Using Boosting to Improve Oil Spill Detection in SAR Images

Geraldo L. B. Ramalho, Fátima N. S. Medeiros
Universidade Federal do Ceará
Image Processing Research Group
Fortaleza, CE, Brazil
glbramalho@gmail.com; fsombra@deti.ufc.br

Abstract

Marine surveillance system which uses Synthetic Aperture Radar (SAR) images to oil spill detection must minimize false alarms in order to improve its reliability. This paper presents an application that uses boosting method to minimize misclassification and yields better generalization. Different feature sets were applied to neural network classifiers and its performance compared do boosting methods. The experiments reached substantial improvement in the classification accuracy to discriminate oil spots from the look-alike ones.

1. Introduction

Mapping ocean oil pollution by using SAR (Synthetic Aperture Radar) images is an important area of interest for most applications. Concerning ocean oil spill surveillance, one can make use of the SAR images to extract feature sets by using different methods to predict if a specific region contains an oil spill or not.

As radar backscatter values for oil slicks are very similar to backscatter values for very calm sea areas and other ocean phenomena, dark areas in SAR imagery tend to be misinterpreted. Also, feature extraction is penalized due to speckle noise present in SAR images. For operational detection systems, it is important to reduce at the minimum the number of false alarms.

The improvement on the detection performance is achieved by increasing a set of features coded by different systems taken in order to provide more nuances. Nevertheless, the classifier design becomes a hard task. The use of neural network (NN) classifiers based on MLP (Multilayer Perceptron) and RBF (Radial Basis Function) produce good solutions [17] for this approach. However, the neural network parameter estimation does not always yield a global minimum solution. So, the use of a single best classifier may not be the best solution to design an automatic system of oil spill detection [5].

In recent works, researchers in computational learning theory have started to consider algorithms that search for a good classification rule by optimizing quantities other than the training error. Algorithms of this type include boosting [14] which has been designed to maximize the margin of a linear classifier. Classifier combination has shown to improve a classifier performance and also minimize the misclassification [5, 11]. Boosting is one of the most common methods found in recent works. The use of boosting method is growing in the classification task because it is a very effective computational procedure to improve the performance of a classifier. Moreover, it is resistant to overfitting [8] which is a good advantage over other methods.

AdaBoost [7] is an adaptive boosting method which is fast, simple and easy to program. The simplicity becomes from the fact that it has only one parameter to tune. LogitBoost [9, 15] is another boosting method that has the property to minimize the classifier bias because its logistic regression model [12]. These methods were tested with success to improve multivariate classification applications like the protein domain structural class prediction based on 3D structures [2, 6], automatic feature combination [19], tumour classification [3] and for automatic target recognition from SAR images [16]. An AdaBoost algorithm was recently modified by Yin et al. [19] to provide high level feature combination. This algorithm is useful to work on automatic feature selection and, thus, increase the classification performance over high dimensional datasets. A method to decompose multiclass problems into a set of binary ones through the error-correction output codes (ECOC) is explained in Sun et al. [16]. This is very suitable for using simple binary boosting algorithms to solve multiclass classification problems.

In this paper we present a methodology that combines classifiers using the AdaBoost and LogitBoost algorithms to improve oil spill identification in SAR images. In the next section we briefly discuss the boosting methods and
its characteristics. In Section 3 we present the results of the proposed approach applied to oil spill detection over 20 SAR images samples each with shape and texture extracted features. The boosting performance is compared to classical classifiers such as neural networks. In the last section we present the concluding remarks.

2. Boosting

Boosting [14] is an ensemble method introduced in 90’s, and used for arbitrarily increase the precision of a “weak” classifier (also called “base classifier” or “base function”) based on the principle of divide-and-conquer [11]. An ensemble is a set of classifiers whose individual predictions are combined to classify new examples [5] with better performance than using a single classifier.

On each boosting round, one “weak” classifier is constituted and trained using a different sample distribution which depends on the misclassified vectors. These classifiers are then combined by weighted voting into a new final classifier. The original boosting algorithm was improved to get AdaBoost (Adaptive Boosting) [7]. AdaBoost is sometimes compared to SVM (support vector machines) because the “weak” classifiers are forced to focus on the patterns nearest to the decision boundaries [11, 18]. Freund and Shapire [8] discuss about how SVMs use the $l_\infty$ norm while AdaBoost uses the $l_\infty$ norm for margin minimizing.

