire gamut of activities related to machine learning, from initial problem formulation, through methodology design, to the usual technical activites. For most of the issues, including the technical ones, we found few pertinent studies in the research literature. The related work we did find was usually by others working on a particular application. The primary purpose of this paper is to present to the machine learning research community a set of open research issues that are of general importance in machine learning applications. We also present the approach taken to these issues in our application.

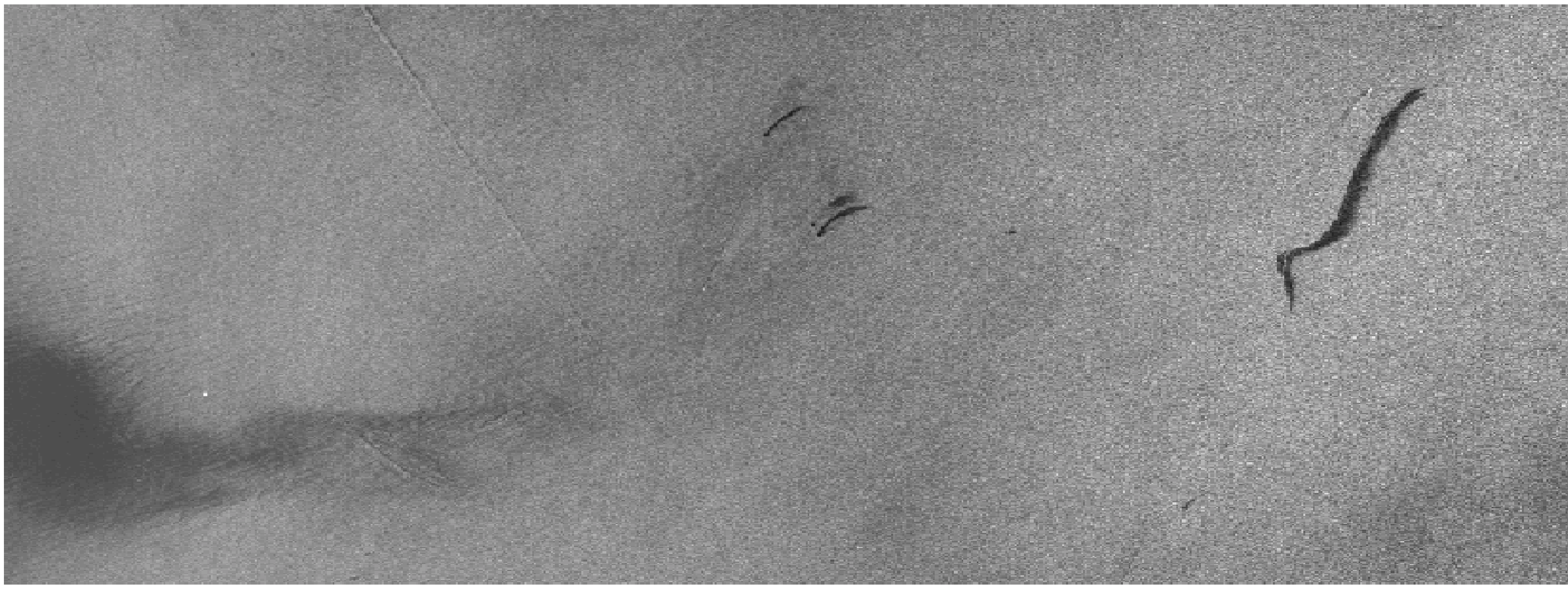


Figure 1. An example of a radar image of the sea surface

1.1. The Application Domain

Only about 10% of oil spills originate from natural sources such as leakage from sea beds. Much more prevalent is pollution caused intentionally by ships that want to dispose cheaply of oil residues in their tanks. Radar images from satellites such as RADARSAT and ERS-1 provide an opportunity for monitoring coastal waters day and night, regardless of weather conditions. Oil slicks are less reflective of radar than the average ocean surface, so they appear dark in an image. An oil slick's shape and size vary in time, depending on weather and sea conditions. A slick usually starts out as one or two slicks that later break up into several smaller slicks. Several natural phenomena (e.g., rain, algae) can closely resemble oil slicks in radar images. They are called *lookalikes.*

Figure 1 shows a fragment of a SAR (Synthetic Aperture Radar) image of the North Sea with an oil slick in it. The full image consists of 8,000x8,000 pixels, with each pixel representing a square of 30x30m; the fragment shown here is approximately 70×50 kilometers. The oil slick is the prominent elongated dark region in the upper right of the picture. The dark regions in the middle of the picture and the lower left are lookalikes, most probably wind slicks (winds with speeds exceeding 10m/sec decrease the reflectance of the radar, hence the affected area looks darker in a radar image).

1.2. Previous Work on Oil Spill Detection

Since the early days of satellite technology and SAR there have been attempts to detect oil spills from radar images. The state of the art is represented by the pre-operational service for identifying oil spills in ERS-1 images that has been offered since 1994 by the Tromso Satellite Station (TSS) in Norway. This service is entirely manual: humans are trained to distinguish oil spills from nonspills in satellite images. TSS recognizes the desirability of having an automatic system to reduce and prioritize the workload of the human inspectors, and has supported research to develop systems for this purpose. This research has recently produced two systems (Solberg & Solberg, 1996; Solberg & Volden, 1997), but neither has yet been incorporated into TSS's service. Our system was developed in parallel with, and independently of, these two systems.

The system described by Solberg & Solberg (1996) uses learning to produce a classifier in the form of a decision tree. As this system is similar to ours in many ways, it will be reviewed in detail in the discussion of our results. TSS's most recent automatic classification system (Solberg & Volden, 1997) is more knowledge intensive. The classifier is a statistical system in which the prior probability of a region being an oil spill is computed using a domain theory that relates features of the region to the prior. The prior is then combined with a Gaussian classifier that has been learned from training data. The system performs very well, correctly classifying 94% of the oil spills and 99% of the nonspills. The system relies heavily on the knowledge of the wind conditions in the image, and it was necessary for the TSS team to develop techniques for inferring this from the image. This crucial piece of information is not available to our system, as we do not at present have methods for inferring wind conditions.

Elsewhere a group of environmental scientists and remote sensing experts have developed a preliminary model of properties of an oil spill image. The model, expressed as a decision tree (Hovland et al., 1994), uses attributes such as the shape and size of the dark regions in the image, the windspeed at the time when the image was taken, the incidence angle of the radar beam, proximity to land, etc. The model has been evaluated on artificial data from a controlled experimental slick, as well as on data from a SAR image of a real slick. Their conclusion was that the model performed well on the artificial data, but was inconsistent with the current physical theory of slick reflectance, and did not agree with the SAR images.

A similar problem of classification of ice imagery into age groups has received attention in the image processing and remote sensing literature. Heerman & Khazenie (1992) used a neural network trained by backpropagation to classify Arctic ice into "new" and "old". Haverkamp et al. (1994) developed a rule based expert system, in which the rules were acquired from experienced human experts. The performance of this system exceeded by some 10% the accuracy of previous systems which relied on the brightness of pixels for classification, without any use of symbolic features describing higher level attributes of the classified objects.

