Machine Learning on Historic Air Photographs for Mapping Risk of Unexploded Bombs

Stefano Merler, Cesare Furlanello, and Giuseppe Jurman

ITC-irst - Trento, Italy
{merler,furlan,jurman}@itc.it

Abstract. We describe an automatic procedure for building risk maps of unexploded ordnances (UXO) based on historic air photographs. The system is based on a cost-sensitive version of AdaBoost regularized by hard point shaving techniques, and integrated by spatial smoothing. The result is a map of the spatial density of craters, an indicator of UXO risk.

1 Introduction

More than 1 million bombs were air-dropped by the Allied Forces during WWII in Italy, and at least 10% did not explode. At least 25,000 unexploded ordnances are thus likely to remain buried, in the optimistic estimate that 3 out of 4 were found and correctly disposed of. Only in Trentino, we have knowledge of 32,019 high explosive bombs (801 of which with Long Delay fuses) aimed at 271 attack targets, accounting for 800 – 1,280 UXO still to be found. After more than 50 years, the ordnances are still operative: unexploded bombs constitute a risk for population in case of accidental discovery, and a high inconvenience for any intervention (e.g., construction of infrastructures, remediation of areas), often delaying works, obliging also to evacuations of thousands inhabitants and to communication blockades.

During the war, over 30 millions aerial photographs were taken for target identification, damage assessment, mapping, and other purposes and still survive in the archives of Keele (UK) and NARA (US). On the basis of available mission data, reconnaissance imagery may be acquired from archives, digitalized and geocoded. Images are selected in order to cover all locations in the study area, and at different dates in the case of repeated attacks to targets.

In this paper we present a learning methodology for the detection of the areas potentially including unexploded ordnances. The classification problem is in many aspects similar to that of identifying volcanoes on Venus [1]. Our task is characterized by two main characteristics: the requirement of high specificity (to reduce the number of false alarms) and the high number of test points (making unfeasible the use of several computationally intensive algorithms). Moreover, it is worthwhile noting that there is no need to precisely identify all the single craters: the detection of their clusters is the more informative target. The proposed solution, SSTBoostReg, extends the classical AdaBoost [2] algorithm and it accounts for the above described issues by means of a regularized cost-sensitive variant coupled to a spatial smoother.

F. Roli and S. Vitulano (Eds.): ICIAP 2005, LNCS 3617, pp. 735–742, 2005.

2 Data Description and Preprocessing

A set of $1,464$ images, taken at different dates throughout the whole WWII period, and rather heterogeneous, in terms of both general quality and ground resolution, was selected for the described detection task. The general quality of the images is affected by sky conditions (presence of clouds or smoke), exposure hour (light conditions, brightness and contrast), exposure season (snow may cover the craters, foliages may change) and preservation status. The pixel ground resolution - approximately between 0.25×0.25 and 1×1 m^2 - depends on the flight altitude which ranges from $9,000$ to $25,000$ feet because of the enemy artillery opposition.

The images were scanned at 600dpi resolution and geolocated within the Geographic Information System GRASS (`http://grass.itc.it`). The ground resolution of all the images was reduced to 1.5×1.5 m^2 for noise reduction and uniformity purposes. Fifty images chosen from different reconnaissance flights and of different quality were retained for training set extraction, learning and evaluation of classification models, whereas all the remaining photographs were used for building the risk map.

The training set consists of a set $\{(\mathbf{v}_i, y_i)\}_{i=1,\ldots,N}$ of $k \times k$ pixel windows around examples of the patterns one is trying to recognize. Each pixel window $\mathbf{v}_i$ is seen as a vector of $\mathbb{R}^{k^2}$ whose label y_i belongs to $\{-1, 1\}$. Taking into account the pattern variability, we selected $1,639$ positive examples (a total of about $30,000$ craters is expected to be found in all the study area) and $2,367$ negative ones from 30 out of 50 training images (the remaining 20 images being only used for model evaluation), for a total of $N = 4,006$ examples. The window size was set to 19.5×19.5 m^2, corresponding to $k = 13$ pixels. We tried to include all the possible observable patterns into the training set. Examples from the two classes are shown in Fig. 1a and 1b. In particular, it should be observed that the examples of trees appear very similar to those of craters.