The AdaBoost [8] algorithm approximates a function $F(x)$:

$$ F(x) = \sum_{m=1}^{M} c_m f_m(x) $$

where $c_m$ are constants to be determined and $f_m$ are the basis functions or “weak” classifiers.

A Discrete Binary AdaBoost algorithm is shown in Figure 1. Given $S = \{(\vec{x}_1, y_1), \ldots, (\vec{x}_N, y_N)\}$, where $\vec{x}_i \in X$, $y_i \in Y = \{-1, 1\}$, and the number of iterations $M$, $f_m$ classifiers are trained to form the ensemble. This algorithm is very simple and it is suitable for most applications to producing accurate classifiers. However, a generic version of this algorithm is shown in Friedman et al. [9]. The Real Binary AdaBoost algorithm minimizes the criterion:

$$ J(F) = E\left(e^{-yF(x)}\right) \tag{2} $$

where $y$ is the true class label in $\{-1, 1\}$. Few changes are necessary in step 2 of the Discrete AdaBoost.

The bias-variance dilemma is always present in all classifiers design [4]. The overall approaches try to minimize the bias and variance. It is noteworthy that the variance has close relationship with the generalization performance of the classifier.

<table>
<thead>
<tr>
<th>The Pseudo-Code of Discrete Binary AdaBoost</th>
</tr>
</thead>
</table>
| 1. Start with weights $w_i = 1/N$, $i = 1, 2, \ldots, N$.
| 2. Repeat $m=1, 2, \ldots, M$:
|   (a) Fit the classifier $f_m(x) \in [-1, 1]$ using weights $w_i$ on the training data.
|   (b) Compute $err_m = E_w[1(y \neq f_m(x))]$, $c_m = \log((1 - err_m)/err_m)$.
|   (c) Set $w_i \leftarrow w_i \exp(c_m 1(y_i \neq f_m(x_i)))$; $i = 1, 2, \ldots, N$, and normalize so that $\sum_i w_i = 1$.
| 3. Output the classifier $sign\left[\sum_{m=1}^{M} c_m f_m(x)\right]$.

| Figure 1. The Pseudo-Code of Discrete Binary AdaBoost [7]. |

<table>
<thead>
<tr>
<th>The Pseudo-Code of Binary LogitBoost</th>
</tr>
</thead>
</table>
| 1. Start with $w_i = 1/N$, $i = 1, 2, \ldots, N$; initialize committee function $F(x) = 0$ and probabilities estimates $p(x) = P(y = 1|x) = 1/2$.
| 2. Repeat $m=1, 2, \ldots, M$:
|   (a) Compute the weights and working response $w_i = p(x_i) [1 - p(x_i)]; z_i = \frac{y_i^* - p(x_i)}{w_i}$, where $y_i^* = (y_i + 1)/2$.
|   (b) Fit the function $f_m(x)$ by a weighted least-squares regression of $z_i$ to $x_i$ using weights $w_i$. In our study we use a MLP to fit the data $\{(x_1, z_1), \ldots, (x_N, z_N)\}$ using weights $w_i$.
|   (c) Update $F(x) \leftarrow F(x) + \frac{1}{2} f_m(x)$ and $p(x) \leftarrow \frac{e^{F(x)}}{e^{F(x)} + e^{-F(x)}}$.
| 3. Output the classifier $sign\left[F(x)\right] = sign\left[\sum_{m=1}^{M} f_m(x)\right]$.

| Figure 2. The Pseudo-Code of Binary LogitBoost [9] |
The LogitBoost algorithm proposed in Friedman et al. [9] is a boosting improvement in order to reduce training errors linearly which provides bias minimizing [12] and hence improves the generalization. This algorithm is shown in Figure 2 and it is based on the observation that AdaBoost is in essence fitting an additive logistic regression model to the training data [9, 12]. It minimizes the criterion by using Newton-like steps to fit an additive logistic regression model to directly optimize the binomial log-likelihood:

\[ J(F) = E \left( -\log \left( 1 + e^{-2yF(x)} \right) \right) \]  