2. Task Analysis

An oil spill detection system based on satellite images could be an effective early warning system, and possibly a deterrent of illegal dumping, and could have significant environmental impact. Oil spill detection currently requires a highly trained human operator to assess each region in each image. A system that reduced and prioritized the operator's workload would be of very great benefit, and the purpose of our project was to produce such a system. CEHDS is not intended for one specific end user. It is to be marketed worldwide to a wide variety of end users (government agencies, companies) with different objectives, applications, and localities of interest. It was therefore essential that the system be readily customizable to each user's particular requirements and circumstances. This requirement motivates the use of machine learning: the system will be customized by training on examples of spills and nonspills provided by the user, and by allowing the user to control the tradeoff between false positives and false negatives. Unlike many other machine learning applications (e.g., the fielded applications described by Langley and Simon, 1995), where machine learning is used to develop a classifier which is then deployed, in our application it is the machine learning algorithm itself that will be deployed.

The input to CEHDS is a raw pixel image from a radar satellite. Image processing techniques are used to normalize the image in certain ways (e.g., to correct for the radar beam's incidence angle), to identify suspicious dark regions, and to extract features (e.g., size, average brightness) of each region that can help distinguish oil spills from lookalikes. This part of the system was developed by Macdonald Dettwiler Associates, a company specializing in remote sensing and image processing. The output of the image processing is a fixed-length feature vector for each suspicious region. During normal operation, these feature vectors are fed into a classifier to decide which image, and which regions within an image, to present for human inspection. The operator then makes the final decision about what response is appropriate.

The classifier is created by the learning algorithm distributed as part of CEHDS. It is the development of this learning system that is the focus of this paper. The learner's input is the set of feature vectors describing the dark regions produced by the image processing subsystem. During training the regions are classified by a human expert as oil slicks and lookalikes (these classifications are imperfect: on some occasions, the expert was not quite sure whether or not the region was an oil slick, and the class labels can thus be erroneous). The learner's output is a classifier capable of deciding whether a specific dark region is an oil spill or not.

The output of the system was to a large extent determined by the requirements set by Macdonald Dettwiler Associates. Early in the design process it was decided that a unit of output will be an original satellite image, with regions classified as spills highlighted with a coloured borderline. The total number of images presented for inspection must not be too large. On the other hand, the fewer images presented, the greater the risk that an actual oil slick will be missed. Users should therefore have control over the system, so they can easily vary the number of images presented to them. The classifier should be provided with a parameter whose one extreme value ensures that the user sees all images (no matter whether they contain oil slicks or not), and whose other extreme value totally blocks the inspection. The intermediate values represent the degree of "confidence" the system must have in the classification of a particular region before the entire image containing the region will be presented for inspection. When an image is presented, the system highlights all the regions in it whose likelihood of being a spill exceeds the parameter value. Learning is not required to be rapid or incremental: training data that becomes available in the future may be used to create a new training set so that the system can induce a new classifier.

3. Key Problem Characteristics

In developing the machine learning component of the system, the main design decisions were critically affected by certain key features of the oil spill detection problem. Many (though not all) of these features concern characteristics of the oil slick data. Brodley & Smyth (1995) refer to these aspects of a system's design as "application factors."

The first critical feature is the *scarcity* of data. Although satellites are continually producing images, most of these images contain no oil spills, and we did not have access to an automatic system for identifying those that do (the TSS data and systems reported by Solberg and collaborators were produced in parallel with our project and, in addition, are proprietary). A human expert therefore has to view each image, detect suspicious regions, and classify these regions as positive and negative examples. In addition to the genuine infrequency of oil spills and the limited time of the expert, the data available was restricted by financial considerations: images cost hundreds, sometimes thousands of dollars each. We currently have 9 carefully selected images containing a total of 41 oil slicks. While many applications work with large amounts of available data (Catlett, 1991), our domain application is certainly not unique in its data scarcity. For example, in the drug activity application reported by Dietterich et al. (1997) the two datasets contain 47 and 39 positive examples respectively.

The second critical feature of the oil spill domain can be called an *imbalanced training set:* there are very many more negative examples (lookalikes) than positive examples (oil slicks). Against the 41 positive examples we have 896 negative examples; the majority class thus comprises almost 96% of the data. Highly imbalanced training sets occur in applications where the classifier is to detect a rare but important event, such as fraudulent telephone calls (Fawcett & Provost, 1996), unreliable telecommunications customers (Ezawa et al., 1996), failures or delays in a manufacturing process (Riddle et al., 1994), or rare diagnoses (e.g., the thyroid diseases in the UCI repository (Murphy & Aha, 1994). Extremely imbalanced classes also arise in information retrieval and filtering tasks: in the domain studies by Lewis & Catlett (1994), only 0.2% (1 in 500) examples are positive. In the high-energy physics learning problem reported by Clearwater & Stern (1991), only 1 example in a million is positive.

The third critical feature is that examples are naturally grouped in *batches:* the examples drawn from the same image constitute a single batch. Whenever data is collected in batches, there is a possibility that the batches systematically differ from one another, or that there is a much greater similarity of examples within a batch than between batches. In our domain, for example, the exact parameter settings of the radar imaging system or low level image processing are necessarily the same for examples within a batch but could be different for different batches. Clearly, in our case, the classifier will be learned from one set of images, and it will be applied on images that were not part of this set. This fact should be taken into account in the evaluation of the system.

This problem has been mentioned by several other authors, including Burl et al. *cite this issue)*, Fawcett & Provost (1996), Ezawa et al. (1996), Kubat et al. (1994), and Pfurtscheller et al. (1992). For instance, in the SKICAT system (Fayyad et al., 1993), the "batches" were plates, from which image regions were selected. When the system trained on images from one plate was applied on a new plate, the classification accuracy dropped well below that of manual classification. The solution used in SKICAT was to normalize some of the original features.

The final critical feature relates to the *performance task.* The classifier will be used as a filter — it will decide which images to present to a human. This requirement is quite pervasive in real-world applications. Fraud detection, credit scoring, targeted marketing, evaluation of EEG signals — all of these domains require that a human expert be able to decide how many "suspicious" cases to pursue. The system must provide the user with a convenient means of varying "specificity" (higher specificity means fewer false alarms at the cost of increased risk of missing a genuine oil spill).

4. Problem Formulation Issues

Machine learning research usually assumes the existence of carefully prepared data that are then subjected only to marginal, if any, further processing: an attribute or two might be deleted, missing values filled in, some classes merged or dropped. In applications, the situation is not that straightforward (Langley & Simon, 1995). In our case we did not have a precise statement of the problem, much less a data file prepared in a standard format.

This section will briefly discuss issues related to problem formulation. Based on the initial vague description of the given problem, a successful designer of a learning system must make crucial decisions about the choice of the learning paradigm, about the representation and selection of the training examples, and about the categories into which the examples are going to be classified.

These choices must be made in all applications and they undoubtedly have a profound effect on the success, or appropriateness, of learning. Yet the exact nature of this effect is unknown and a systematic study of these aspects is needed.

1. The first decision concerned *granularity*. In our application three different approaches are possible. One of them works with the whole image and its output simply states whether the given image contains an oil slick. The second approach works with the dark regions detected in the images, and provides the user with coordinates of those regions that are considered as oil spills. Finally, the third approach classifies individual pixels ("this pixel is part of an oil slick"), as has been done, for instance by Ossen et al. (1994). The approach operating with pixels represents the finest granularity, whereas the approach operating with the images represents the coarsest granularity.

