H.G. Chew, D.J. Crisp, R.E. Bogner, and C.C. Lim. Target Detection in Radar Imagery using Support Vector Machines with Training Size Biasing. In *Proceedings of the Sixth International Conference on Control, Automation, Robotics and Vision (ICARCV 2000)*, CDROM, Singapore, 2000.

Correspondence:

Hong-Gunn Chew
Department of Electrical and Electronic Engineering
The University of Adelaide
Adelaide
SA 5005
Australia
hgchew@eleceng.adelaide.edu.au

-blank page-

TARGET DETECTION IN RADAR IMAGERY USING SUPPORT VECTOR MACHINES WITH TRAINING SIZE BIASING

Hong-Gunn Chew † *, David J. Crisp † ‡, Robert E. Bogner † *, Cheng-Chew Lim *
email: hgchew@eleceng.adelaide.edu.au, david.crisp@dsto.defence.gov.au,
{bogner, cclim}@eleceng.adelaide.edu.au

† CRC for Sensor Signal and Information Processing,
SPRI Building, Technology Park, Mawson Lakes Boulevard,
Mawson Lakes, SA 5095, Australia

* Department of Electrical and Electronic Engineering,
The University of Adelaide, SA 5005, Australia

‡ Surveillance Systems Division,
Defence Science and Technology Organisation,
PO Box 1500, Salisbury, SA 5108, Australia

Keywords: pattern recognition, Support Vector Machine, target detection, training size biasing

ABSTRACT

We considered the use of Support Vector Machines (SVMs) in the classification of synthetic aperture radar images. The images are divided into two types: those images with man-made vehicles, which are targets of interest, and those images without. This real world problem is important and non-trivial due to the low resolution of the images, small targets and large number of images to be classified. This paper describes the procedure required in the manipulation of the images to create the training and test sets. A series of SVMs are trained and validated with a test set to determine the best performing SVM. The resulting SVM has a probability of detection of 89% at a probability of false alarm of 10%.

1 INTRODUCTION

The Support Vector Machine (SVM) paradigm is a classification paradigm based on statistical learning [1][2][7]. It is relatively new in the pattern recognition area and has only been researched extensively for the past few years. One major advantage of SVMs over more traditional classifiers is that pre-processing, or feature extraction, of the data is not essential before training or classifying. This removes a large variable in the search for the best performing classifier.

The kernel functions in SVMs allow the classifier to be extremely flexible to suit the requirements of the classification problem. There are kernel functions that emulate the traditional classifiers. These kernel functions allow the SVM performance to be checked by reference to corresponding traditional classifier solutions, as Schölkopf *et al.* [6] have shown for radial basis functions.

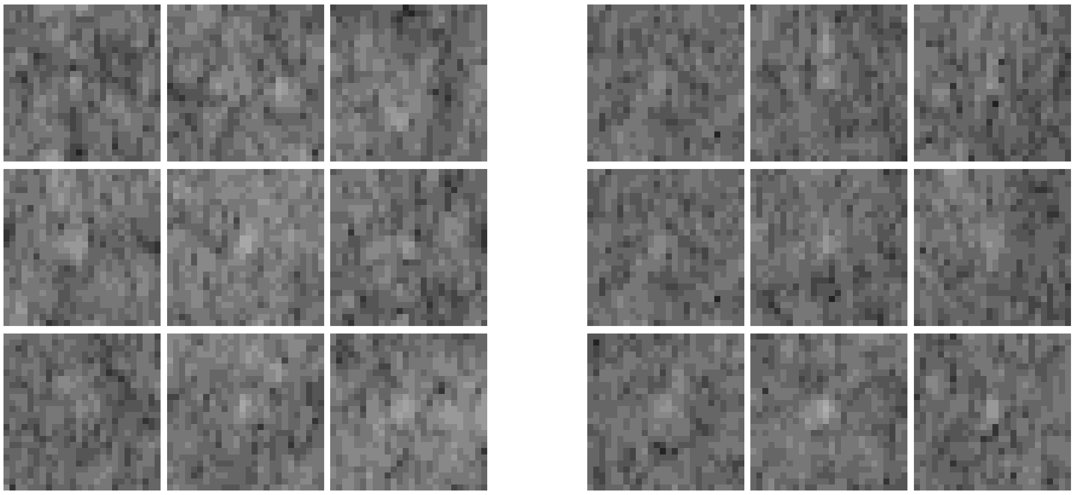
In target detection, a series of images are checked for the existence of a possible target. In the target detection problem we are looking at, the number of images without targets far outnumbers the images with targets. The training set for the SVM has considerably more non-target images than target images, and the ratio could be as much as 1000:1. We propose a novel scheme in Support Vector Learning by applying different error weightings to each of the two classes with a ratio to offset the training class size difference. The scheme is tested on a large set of real world data and the results are promising. The performance of the resulting SVMs compares well with other classifiers.