It is worthwhile mentioning that the extraction of training data was not trivial, both for the positive and the negative examples, because of the heterogeneity of the involved patterns. All the training examples were normalized with respect to the local brightness to make it independent from the light conditions of the particular reconnaissance flight according to the transformation $\tilde{\mathbf{v}}_i = (\mathbf{v}_i - \mu_i)/\sigma_i$ for $i = 1, \ldots, N$, where μ_i and σ_i represent the mean value and the standard deviation of the training example $\mathbf{v}_i$ respectively.

For feature reduction, a principal component analysis was performed on the positive samples of training set. A graphical representation of the Eigencraters (i.e., the eigenvectors corresponding to the positive samples) in order of non increasing eigenvalues is shown in Fig. 1c. We took into account only the first $n = 11$ principal components, accounting for more than 95% of the total variance, for the development of the classification models. The projection of the original data into the subspace spanned by the first n Eigencraters is obtained as $\mathbf{x}_i = \tilde{\mathbf{v}}_i[\mathbf{e}_1, \ldots, \mathbf{e}_n]$, where $[\mathbf{e}_1, \ldots, \mathbf{e}_n]$ is the matrix of the Eigencraters in order of non increasing eigenvalues. Therefore, the training set becomes now the ensemble $\{(\mathbf{x}_i, y_i)\}_{i=1,\ldots,N}$.

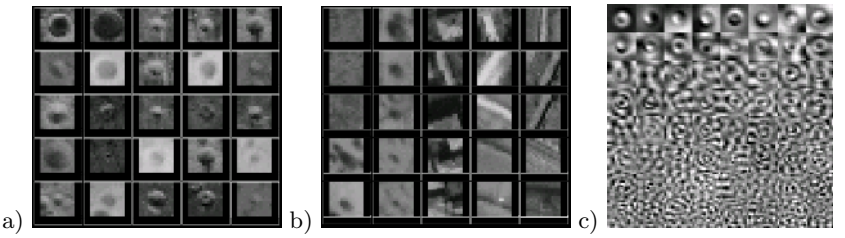

Fig. 1. Examples of positive (left panel) and negative (middle panel) samples (including examples of trees, one of the most confounding patterns, buildings and roads) and Eigencraters reported in order of non increasing eigenvalue (right panel)

3 The SSTBoostReg Model

Standard classification techniques like Nearest Neighbor (NN), Maximum Likelihood Gaussian Classifier (MLGC) and Tree Based Classifiers (TBC) [3] were firstly applied. Bagging [4] and AdaBoost of maximal trees were also considered (100 aggregated models). Performances of these classifiers are reported in Table 1 and they are estimated by 10-fold crossvalidation on the $N = 4,006$ training examples extracted from the 30 out of the 50 training images. AdaBoost achieved the best results in terms of global accuracy (0.92) and performed slightly better than Bagging. However, simpler single classifiers like NN and MLGC achieved higher specificity (0.96) and higher sensitivity (0.89), respectively. Nevertheless, both these last classifiers suffer from not trivial drawbacks. NN classifiers are too slow during the test phase while the computation time is a key factor in our application ($1,464$ test images, corresponding to about 6.10^9 test points); for instance, Bagging and AdaBoost, the fastest models, require an execution time of about 350 hours on a Pentium IV, 3GHz. MLGC is characterized by too low specificity (0.86) and it is very likely to cause many false alarms.