   Finer granularity provides more examples — compare the millions of pixels, with the 937 regions and 9 images. Moreover, in our application a higher misclassification rate can be tolerated at the pixel level: 80% accuracy on the pixels in oil slicks is very likely to identify more than 80% of the oil slicks. On the other hand, pixels can be described only with an impoverished set of features, and the result need not necessarily seem coherent to the user (e.g., if pixels are classified individually there is no guarantee that the "oil slick" pixels will form coherent regions in an image). We decided the system would classify regions.

   The need to choose the degree of granularity arises naturally in many applications. For instance in semiconductor manufacturing (Turney, 1995) circuits are manufactured in batches of wafers, and the system can be required to classify an entire batch, or each wafer, or to operate at even lower levels. Likewise, the text-to-speech mapping discussed by Dietterich et al. (1995) can be addressed in four distinct levels of granularity.

2. Another question was how many and which categories to define. Should all lookalikes be treated as a single category? Does it make sense to establish separate categories for different kinds of lookalikes? Discussions with the expert who classified the training examples have revealed that she might be able to place many lookalikes into subcategories like rain cells, wind, ship wakes, schools of herrings, red tide (plankton) and some others. Information about which categories are more prone to be misclassified could provide us with a clue for a better choice of training examples. Unfortunately, we overlooked this possibility during the initial data collection and so have been forced to have just one nonspill class. Solberg & Solberg (1996) divide their oil spills into subclasses based on shape. We decided not to do this because our dataset contained too few oil spills. We therefore have a two class learning problem.

Table 1. Confusion matrix

		guessed:	
		negative	positive
true:	negative	a	b
	positive	c	d

Once the initial decisions have been made, the designers must be clear about how to assess the merits of different variants of the learning system. To define performance criteria, researchers use a *confusion matrix,* such as the one in Table 1. Here, a is the number of True negatives (correctly classified negative examples), d is the number of True positives (correctly classified positive examples), c is the number of False negatives (positive examples incorrectly classified as negative), and b is the number of False positives (negative examples incorrectly classified as positive).

5. Performance Measure

The standard performance measure in machine learning is *accuracy*, calculated as $acc = \frac{a+d}{a+b+c+d}$. In other words, accuracy gives the percentage of correctly classified examples. This measure is inappropriate in applications where the classes are unequally represented in the training set. To see this, consider our case where the relative frequency of lookalikes is 96%. A classifier that labels all regions as lookalikes will achieve an accuracy of 96%. Even though this looks high, the classifier would be useless because it totally fails to achieve the fundamental goal of oil spill detection. By contrast, a system achieving 94% on spills and 94% on nonspills will have a worse accuracy and yet be deemed highly successful; very few spills would be missed and the number of false alarms would be small.

Informally, we want to (a) present to the user as many spills as possible, provided that (b) the total number of false alarms is not too large. Curves used to visualize the tradeoff between these two requirements are called ROC curves (Swets, 1988). In particular, the plot in Figure 2 shows how the percentage of correctly recognized positive examples ($\frac{d}{c+d}$) depends on the false positive rate ($\frac{b}{a+b}$). Many classifiers, including the one described below, make it possible to move along this curve, for instance by adjusting the bias of an output neuron in a multilayer perceptron. The number of correctly recognized positive examples can thus be increased at the cost of increased number of false alarms, or vice versa. Provost & Fawcett (1997) argue that ROC curves are good indicators of the classifier's performance in many reasonable applications. Swets (1988) proposes to measure the performance by the area under the ROC curve.

To measure performance in environments with imbalanced classes, the information retrieval community works with *recall* ($r = \frac{d}{c+d}$) and *precision* ($p = \frac{d}{b+d}$) and combines them by way of a geometric mean ($\sqrt{r \cdot p}$) or the more sophisticated F-measure (Lewis & Gale, 1994). Other measures have been suggested (van Rijs-

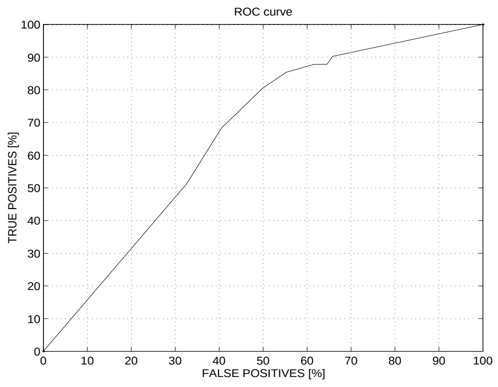


Figure 2. The ROC curve

bergen, 1979, Chapter 7), including an information theoretic formula suggested by Kononenko & Bratko (1991).

The standard decision theoretic approach to defining the "optimal" tradeoff between false and true positives is to assign relative costs, or utilities, to errors of omission and errors of commission, and to make the classification that minimizes expected cost (Pazzani et al., 1994). One deterrent to using this approach is that the costs are often hard to determine and may involve multiple considerations whose units are incommensurable (e.g., monetary cost, pollution levels, international reputation). Decision analysis techniques have been developed to cope with this difficulty (von Winterfeldt & Edwards, 1986; Keeney & Raiffa, 1993). But these techniques, which involve the eliciting of subjective judgements from the end user by a trained decision analyst, are awkward to use with a system such as ours which is not targeted at a particular end user. Moreover, MacDonald Dettwiler Associates required the system to be readily customizable to each user; this precludes the labour intensive knowledge elicitation techniques of decision analysis or knowledge based systems.

We decided that in the version of the system that will be delivered to end users there will not be a preprogrammed way of condensing the ROC curve to a single performance measure. Instead, the user will be able to move along the curve and choose the point that best meets his/her current needs. In this way, the user perceives the performance in terms of two parameters (the frequency of true positives and of false positives). This is typical of fielded systems: as pointed out by Saitta et al. (1995), systems that serve as tools for users confronting a specific decision (e.g., whether to send an aircraft to verify a spill and document the incident) should not be constrained to use a scalar performance measure. The user needs to be able to tune the system's behavior so as to trade off various conflicting needs.

Although, in general, the challenge is to build a system that can produce classifiers across a maximally broad range of the ROC curve, in the course of development we needed a performance measure to provide immediate feedback (in terms of a single value) on our design decisions. To this end, we have mainly used the geometric mean (*g-mean*), $g = \sqrt{acc{+} \times acc{-}}$, where $acc{+} = \frac{d}{c+d}$ is the accuracy on the positive examples, and $acc{-} = \frac{a}{a+b}$, is the accuracy on the negative examples. This measure has the distinctive property of being independent of the distribution of examples between classes, and is thus robust in circumstances where this distribution might change with time (or be different in the training and testing sets). Another very important and distinctive property is that *g-mean* is nonlinear. A change of p percentage points in $acc{+}$ (or $acc{-}$) has a different effect on *g-mean* depending on the magnitude of $acc{+}$: the smaller the value of $acc{+}$, the *greater* the change of g-mean. This property means that the "cost" of misclassifying each positive example increases the more often positive examples are misclassified. A learning system based on g-mean is thereby forced to produce hypotheses that correctly classify a non-neglible fraction of the positive training examples. On the other hand, g-mean is less than ideal for filtering tasks because it ignores precision.