(3)

An important advantage of boosting compared to other methods like NNs or SVMs is that it works without fine tuning and no sophisticated nonlinear optimization is necessary [3]. Both boosting algorithms have only one parameter to tune: the number \( M \) of “weak” classifiers. As boosting is generally resistant to overfitting, the choice of \( M \) is typically not very critical. Moreover, there is an empirical approach to automatically choose this parameter explained in [1, 3]. This characteristic makes boosting algorithms very useful for use in practical applications.

3. Experimental Results and Discussions

3.1. Implementation

We have implemented the boosting algorithms using a three-layer MLP as the base learner. This MLP was designed with 10 neurons in the hidden layer. We have fixed a maximum number of 10 base learners for each boosting algorithm. The binary version of both AdaBoost and LogitBoost algorithm was chosen because its simplicity. In addition, these boosting algorithms are suitable for our purposes because the nature of our binary classification problem.

In the step two of the boosting methods (see Figures 1 and 2), the MLP training is based on weighted input vectors which is closely related to its probability of occurrence. An appropriate method to implement this approach is by using resample. In our experiments, an input vector was resampled as many times as needed to achieve its weight in training.

The working dataset was taken from 20 SAR images separated into two classes of 10 samples with oil spill and look-alike images. Two examples of SAR images used in the experiments are shown in Figure 3. Figure 3a concerns a real oil spill image in the ocean and Figure 3b exhibits a similar one. Three feature sets were formed to test a detection system using each of the classifiers in Table 1. The first two sets are shape features \( (S1) \) generated as described in Nirchino et al. [13] and texture features \( (S2) \) as suggested in Haralick et al. [10]. The third one is formed by the union \( (S3 = S1 \cup S2) \) of the both cited. The sets \( S1, S2 \) and \( S3 \) are respectively 8, 15 and 23-dimensional.

3.2. Results

The datasets were submitted to 5 different classifiers: one classical kNN (k-Nearest Neighbours), two neural networks (RBF and MLP) and two boosting algorithms (AdaBoost and LogitBoost). In Table 1 we report the estimated success rate generated from 1000 round tests of each classifier using a hold-out method with 70% of the samples used for training. Boosting methods have shown better performance over all datasets. For real applications, as the one proposed in this paper, the use of a single NN, brings up the need to select one of the 1000 trained NNs. Indeed, most applications makes use of the NN with the best performance or that one with the least training error. However, the choice of the best one is not a guarantee to generalize well. Instead, if we use the classifier ensemble trained using one of the proposed boosting methods, the performance is improved without loss of generalization ability.

The classifier variance is an important parameter to help understand the generalization capacity of the classifier [4]. In Table 2 the estimated classifier variances are reported. The boosting classifier variances are smaller than the others, which means that the generalization of this approach is also better.
4. Concluding Remarks

In this paper we propose an approach based on boosting to minimize misclassification in oil spill detection systems. The use of boosting as a classifier significantly improves the performance of oil spill identification even if the dataset has high dimensionality. This makes both proposed boosting algorithms suitable for application in marine surveillance systems using SAR images, which requires good generalization and lower false-positive rates. Although boosting methods have high computational cost for training, the ensemble response has shown to be as fast as the response of a single neural network. Moreover, the boosting variances are significantly smaller than the other classifiers used for comparison. LogitBoost has shown a slightly better variance when compared with AdaBoost. This occurs because its lower sensibility to noise. The results, with this set of test images, obtained by using boosting are promising, and the classification error has shown to be smaller than using single neural network classifiers. Also, the use of boosting methods facilitates the classifier design. Indeed, only one parameter must be tuned and there is no need to choose a single trained base learner, as the ensemble automatically takes care on it. In future works we will investigate the use of AdaBoost and LogitBoost algorithms and its variants [19] as automatic feature selection in multi-class and high dimensional classification problems.

5 Acknowledgement

The authors would like to thank CNPq (#476177/2004-9) and CAPES for their financial support.

References