Table 1. Accuracy, sensitivity and specificity (by 10-fold crossvalidation), with the standard deviation, are shown for different standard models and combining methods

Classifier	$Acc. \pm SD$	$Sens. \pm SD$	$Spec. \pm SD$
1NN	0.88 ± 0.02	0.83 ± 0.03	0.93 ± 0.02
3NN	0.90 ± 0.01	0.84 ± 0.02	0.95 ± 0.01
5NN	0.91 ± 0.01	0.84 ± 0.02	0.96 ± 0.01
7NN	0.91 ± 0.01	0.84 ± 0.02	0.96 ± 0.01
MLGC	0.87 ± 0.03	0.89 ± 0.01	0.86 ± 0.02
TBC	0.83 ± 0.02	0.79 ± 0.03	0.85 ± 0.03
Bagging	0.90 ± 0.01	0.85 ± 0.02	0.94 ± 0.01
AdaBoost	0.92 ± 0.01	0.88 ± 0.02	0.95 ± 0.01

Despite the relatively high specificity of the model, the application of AdaBoost to the test images was not satisfactory because of the plethora of false alarms inserted in the test images. On the other side, the percentage of correct recognition of craters resulted to be highly satisfactory. In summary, a specificity of 0.95 is not sufficient to avoid a too high number of false alarms, meanwhile a sensitivity of about 0.75 should be sufficient to identify all the clusters of craters. To achieve higher specificity (at least 0.975), we developed a cost-sensitive variant of AdaBoost, further refined by introducing a regularization factor. A spatial smoothing bivariate function was finally applied.

3.1 The SSTBoost

Cost-sensitive variants of the AdaBoost have been already developed, but they require the introduction of a misclassification cost into the learning procedure [5,6]. Unfortunately, the misclassification costs concerning our task were not available. However, the minimal requirements in terms of accuracy were given as discussed at the beginning of Sec. 3: we could consider satisfactory a model characterized by a very high specificity, at least 0.975, and a sensitivity of at least 0.75. Moreover, there is no need to know the class priors (unknown in our case): the only point is to drive the system into the sensitivity – specificity constraints. To this purpose we have developed SSTBoost (Sensitivity Specificity Tuning Boosting) [7], a variant of AdaBoost where (1) the model error is weighted with separate costs for errors (false negative and false positives) in the two classes, and (2) the weights are updated differently for negatives and positives at each boosting step. Finally, (3) SSTBoost includes a practical search procedure which allows reaching the sensitivity – specificity constraints by automatically selecting the optimal costs. Given a parameter $w \in [0, 2]$, the SSTBoost algorithm allows

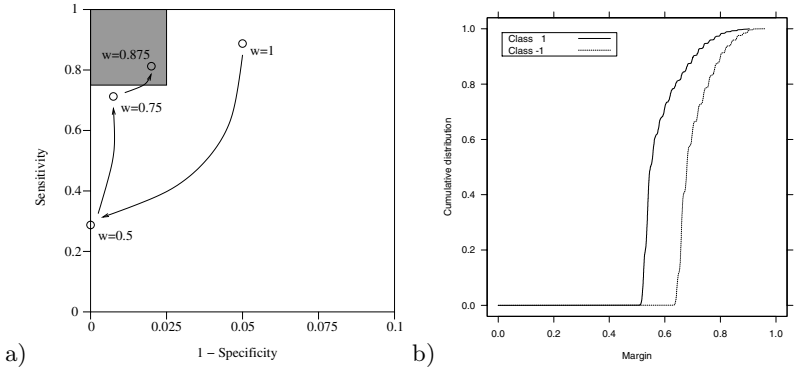

Fig. 2. a) The results of the tuning procedure in terms of sensitivity and specificity, estimated by 10-fold crossvalidation: the parameter w is initialized to 1. The gray square region on top left indicates the constrain target region A. The final value of the parameter is $w^* = 0.875$. b) The corresponding margin distribution for the two classes.