We would like to emphasize that the use of ROC curves by a user who by moving on that curve can tune the system to the trade-off between the two types of errors that best meets their needs seems to be the best performance measure in this type of application. However, since during the development of the system on the one hand we did not have access to such users, and on the other hand we needed a straightforward way (preferably a single value) to monitor our progress, we have chose the g-mean as a compromise. Moreover, even though the ROC curves in themselves are useful, one needs a tool to compare them. The best currently available tool for this purpose is ROC convex hull proposed recently by Provost & Fawcett (1997).

6. Methodological Issues

After the problem formulation and the choice of the performance measure, the designer of a learning system must turn his or her attention to the specific idiosyncracies of the data available for learning. In the oil spill detection problem we faced the following issues.

1. Examples come in small batches, each with different characteristics.
2. The data are sparse and the training set is imbalanced.
3. There is no guarantee that the examples available for the system development are representative of the examples that will arise after deployment.
4. The need for feature engineering.
5. System development in a dynamically change environment.

Let us look at these issues in turn. The first issue is how to learn and experiment with *batched* examples. Table 2 gives some details about our data. The nine images

Table 2. The numbers of positive and negative examples in the images

image	1	2	3	4	5	6	7	8	9	all
positives	8	4	2	6	2	4	3	5	7	41
negatives	3	180	101	129	60	70	76	80	197	896
total	11	184	103	135	62	74	79	85	204	937

(batches) contain a total of 41 positive examples and 896 negative examples, and the characteristics of the individual images can vary significantly. Whereas the images all come from the same satellite, they represent various geographical locations and times. One can thus expect that the images contain oil spills of different origin and of different types.

One possible approach is to view the individual batches in the training set as coming from a different "context" and to use a context sensitive learning algorithm as suggested by Turney (1993), Widmer & Kubat (1996), and Kubat & Widmer (1996). However, our initial experiments with simple contextual normalization techniques were not entirely successful (partly because of the scarcity of the data), so we decided not to pursue context sensitive learning. Moreover, we do not always know what the contextual parameters are. Even when we know that in reality there is a contextual variable that influences the classifier (e.g. the wind speed), often we have no way to compute the value of this variable.

An alternative approach is to combine all the examples into one large dataset in the hope that the learning algorithm will be able to detect systematic differences between batches and react by creating a disjunctive definition with a disjunct, say, for each batch. However, if the individual batches have a relatively small number of positive examples such a system will be prone to the problem of small disjuncts (Holte et al., 1989). Moreover, the batches that have many examples (images 2 and 9) will dominate those that have few (image 1).

Batched examples also raise an issue about experimental methodology. Suppose that all examples are mixed in one data file from which a random subset is selected for training, leaving the rest for testing. This means that examples from the same batch can appear both in the training and testing sets. As a result, the observed performance will be overly optimistic compared to the deployed system's actual performance on completely unseen batches. This phenomenon will be experimentally demonstrated below. There really is no valid alternative but to separate the batches used for training from those for testing (Ezawa et al., 1996; Fawcett & Provost, 1996). The particular testing method we use is "leave-one-batch-out," which is the same as the traditional leave-one-out methodology except that one whole batch is left out on each iteration rather than just one example.

Overfitting is a term normally applied to a learning algorithm that constructs a hypothesis that "fits" the training data "too well." A different sense of "overfitting" has recently been introduced by Dietterich et al. (1997). Here the term is applied when an algorithm developer tunes a learning algorithm, or its parameter settings, to optimize its performance on all the available data. This type of overfitting, which we call *overtuning*, is a danger whenever all available data is used for algorithm development/tuning. Like ordinary overfitting, overtuning can be detected, if not avoided, by using only part of the data for algorithm development and using the remainder of the data for final system testing, as was done by Lubinsky (1994). If data is not held out for final system testing, the observed performance of the system cannot be confidently used as an estimate of the expected performance of the deployed system. Unfortunately, we had too little oil spill data to hold any out for final system testing in the development phase of the project. The system will be tested on fresh data in the field trials that are scheduled for winter 1998.

Dietterich et al. (1997) circumvented their lack of data by modelling some key characteristics of the data and generating a large artificial dataset, an idea proposed by Aha (1992). The point is to use the synthetic data for system development, and to return to the real data only for final system testing. We were hampered in using this method by the fact that our data came in small batches which were fairly dissimilar. We would have had to model both the within-batch characteristics and the across-batch characteristics, and we simply did not have enough batches to do this with any certainty. To try to ensure that our learning algorithm is not specific to our particular dataset we have tested it on other datasets having similar characteristics (Kubat et al., 1997).

Another issue is the *imbalance* of the dataset's class distribution. As shown by Kubat et al. (1997) and by Kubat & Matwin (1997), such well established learners as C4.5 (Quinlan, 1993) and the 1-nearest-neighbor rule (1-NN) can behave poorly if the training set is highly imbalanced: the induced classifiers tend to be highly accurate on negative examples but usually misclassify most of the positives. This will be demonstrated in the experimental part of this paper.

Two approaches promise to solve the problem. The first attempts to *balance* the classes. One way to do this is to discard those examples that are considered harmful. As early as the late sixties, Hart (1968) presented a mechanism that removes redundant examples and, somewhat later, Tomek (1976) introduced a simple method to detect borderline and noisy examples. In machine learning the best known sampling technique is windowing (Catlett, 1991). For more recent alternatives, see, for instance, Aha et al. (1991), Zhang (1992), Skalak (1994), Floyd & Warmuth (1995), and Lewis & Catlett (1994). Variations of data reduction techniques, namely those that remove only negative examples, are analyzed by Kubat & Matwin (1997). Conversely, the training set can be balanced by duplicating the training examples of the minority class or by creating new examples by corrupting existing ones with artificial noise (DeRouin et al., 1991). Solberg & Solberg (1996) do both: positives are duplicated and negatives are randomly sampled. Honda et al. (1997) reduce the imbalance by doing classification in two stages. In the first stage, the negatives most similar to the positives are included in the positive class. The second stage distinguishes these negatives from the true positives. This can be seen as a special case of multitask learning (Caruana, 1993), the more general idea being to define supplementary classification tasks in which the classes are more equally balanced. Pazzani et al. (1994) assign different weights to examples of different classes, Fawcett & Provost (1996) prune the possibly overfit ruleset learned from an imbalanced set, and Ezawa et al. (1996) force the learner to consider relationships between certain attributes above others.

The second approach is to develop an algorithm that is *intrinsically insensitive* to the class distribution in the training set. Extreme examples of this are algorithms that learn from positive examples only. A less extreme approach is to learn from positive and negative examples but to learn only rules that predict the positive class, as is done by BRUTE (Riddle et al., 1994). By measuring performance only of the positive predicting rules BRUTE is not influenced by the invariably high accuracy on the negative examples that are not covered by the positive predicting rules. Our SHRINK algorithm (Kubat et al., 1997) follows the same general principle – find the rule that best summarizes the positive examples – but uses a definition of "best" (g-mean) that takes into account performance of the negative predicting rules as well as the positive predicting ones. In section 7 we describe SHRINK and demonstrate empirically that its performance does not change as imbalance grows.