This paper is organised into 6 sections. Section 2 describes the target detection problem, while section 3 introduces support vector machines and extends the theory to machines with separate error weightings. Section 4 lays out the procedure in training an SVM, and section 5 reports on the results obtained. Section 6 gives the conclusion of this paper.

2 THE TARGET DETECTION PROBLEM

The images used in this paper are wide area images of open land obtained with a synthetic aperture radar (SAR). The objective is to determine the existence of man-made objects (vehicles) in the images, and to obtain the location of the objects if they exist. Since the images are of low resolution, the vehicles are typically only a few pixels in size.

The wide area images are first processed with a fast detector (pre-screener) that has a high probability of



(a) (b)

Figure 1 Examples of (a) non-target images, and (b) target images.

detection (P_d) but also a high probability of false alarm (P_f). The detector picks out 64×64-pixel regions where there may be a possible target in the centre of the region. This process reduces the amount of data that needs to be processed downstream as most of the regions without targets are removed from consideration.

The classification task is to reduce the probability of false alarm while maintaining the probability of detection of targets within the pre-detected regions. In the data set used for training, the vehicles have been positioned in known locations. The regions can be labelled as hits (with vehicles at the centre), and as false alarms (without a vehicle at the centre). Examples of the images are shown in Figure 1. It can be seen that the target images look similar to the non-target images to the untrained eye.

We have a total of about 150,000 non-target images and 1,000 target images in the database. Of these, half of the images are used as the training set and the remaining half are used as the test set. The aim is to obtain a machine that has a high probability of detection while retaining a low probability of false alarm. Typically, a P_d of greater then 90% with a P_f of 10% is required.

3 SUPPORT VECTOR MACHINES

Consider a set of l data vectors

$$\{\boldsymbol{x}_i, y_i\}, \quad i = 1, \ldots, l, \quad y_i \in \{-1,1\}, \quad \boldsymbol{x}_i \in \Re^d$$

where $\boldsymbol{x}_i$ is the i-th data vector that belongs to a binary class y_i. We seek the hyperplane that best separates the two classes where by best we mean with the largest margin.

That is, we want to find the hyperplane

$$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$$

subject to the constraints

$$y_i (\boldsymbol{x}_i \cdot \boldsymbol{w} + b) \geq 1 - \xi_i \forall\ i$$

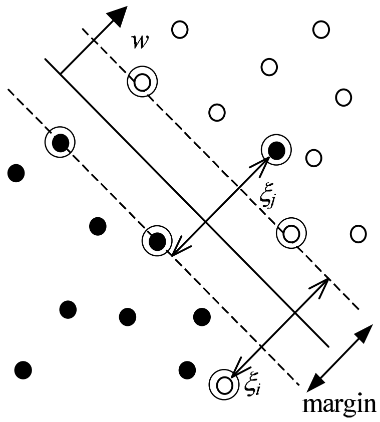


Figure 2 Two-dimension example of an SVM

that maximises the margin $2/\|\boldsymbol{w}\|_2$, while minimising the cost of the errors $C(\Sigma\ \xi_i)$, where $\boldsymbol{w}$ is normal to the hyperplane, $|b|/\|\boldsymbol{w}\|_2$ is the perpendicular distance from the hyperplane to the origin, $\|\boldsymbol{w}\|_2$ is the L2-norm of $\boldsymbol{w}$, ξ_i is the slack variable for the i-th data vector and C is the cost penalty. Figure 2 shows a two-dimension example of an SVM. The objective function for minimisation is defined to be

$$\|\boldsymbol{w}\|_2/2 + C(\Sigma\ \xi_i)\ . \tag{1}$$

This problem can be expressed in its dual form as

Problem D:

Maximise

$$L_D \equiv \sum_i \alpha_i - \tfrac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{2}$$

subject to

$$0 \leq \alpha_i \leq C \tag{3}$$

$$\sum_i \alpha_i y_i = 0 \tag{4}$$

where K is the kernel function that results in non-linear hyperplanes, and α_i are the Lagrange multipliers. ***Support vectors*** are vectors with $\alpha_i > 0$, and ***bounded support vectors (BSVs)*** are support vectors with $\alpha_i = C$ and $\xi_i > 0$.

The decision function from the trained SVM is

$$f(\mathbf{z}) = \operatorname{sgn}\left(\sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{z}) + b\right) \qquad (5)$$

3.1 Training Support Vectors Machines with Uneven Class Sizes

In target detection, we are usually faced with the problem of having a large number of non-target images compared to a relatively small number of target images. We seek a machine that provides a high probability of detection with a low probability of false alarm.

In our experience, we have found that as a rule of thumb, an SVM should have a large number of bounded support vectors to prevent the machine from over-fitting to the training set. Constraint (4) requires that the sum of α_i for positive vectors be equal to that for negative vectors. Since the resulting SVM has a large number of BSVs, we have

$$\begin{aligned} &\Sigma\alpha_i \approx B_+ \cdot C, && \forall\, y_i = +1 \\ &\Sigma\alpha_i \approx B_- \cdot C, && \forall\, y_i = -1 \\ &B_+ \approx B_- && \end{aligned} \qquad (6)$$

where B_+ is the number of positive BSVs, and B_- is the number of negative BSVs. Equation (6) shows that in training a SVM, the resulting machine will have similar numbers of positive and negative BSVs. Recall that BSVs have $\xi_i > 0$, and hence, having crossed the margin, are called ***margin errors***. With a large number of BSVs, the margin error rate is approximately equal to the classification error rate, where the margin error rate is the number of margin errors per training point. In target detection terms, the positive classification error rate is the miss detection rate, while the negative classification error rate is the false alarm rate. This gives the positive and negative error rates as B_+/N_+ and B_-/N_- respectively, where N_+ is the number of target images and N_- is the number of non-target images.

The error rates ratio becomes

$$\begin{aligned} B_+ / N_+ &: B_- / N_- \\ 1 / N_+ &: 1 / N_- \\ N_- &: N_+ \end{aligned} \qquad (7)$$

which means that with a large N_-, the positive error rate is much larger than the negative error rate. This results in the machine biasing towards the negative class, or a low probability of detection.

We propose a strategy to handle and correct this bias effect by setting a ratio for the different error weightings of the positive (C_+) class and the negative (C_-) class. Equation (6) becomes

$$\begin{aligned} &\Sigma\alpha_i \approx B_+ \cdot C_+, && \forall\, y_i = +1 \\ &\Sigma\alpha_i \approx B_- \cdot C_-, && \forall\, y_i = -1 \\ &B_+ \cdot C_+ \approx B_- \cdot C_- && \end{aligned} \qquad (8)$$

and the error rates ratio (7) becomes

$$\begin{aligned} B_+ / N_+ &: B_- / N_- \\ 1 / (C_+ N_+) &: 1 / (C_- N_-) \\ C_- N_- &: C_+ N_+. \end{aligned} \qquad (9)$$

By setting

$$C_+ / C_- = N_- / N_+ \qquad (10)$$

a machine that offers a balanced error rate in both classes is derived.

3.2 Support Vector Machines with Positive and Negative Error Weightings

It is natural for SVMs to be extended with different error penalties for the positive and negative classes as first proposed by Osuna *et al.* [4]. Since C is the error penalty attached to the data vectors, two Cs can be used to implement two different error penalties. Positive vector errors are weighted by C_+, and negative vectors errors by C_-.