the development of a variant H_w of the AdaBoost characterized by higher sensitivity with respect to the AdaBoost itself when $w > 1$, by higher specificity when $w < 1$. For $w = 1$, we obtain AdaBoost. In [7], a procedure is also proposed for the automatic selection of an optimal cost parameter w^* in order to satisfy or to get as close as possible to admissible sensitivity – specificity constraints. The problem can be addressed as a minimization problem of the real function $\Delta : [0, 2] \rightarrow \mathbb{R}^+$ defined as $\Delta(w) = \text{dist}(\phi_H(w), A) = \min_{a \in A} \|\phi_H(w) - a\|$, describing the distance of a suitable target region A in the ROC space and the ROC curve $\phi_H(w)$. The problem admits a solution, not necessarily unique: the possible optimal cost parameters are selected by $w^* = \text{argmin}_w \Delta(w)$.

We applied the SSTBoost to our training set by using maximal classification trees as base learners (100 models). As discussed in Sec. 3, the target region A was defined by the constraints $Se > 0.75$, $Sp > 0.975$. To obtain predictive estimates of both sensitivity and specificity we used 10-fold crossvalidation. The steps of the automatic procedure for the selection of the parameter w are sketched in Fig. 2a, showing that the procedure falls into the region A in only 4 optimization steps. Fig. 2b can explain how SSTBoost works in terms of the margin. For $w^* = 0.875$ (and in general for $w \neq 1$), the margin of the two classes is differently maximized; the margin of the negative examples results higher than the margin of the positives one. The SSTBoost concentrates more resources to learn the negative examples.

3.2 Regularization

Regularization techniques for AdaBoost have been already discussed in literature, e.g. as in [8]. The alternative method introduced in [9] consists in a bias-variance control procedure based on removal of training data points according to a suitable threshold of an hardness measure, to be interpreted as the classification task difficulty for the predictors. Such hardness measure is linked to the dynamics of the AdaBoost weights, while the threshold can be evaluated by minimizing the generalized error of a loss function. In details, the degree of hardness for each training point $p_i \in D$ can be derived by the misclassification ratio of a point $\rho(p_i)$ in the aggregation of K AdaBoost models M_k, yielding a family of telescopic training subsets $\{D_r\}_{r \in (0,1]} = \{p_i \in D | \rho(p_i) < r\}$, shaved of the r-hardest points. After the optimal parameter is detected, the training set can be pruned by shaving the hardest points and the classifier retrained. We denote this regularized cost-sensitive version of AdaBoost by SSTBoostReg. In the present task, a portion of the training data is put aside as validation set for the threshold estimation: results are discussed in Sec. 4.

3.3 Spatial Analysis

Given a test image, the classification is based on a moving window procedure. The SSTBoostReg is applied and the corresponding output $H_{w^*}(\mathbf{x}) \in [-1, 1]$ is assigned to the central pixel of the window. For a rejection threshold $T_0 = 0$, a crude application of the SSTBoost leads to classify as craters all the pixels such

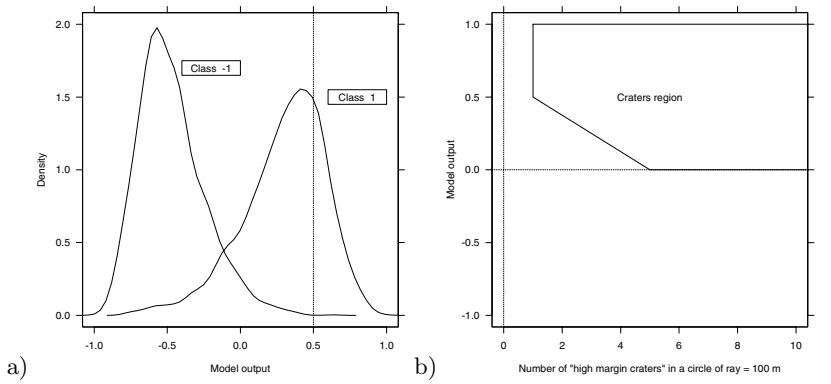

Fig. 3. a) Class densities of the 10-fold SSTBoostReg outputs. The vertical line indicates the threshold T_u for the identification of "high margin craters". b) Class labels are assigned according to the pair model output - density of "high margin craters".