The third methodological issue is the *validity of the data selection.* We deliberately acquired only images containing oil slicks so as to maximize the number of positive examples in our dataset. However, this means that the distribution of examples in our dataset is different from the distribution that will arise naturally when the system is fielded. Fortunately, in our domain, all the lookalikes are natural phenomena whose presence in an image is independent of the presence of an oil slick. It is only because of this fact that we can have confidence that our performance on the acquired images will extend to "normal" images which mostly will not contain slicks.

Another methodological issue is *feature engineering.* We did not do any large scale constructive induction, as, for example was done by Cherkauer & Shavlik (1994); instead we relied on our domain experts to define useful features. The importance of good features was impressed upon them from the outset, and a significant fraction of their energy has been invested in this direction. While some of the features were generic (e.g., size and average brightness of the regions), others were motivated by theoretical considerations, and as such they implicitly represented domain knowledge. This approach has not been entirely successful: many of the features for which the experts had high expectations have not proven particularly helpful for classification. An open research issue is whether the domain knowledge used to define these features could have been used to better advantage if it had been captured explicitly and used to guide learning as suggested by Clark & Matwin (1993) or perhaps to guide constructive induction as investigated by Ioerger et al. (1995).

The discriminating power of our features is significantly influenced by the parameter settings of the low level image processing. Unfortunately, the settings that extract the greatest number of oil spills do not optimize the features' discriminating power, and we decided it was most important to maximize the number of oil spills. If we had had more images, we would have been able to improve our learning algorithm's performance by setting the image processing parameters to optimize the features. On the positive side, machine learning provided valuable feedback to the experts about the direct impact of a new feature on the performance measure and about the role played by the feature in the induced classifier. It was important that the feature's contribution to a decision be consistent with the experts' expectations (*cite: Lee et al. in this special issue*).

The final methodological issue relates to our working in a highly dynamic environment. The set of images and the set of features for each image changed throughout the project, as did the exact details of the low level image processing. Each of these changes produced a new dataset. The learning algorithms, too, were under constant development, and the experimental method changed several times before settling on leave-one-batch-out. The constant flux demanded careful bookkeeping about the exact versions of the datasets, algorithms, and methodology used in each experiment. This was done manually. A tool for this bookkeeping, for example an extension of the data preparation tool reported by Rieger (1995), would be a valuable contribution to applications of this kind.

7. Experimental Results

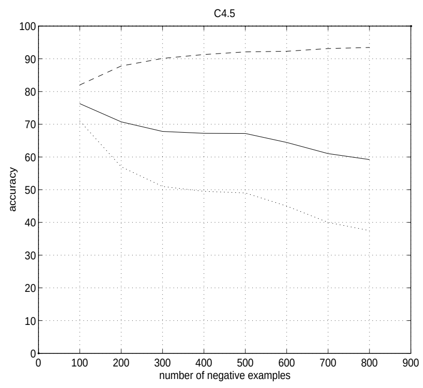
In this section we present experimental studies for two of the central research issues that arose in our application: (1) imbalanced training sets, and (2) batched examples.

Figure 3 illustrates the problem of imbalanced training sets. It shows the performance achieved by C4.5 and the 1-nearest-neighbor (1-NN) rule, for varying numbers of lookalikes while the set of oil spills remains unchanged. The curves represent average values obtained from 5 random runs of 10-fold cross-validation (for 5 different selections of negative examples). The figure shows that severe imbalance in the class distribution can have a detrimental effect on the quality of the resulting classifier: the g-mean and the accuracy on the positives both decrease considerably as the number of negative examples increase.

The behavior depicted in Figure 3 suggested a simple mechanism to alleviate the problem: induce the classifier using only a small subset of the existing negative examples. However, since not all negative examples have the same merit, Kubat & Matwin (1997) proposed a simple technique (*one-sided selection*) that removes from the training set redundant and noisy negative examples. The results achieved using this technique are summarized in Table 3. These results were obtained using 5 random runs of 10-fold cross-validation using all examples; in each run, the training set was reduced using the one-sided selection before the C4.5 or 1-NN rule were applied. C4.5 clearly benefited from this data reduction, both on the positive and on the negative examples (the improvement is statistically significant according to a t-test). However, in the case of the 1-NN rule, no significant improvement was observed compared to random selection of lookalikes.

One-sided selection is a method for altering an imbalanced training dataset so that accuracy-based systems will perform reasonably well. An alternative approach is

Figure 3. Performance of C4.5 (left) and the 1-nearest-neighbor rule (right) achieved on the testing set for different numbers of negative examples. Solid: g-mean; dashed: accuracy on negative examples; dotted: accuracy on positive examples.



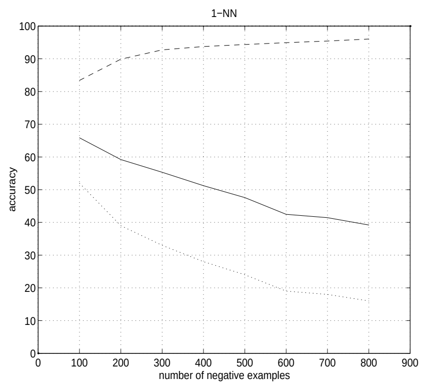


Table 3. Accuracies achieved with C4.5 and 1-NN after *one-sided sampling*

classifier	g-mean	acc+	acc-
C4.5	81.1	76.0	86.6
1-NN	67.2	54.0	83.4

to develop an algorithm that is insensitive to imbalance. With this aim in mind, we developed the SHRINK algorithm.

Three principles underlie SHRINK's design. First, if positive examples are rare, do not subdivide them when learning — dividing a small number of positive examples into two or three groups would make reliable generalization over each group impossible. The second principle is to induce a classifier of very low complexity. In particular, the classifier is represented by a *network of tests,* such as the one in Figure 4. The tests have the form $x_i \in [\min a_i; \max a_i]$ where i indexes the attributes. Denote by h_i the output of the i-th test, and let $h_i = 1$ if the test suggests a positive label, and $h_i = -1$ otherwise. The example is classified as positive if $\sum_i h_i \cdot w_i > \theta$, where w_i is the weight of the i-th test; the threshold θ gives the user the opportunity to relax or tighten the weight of evidence that is necessary for a region to be classified as an oil spill.

The third principle is to focus exclusively on the regions of instance space where positive examples occur. In the induction of the tests, SHRINK begins by establishing the "best" interval along each attribute, starting with the smallest interval containing all positive examples, and on every iteration shrinking the interval by

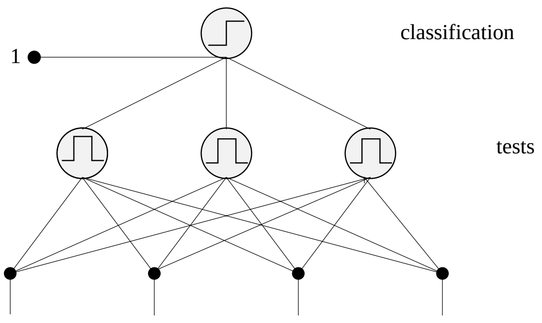


Figure 4. A classifier in the form of a network of tests.

eliminating either the left or right endpoint, whichever results in the better g-mean score. For each attribute, this produces a set of nested intervals from which the one with the maximum g-mean is selected as the test. Tests with g-mean $g < 0.5$ are then discarded. The *weight,* w_i, of the i-th test is calculated using the value of its g-mean, g: $w_i = \log(g_i/(1-g_i))$. This expression assigns higher weights to tests with small errors. The fact that the system uses only tests with $g_i > 0.5$ ensures that all weights are positive.

Figure 5 shows SHRINK's performance for $\theta = 0$, expressed in the same terms as Figure 3 (and using the same experimental methodology). It can be seen that SHRINK's performance is virtually unaffected by the number of negative examples. Comparing g-mean with that of the two conventional learners, we can see that SHRINK outperforms the 1-NN rule even for a small number of examples, perhaps because of the presence of many irrelevant attributes. On the other hand, C4.5 outperforms SHRINK if the negative examples are sampled. When presented with heavily imbalanced training sets, SHRINK scores better, but this advantage can be eliminated by the use of the one-sided sampling prior to the decision generation in C4.5.

The 1-NN rule has very high accuracy on negative examples and poor accuracy on the positive examples. C4.5 is similar, except when the number of negatives is most reduced by one-sided sampling. In that case its performance is like SHRINK's: accuracy on the positive is relatively good, while accuracy on the negatives is relatively poor. As mentioned earlier, it is important in our application to be able to explore the tradeoff between true and false positives dyanimically. In SHRINK, the user can move along the ROC curve by adjusting the threshold θ (the ROC curve is shown in Figure 6). The operator can thus reduce the frequency of false positives at the cost of an increased number of false negatives.

One of the methodological issues mentioned in the previous sections is the requirement that the classifier be trained on one set of images and tested on another set of images. Table 4 illustrates this point using some results obtained from exper-

Figure 5. SHRINK's performance. Solid: g-mean; dashed: accuracy on negatives; dotted: accuracy on positives.

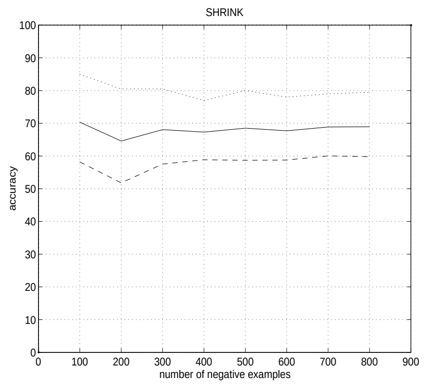


Figure 6. An example ROC curve (testing set) obtained from SHRINK on the oil spill data

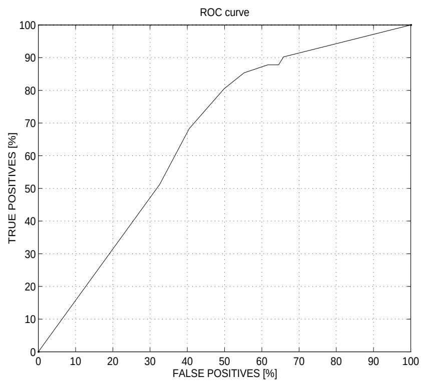


Table 4. Leave-one-batch-out (LOBO) compared conventional cross-validation (CV) with the same number of folds

	training set			testing set		
	g-mean	acc+	acc-	g-mean	acc+	acc-
CV	74.9	90.6	61.8	70.9	82.5	60.9
LOBO	75.0	85.7	65.6	62.5	78.1	50.1

imenting with SHRINK ($\theta = 0$). The first row (CV) contains the results obtained using the 10-fold cross-validation (average from 5 random runs) applied to the dataset containing all the examples from all images (so that examples from the same image can occur in both the training and the testing sets). These results are clearly superior to those in the second row, which are obtained using the leave-one-batch-out (LOBO) methodology. The experiment indicates that the images differ systematically, and therefore cannot be safely combined into one large dataset.

8. Comparison with Solberg & Solberg (1996)

As mentioned in the introduction Solberg & Solberg (1996) use machine learning to classify oil spills. Their classifier is represented as a decision tree and is learned from training data using S-plus (Venables & Ripley, 1994). To cope with the imbalanced classes Solberg & Solberg (1996) sample (with replacement) 100 examples from each class (four oil spill classes and a nonspill class). The accuracies reported by Solberg & Solberg (1996) — 86% accuracy on the oil spills, and 96% on the non-spills — are superior to the accuracies in our most similar experiment (C4.5 with the 100 negative examples and 41 positive examples — see Figure 3) and it is instructive to consider the possible causes of the difference.

The dataset of Solberg & Solberg (1996) is larger than ours, and based on more images (59, of which 29 contain slicks, compared to our 9), but has about the same number of oil spills (44 compared to our 41). However, their sampling technique causes nine copies of each oil spill to be included in the training set, on average, whereas we included just one. In their images there often occurred oil platforms or ships: these are bright objects in satellite radar images and a likely source of oil spills. Thus in their dataset a bright object being near a dark region is highly indicative of the region being an oil spill. Over half of their oil spills have a bright object nearby. Knowing this, they defined a feature, "distance to the nearest bright object" that considerably improved their results. If this feature is disabled, their accuracy on the oil spills drops to 73%, which is very similar to our C4.5 accuracy.

Finally, the experimental method used by Solberg & Solberg (1996) gives optimistic estimates for nonspills. On each run they hold out one oil spill for testing, then do the sampling described above to produce the training set. The accuracy of the resulting decision tree on the nonspills *in the training set* is recorded. Because their accuracy on the nonspills is based on data in the training set it is optimistic. In a personal communication, Anne Schistad Solberg has explained that these 1996 experiments were regarded as preliminary and that attention was mainly on the accuracy on the spills. In her subsequent work with E. Volden (1997), the leave-one-batch-out methodology is used.

9. Conclusions

The oil spill detection workstation has been delivered, under the name of CEHDS, to Macdonald Dettwiler Associates and will soon undergo field testing in several European countries (Spain, France, Portugal, and Italy). It has image processing suites for two satellites, RADARSAT and ERS-1. Two learning algorithms were included: one-sided selection and SHRINK. In the latter case, the user can control the rate of false alarms, and trade false alarms for missed oil spills. The user can also decide to retrain the system should more data become available in the future.

In developing the Oil Spill Detection Workstation, we faced numerous issues. Most are not specific to the oil spill detection problem: they arose because of properties of the application that arise frequently in machine learning applications. Although each application that has faced these issues has, of necessity, developed some solution, they have not yet been the subject of thorough scientific investigation. They are open research issues of great importance to the applications community.

Perhaps the most important issue is that of imbalanced classes. It arises very often in applications and considerably reduces the performance of standard techniques. Numerous methods for coping with imbalanced classes have been proposed, but they are scattered throughout the literature. At the very least, a large scale comparative study is needed to assess the relative merits of these methods and how they work in combination. Many individual methods, the SHRINK algorithm for example, can undoubtedly be improved by further research. It seems important to study small imbalanced training sets separately from large ones. In the latter, positive examples are numerous even though they are greatly outnumbered by negative examples. Some of the published methods for learning from imbalanced classes, require numerous examples of the minority class. The Bayesian approach described by (Ezawa et al. (1996), for example, works with several thousand examples in the minority class, while we were limited to several tens of examples of oil spills.