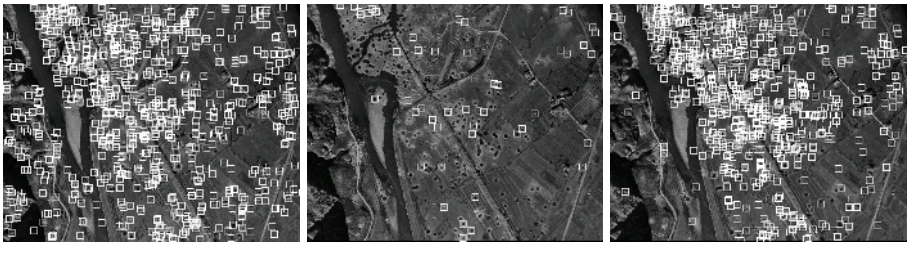

Fig. 4. The left image shows the classification obtained by applying SSTBoostReg, without spatial analysis. The middle image shows the "high margin craters" and the right image includes all the craters recognized by the local thresholding procedure.

that $H_{w^*}(\mathbf{x}) > T_0$ (obviously, groups of adjacent pixels over this threshold shall be aggregated). According to the results reported in Tab. 2, where the different versions of AdaBoost are compared, a percentage of false alarms of about 2% is expected. We now introduce an heuristics for reducing the percentage of false alarms, motivated by the observation that the craters are not randomly spatially distributed but they are clustered near the main targets. As shown in Fig. 3a, nearly ever a negative example is classified as positive with a response above 1/2 (by 10−fold crossvalidation). We can thus use a threshold $T_u = 1/2$ for identifying what we call "high margin craters". The idea is now to exploit the spatial density of "high margin craters" for locally modifying the rejection threshold of the model: the higher the density of "high margin craters", the lower the rejection threshold T. In particular, the threshold is decreased linearly from T_u to T_0 as the density of "high margin craters" increase from 1 to 5, and it remains constant for higher densities of "high margin craters" (see Fig. 3b for details). The "high margin craters" identified on a test image are shown in Fig. 4: as can

Table 2. Accuracy, sensitivity and specificity, including standard deviations, for AdaBoost, SSTBoost and SSTBoostReg (all obtained by aggregating 100 tree models)

Classifier	$Acc. \pm SD$	$Sens. \pm SD$	$Spec. \pm SD$
AdaBoost	0.92 ± 0.01	0.88 ± 0.02	0.95 ± 0.01
SSTBoost	0.91 ± 0.01	0.81 ± 0.02	0.98 ± 0.01
SSTBoostReg	0.92 ± 0.01	0.83 ± 0.02	0.98 ± 0.01

be easily seen, no false alarms are introduced at this level. Fig. 4 also includes the craters identified by the local thresholding procedure.

4 Performance Evaluation

The performances of AdaBoost, SSTBoost and SSTBoostReg, computed by 10−fold crossvalidation, are compared in Tab. 2: as expected, SSTBoost outperforms standard AdaBoost in terms of specificity (0.95 for AdaBoost, 0.98 for SSTBoost), meanwhile AdaBoost achieved higher global accuracy (0.92 for AdaBoost, 0.91 for SSTBoost). By looking at the results of the classifiers reported in Tab. 1, only SSTBoost falls into the target region A determined by the sensitivity – specificity constraints. The application of the regularization procedure allowed the global accuracy to be increased, improving from 0.91 of SSTBoost to 0.92 of SSTBoostReg, without decreasing neither specificity (0.98