Learning from batched examples is another issue which requires further research. With the resources (manpower, data) available in this project, we were not able to devise a learning algorithm that could successfully take advantage of the grouping of the training examples into batches. However, we believe further research could yield such an algorithm. Learning from batched examples is related to the issues of learning in the presence of context, as the batches often represent the unknown context in which the training examples were collected. Learning in context has only recently been recognized as an important problem re-occurring in applications of machine learning (Kubat & Widmer, 1996).

Various tradeoffs arose in our project which certainly warrant scientific study. In formulating a problem, one must choose the granularity of the examples (images, regions, or pixels in our application) and the number of classes. Different choices usually lead to different results. For instance, having several classes instead of just two reduces the number of training examples per class but also provides additional information to the induction process. How can one determine the optimal choice ? Another tradeoff that arose was between the discriminating power of the features and the number of examples.

In machine learning applications there is no standard measure of performance. Classification accuracy may be useful in some applications, but it is certainly not ideal for all. The research challenge is to develop learning systems that can be easily adapted to different performance measures. For example, "cost sensitive" (or decision theoretic) learning algorithms work with a parameterized *family* of performance measures. Before running the learning algorithm the user selects a specific measure within this family by supplying values for the parameters (i.e., the costs). A second example is the "wrapper approach" to feature selection (Kohavi & John, to appear), parameter setting (Kohavi & John, 1995), or inductive bias selection (Provost & Buchanan, 1995). It can be adapted easily to work with any performance measure. Our approach was to have the learning system generate hypotheses across the full range of the ROC curve and permit the user to interactively select among them.

Feature engineering is a topic greatly in need of research. Practitioners always emphasize the importance of having good features, but there are few guidelines on how to acquire them. Constructive induction techniques can be applied when there is sufficient data that overtuning will not occur. The alternative to purely automatic techniques are elicitation techniques such as structured induction (Shapiro, 1987). More generally, one can elicit domain knowledge, as Solberg & Volden (1997) have done, and use a learning algorithm guided by a weak domain theory as done by Clark & Matwin (1993).

Our experience in this project highlights the fruitful interactions that are possible between machine learning applications and research. The application greatly benefited from — indeed would not have succeeded without — many ideas developed in the research community. Conversely, the application opened new, fertile research directions. Future research in these directions will directly benefit the next generation of applications.

Acknowledgments

The research was sponsored by PRECARN, Inc. Thanks are due to Mike Robson, the principal contributor to the image processing part, for his active cooperation with us and for providing invaluable comments also on the development of the machine learning subsystem. We also thank Peter Turney (National Research Council of Canada, Ottawa) for his advice on all aspects of this work, and Anne Schistad Solberg (Norwegian Computing Center, Oslo) for the information she provided about her work.

References

Aha, D., Kibler, D., & Albert, M. (1991). Instance-Based Learning Algorithms. *Machine Learning*, 6(1), 37–66.

Aha, D. (1992). Generalizing from Case Studies: A Case Study. In *Proceedings of the Ninth International Conference on Machine Learning* (pp. 1–10), Morgan Kaufmann.

Brodley, C., & Smyth, P. (1995). The Process of Applying Machine Learning Algorithms. *Working Notes for Applying Machine Learning in Practice: A Workshop at the Twelfth International Conference on Machine Learning*, Technical Report AIC-95-023 (pp. 7–13), NRL, Navy Center for Applied Research in AI, Washington, DC.

Burl, M.C., Asker, L., Smyth, P., Fayyad, U.M., Perona, P., Crumpler, L., & Aubele, J. (this issue). Learning to Recognize Volcanoes on Venus. *Machine Learning*, ??(??), ??–??.

Caruana, R. (1993). Multitask Learning: A Knowledge-based source of Inductive Bias. *Proceedings of the Tenth International Conference on Machine Learning* (pp. 41–48), Morgan Kaufmann.

Catlett, J. (1991). Megainduction: A Test Flight. *Proceedings of the Eighth International Workshop on Machine Learning*, pp. 596–599, Morgan Kaufmann.

Cherkauer, K.J., & Shavlik, J.W. (1994). Selecting Salient Features for Machine Learning from Large Candidate Pools through Parallel Decision-Tree Construction. In Kitano, H. & Hendler, J. (Eds.), *Massively Parallel Artificial Intelligence* (pp. 102-136), AAAI Press/MIT Press.

Clark, P. & Matwin, S. (1993). Using Qualitative Models to Guide Inductive Learning. *Proceedings of the Tenth International Conference on Machine Learning* (pp. 49–56), Morgan Kaufmann.

Clearwater, S., & Stern, E. (1991). A rule-learning program in high energy physics event classification. *Comp. Physics Comm.*, 67, 159–182.

DeRouin, E., Brown, J., Beck, H., Fausett, L, & Schneider, M. (1991). Neural Network Training on Unequally Represented Classes. In Dagli, C.H., Kumara, S.R.T. & Shin, Y.C. (Eds.), *Intelligent Engineering Systems Through Artificial Neural Networks*, pp. 135–145, ASME Press.

Dietterich, T.G., Hild, H., & Bakiri, G. (1995). A Comparison of ID3 and Backpropagation for English Text-to-Speech Mapping. *Machine Learning*, 18, 51–80.

Dietterich, T.G., Lathrop, R.H., & Lozano-Perez, T. (1997). Solving the Multiple-Instance Problem with Axis-Parallel Rectangles. *Artificial Intelligence*, 89(1–2), 31–71.

Ezawa, K.J., Singh, M., & Norton, S.W. (1996). Learning Goal Oriented Bayesian Networks for Telecommunications Management. *Proceedings of the Thirteenth International Conference on Machine Learning* (pp. 139–147), Morgan Kaufmann.

Fawcett, T., & Provost, F. (1996). Combining Data Mining and Machine Learning for Effective User Profiling. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (pp. 8–13), AAAI Press.

Fayyad, U.M., Weir, N., & Djorgovski, S. (1993). SKICAT: A Machine Learning System for Automated Cataloging of Large Scale Sky Surveys. *Proceedings of the Tenth International Conference on Machine Learning* (pp. 112–119), Morgan Kaufmann.

Floyd, S., & Warmuth, M. (1995). Sample Compression, Learnability, and the Vapnik-Chervonenkis Dimension. *Machine Learning*, 21, 269–304.

Hart, P.E. (1968). The Condensed Nearest Neighbor Rule. *IEEE Transactions on Information Theory*, IT-14, 515–516.

Haverkamp, D., Tsatsoulis, C., & Gogineni, S. (1994). The Combination of Algorithmic and Heuristic Methods for the Classification of Sea Ice Imagery. *Remote Sensing Reviews*, 9, 135–159.

Heerman, P. D., & Khazenie, N. (1992). Classification of Multispectral Remote Sensing Data using a back-propagation Neural Network. *IEEE Trans. of Geoscience and Remote Sensing*, 30, 81–88.

Holte, R. C., Acker, L., & Porter, B. W. (1989). Concept Learning and the Problem of Small Disjuncts. *Proceedings of the International Joint Conference on Artificial Intelligence* (pp. 813–818), Morgan Kaufmann.

Honda, T., Motizuki, H., Ho, T.B., & Okumura, M. (1997). Generating Decision Trees from an Unbalanced Data Set. In van Someren, M., & Widmer, G. (Eds.), *Poster papers presented at the 9th European Conference on Machine Learning* (pp. 68-77).

Hovland, H. A., Johannessen, J. A., & Digranes, G. (1994). Slick Detection in SAT Images. *Proceedings of IGARSS'94* (pp. 2038–2040).

Ioerger, T.R., Rendell, L.A., & Subramaniam, S. (1995). Searching for Representations to Improve Protein Sequence Fold-Class Prediction. *Machine Learning*, 21, 151–176.

Keeney, R.L., & Raiffa, H. (1993). *Decisions with Multiple Objectives: Preferences and Value Tradeoffs,*, Cambridge University Press.

Kohavi, R., & John, G.H. (to appear). Wrappers for Feature Subset Selection. *Artificial Intelligence* (special issue on relevance).

Kohavi, R., & John, G.H. (1995). Automatic Parameter Selection by Minimizing Estimated Error. *Proceedings of the Twelfth International Conference on Machine Learning* (pp. 304-312), Morgan Kaufmann.

Kononenko, I., & Bratko, I. (1991). Information-Based Evaluation Criterion for Classifier's Performance. *Machine Learning*, 6, 67–80.

Kubat, M., Holte, R., & Matwin, S. (1997). Learning when Negative Examples Abound. *Machine Learning: ECML-97, Lecture Notes in Artificial Intelligence 1224*, pp. 146–153, Springer.

Kubat, M., and Matwin, S. (1997). Addressing the Curse of Imbalanced Training Sets: One-Sided Sampling. *Proceedings of the Fourteenthth International Conference on Machine Learning* (pp. 179–186), Morgan Kaufmann.

Kubat, M., Pfurtscheller, G., & Flotzinger D. (1994). AI-Based Approach to Automatic Sleep Classification. *Biological Cybernetics*, 79, 443–448.

Kubat, M., & Widmer, G. (Eds.) (1996). *Proceedings of the ICML'96 Pre-Conference Workshop on Learning in Context-Sensitive Domains.*

Langley, P., & Simon, H.A. (1995). Applications of Machine learning and Rule Induction. *Communications of the ACM*, 38(11), 55–64.

Lee et al. (this issue). *Machine Learning*, ??(??), ??–??.

Lewis, D., & Catlett, J. (1994). Heterogeneous Uncertainty Sampling for Supervised Learning. *Proceedings of the Eleventh International Conference on Machine Learning* (pp. 148–156), Morgan Kaufmann.

Lewis, D., & Gale, W. (1994). A Sequential Algorithm for Training Text Classifiers. *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 3–12), Springer-Verlag.

Lubinsky, David (1994). *Bivariate Splits and Consistent Split Criteria in Dichotomous Classification Trees.* Ph.D. thesis, Computer Science, Rutgers University.

Murphy, P., & Aha, D. (1994). UCI Repository of Machine Learning Databases (machine-readable data repository). University of California, Irvine.

Ossen, A., Zamzow, T, Oswald, H., & E. Fleck (1994). Segmentation of Medical Images Using Neural-Network Classifiers. *Proceedings of the International Conference on Neural Networks and Expert Systems in Medicine and Healthcare (NNESMED'94)*, pp. 427–432.

Pazzani, M., Merz, C., Murphy, P., Ali, K., Hume, T., & Brunk, C. (1994). Reducing Misclassification Costs. *Proceedings of the Eleventh International Conference on Machine Learning* (pp. 217–225), Morgan Kaufmann.

Pfurtscheller, G., Flotzinger, D., & Kalcher, J. (1992). Brain-Computer Interface - A New Communication Device for Handicapped Persons. In Zagler, W. (Ed.), *Computer for Handicapped Persons: Proceedings of the Third International Conference* (pp. 409–415).

Provost, F.J., & Buchanan, B.G. (1995). Inductive Policy: The Pragmatics of Bias Selection. *Machine Learning*, 20(1/2), 35–62.

Provost, F.J., & Fawcett, T. (1997). Analysis and Visualization of Classifier Performance: Comparison under Imprecise Class and Cost Distributions. *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining* (pp. 43–48).

Quinlan, J.R. (1993). *C4.5: Programs for Machine Learning.* Morgan Kaufmann.

Riddle, P., Segal, R., & Etzioni, O. (1994). Representation design and brute-force induction in a Boeing manufacturing domain. *Applied Artificial Intelligence*, 8, 125–147.

Rieger, Anke (1995). Data Preparation for Inductive Learning in Robotics. *Proceedings of the IJCAI-95 workshop on Data Engineering for Inductive Learning* (pp. 70–78).

Saitta, L., Giordana, A., & Neri, F. (1995). What Is the “Real World”?. *Working Notes for Applying Machine Learning in Practice: A Workshop at the Twelfth International Conference on Machine Learning*, Technical Report AIC-95-023, pp. 34–40, NRL, Navy Center for Applied Research in AI, Washington, DC.

Shapiro, A.D. (1987). *Structured Induction in Expert Systems.* Addison-Wesley.

Skalak, D. (1994). Prototype and Feature Selection by Sampling and Random Mutation Hill Climbing Algorithms. *Proceedings of the Eleventh International Conference on Machine Learning* (pp. 293–301), Morgan Kaufmann.

Solberg, A.H.S., & Solberg, R. (1996). A Large-Scale Evaluation of Features for Automatic Detection of Oil Spills in ERS SAR Images. *IEEE Symp. Geosc. Rem. Sens (IGARSS)* (pp. 1484–1486).

Solberg, A.H.S., & Volden, E. (1997). Incorporation of Prior Knowledge in Automatic Classification of Oil Spills in ERS SAR Images. *IEEE Symp. Geosc. Rem. Sens (IGARSS)* (pp. 157–159).

Swets, J.A. (1988). Measuring the Accuracy of Diagnostic Systems. *Science*, 240, 1285–1293.

Tomek, I. (1976). Two Modifications of CNN. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-6, 769–772.

Turney, P. (1995). Data Engineering for the Analysis of Semiconductor Manufacturing Data. *Proceedings of the IJCAI-95 workshop on Data Engineering for Inductive Learning*, pp. 50–59.

Turney, P. (1993). Exploiting Context when Learning to Classify. *Proceedings of the European Conference on Machine Learning.* pp. 402–407. Springer-Verlag.

van Rijsbergen, C.J. (1979). *Information Retrieval (second edition)*, Butterworths. http://www.dcs.gla.ac.uk/Keith/Preface.html

Venables, W.N., & Ripley, B.D. (1994). *Modern Applied Statistics with S-Plus.* Springer-Verlag.

von Winterfeldt, D., & Edwards, W. (1986). *Decision Analysis and Behavioral Research.* Cambridge University Press.

Widmer, G., & Kubat, M. (1996). Learning in the Presence of Concept Drift and Hidden Contexts. *Machine Learning*, 23, 69–101.

Zhang, J. (1992). Selecting Typical Instances in Instance-Based Learning. *Proceedings of the Ninth International Machine Learning Workshop* (pp. 470–479), Morgan Kaufmann.

Received Date

Accepted Date
Final Manuscript Date