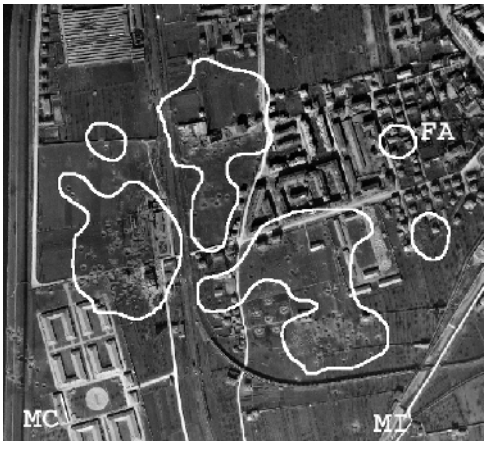

Image	MC	MI	FA
TS1	0	0	1
TS2	3 (13-11-5)	0	1
TS3	1 (9)	2	0
TS4	0	0	1
TS5	0	0	1
TS6	0	0	1
TS7	0	3	1
TS8	0	5	0
TS9	0	0	1
TS10	0	0	2
TS11	1 (16)	0	3
TS12	0	0	1
TS13	0	7	1
TS14	0	5	4
TS15	1 (10)	11	3
TS16	0	2	2
TS17	0	0	4
TS18	1 (18)	6	4
TS19	1 (> 500)	17	1
TS20	0	0	1

Fig. 5. Example of error detection based on the analysis of the risk map: the contour lines indicate regions with non null density of craters (by machine learning). The figure includes examples of missed craters clusters (MC), missed isolated craters (MI) and false alarms (FA). For each test image the values for MC (with relative number of craters), MI and FA are also reported.

for both SSTBoost and SSTBoostReg) nor sensitivity (0.81 for SSTBoost, 0.83 for SSTBoostReg).

The SSTBoostReg procedure coupled with the spatial analysis allowed the avoidance of a high number of false alarms without reducing too much the sensitivity of the model. The final output of the system is a map of the spatial density of craters. Performances were then evaluated in terms of number of false alarms in unbombed areas (FA), number of missed isolated craters (MI) and number of missed clusters of craters (MC), obtained by manual census. An example of error detection is shown in Fig. 5.

A set of 20 images was used for evaluating the model performances. The model failed to identify big clusters of craters only once (but the quality of the corresponding image is really very low). In 5 of the 20 images a small isolated cluster of craters was missed and only a total of 58 isolated craters were missed. The number of false alarms is considerably low, with an average of 1.6 false alarms per image (see Fig. 5).

5 Conclusions

The classification system was developed within the UXB-Trentino Project (`http://uxb.itc.it`), funded by the Province of Trento. The new risk map is used for public consultation in case of building planning, with semi-automatic production of risk reports. Consultation of the risk map under indication of the Civil Defense Department has become a standard procedure; its use as a mandatory procedure for new construction works is being considered.

References

1. Burl, M.C., Asker, L., Smyth, P., Fayyad, U.M., Perona, P., Crumpler, L., Aubele, J.: Learning to recognize volcanoes on Venus. Machine Learning **30** (1998) 165–194
2. Schapire, R., Freund, Y., Bartlett, P., Lee, W.: Boosting the margin: A new explanation for the effectiveness of voting methods. Annals of Statistics **26** (1998) 1651–1686
3. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth, Belmont (CA) (1984)
4. Breiman, L.: Bagging predictors. Machine Learning **24** (1996) 123–140
5. Ting, K.M., Zheng, Z.: Boosting trees for cost-sensitive classifications. In: European Conference on Machine Learning. (1998) 190–195
6. Fan, W., Stolfo, S.J., Zhang, J., Chan, P.K.: AdaCost: misclassification cost-sensitive boosting. In: Proc. 16th International Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA (1999) 97–105
7. Merler, S., Furlanello, C., Larcher, B., Sboner, A.: Automatic model selection in cost-sensitive boosting. Information Fusion **4** (2003) 3–10
8. Rätsch, G., Onoda, T., Müller, K.: Soft margins for Adaboost. Machine Learning **42** (2001) 287–320
9. Merler, S., Caprile, B., Furlanello, C.: Bias-variance control via hard points shaving. International Journal of Pattern Recognition and Artificial Intelligence **18** (2004) 891–903