We can modify the objective function (1) to

$$\|\boldsymbol{w}\|_2/2 + \Sigma\, C_i\xi_i \qquad (11)$$

to extend it to having individual Cs, where $C_i = C_+$ for positive class vectors and $C_i = C_-$ for negative class vectors. Problem D can be extended to

Problem D':

Maximise

$$L_D \equiv \sum_i \alpha_i - \tfrac{1}{2}\sum_{i,j} \alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \qquad (12)$$

subject to

$$0 \le \alpha_i \le C_+ \qquad \forall\, y_i = +1 \qquad (13)$$

$$0 \le \alpha_i \le C_- \qquad \forall\, y_i = -1 \qquad (14)$$

$$\sum_i \alpha_i y_i = 0 \qquad (15)$$

where the Lagrange multipliers α_i, which are the multipliers for support vectors, are bounded by C_+ for the positive vectors and C_- for the negative vectors.

4 TRAINING A SUPPORT VECTOR MACHINE

A 16×16 window in the centre of each region was used. This reduced the computational and memory requirements of training while retaining enough information for classification. The training set is also trimmed to reduce the number of training points, to speed up the training. The resulting training set has 498 target images and 9930 non-target images, and a class size ratio of 1:20.

The first step in getting the data ready for training is to transform the image data into vectors. Each 16×16-pixel window was vectorised to a 256-element vector.

Next, we choose a suitable kernel, associated kernel parameters, as well as an error penalty. This process is largely iterative, although past experiences do provide some hints to what may be suitable. Linear SVMs are used as they are computationally fast and provide a good starting point. We have found that a good rule in

Training Parameters		Training Results			Test Results		
C_+ ($\times 10^{-6}$)	C_- ($\times 10^{-6}$)	+ve SV (B_+)	-ve SV (B_-)	% of errors +ve / –ve	% of errors +ve / –ve	P_d/P_f (% / %)	P_d at 10% P_f (%)
0.1	0.1	498 (498)	528 (475)	100 / 0.0	100 / 0.0	0 / 0.0	87
2	0.1	252 (234)	4858 (4843)	12 / 8.8	14 / 8.9	86 / 8.9	88
100	0.1	54 (0)	9387 (9381)	0 /53.4	1 /55.1	99 /55.1	86
1	1	491 (489)	516 (465)	79 / 0.1	79 / 0.1	21 / 0.1	87
20	1	164 (133)	3038 (2967)	9 / 8.5	11 / 9.1	89 / 9.1	89
1000	1	69 (0)	5198 (5147)	0 /25.2	4 /25.8	96 /25.8	87
10	10	422 (413)	475 (363)	57 / 0.3	58 / 0.5	42 / 0.5	88
200	10	147 (77)	2217 (2126)	6 / 8.1	12 / 9.0	88 / 9.0	89
10000	10	91 (0)	3047 (2971)	0 /15.3	9 /16.0	91 /16.0	87
100	100	381 (354)	457 (292)	47 / 0.4	49 / 0.8	51 / 0.8	89
2000	100	145 (44)	1772 (1641)	4 / 7.8	16 / 8.9	84 / 8.9	85
10000	100	112 (0)	2102 (1982)	0 /11.3	12 /12.4	88 /12.4	85
1000	1000	358 (316)	459 (250)	41 / 0.5	44 / 1.2	56 / 1.2	86
20000	1000	137 (34)	1608 (1451)	3 / 8.0	17 / 8.8	83 / 8.8	85
100000	1000	113 (0)	1749 (1603)	0 /10.1	15 /11.1	85 /11.1	83
10000	10000	351 (305)	457 (242)	42 / 0.5	44 / 1.3	56 / 1.3	85
100000	10000	175 (77)	1283 (1113)	7 /5.49	19 / 6.4	81 / 6.4	87

Table 1 Training Results with Linear SVMs.

choosing the error penalty is to achieve about 80% of support vectors being bounded. A different error penalty is applied to the target images (positive class) and to the non-target images (negative class) with a ratio of 20:1 as stated by equation (10) to counter the effects of different training class sizes.

We used a Matlab function written in C that is based on Platt's SMO [3] to train and test the SVM.

5 RESULTS

The training of the SVMs was performed on a 166MHz Sun Ultra 5 and it took one week of computational time to produce the results discussed below. The large number of training vectors increased the computational load considerably [5], but the time taken is still significantly less than what would be required to search for features used in traditional classifiers.

Table 1 tabulates some of the results that were obtained. An important observation is that in the cases where C_+ and C_- are equal, the percentages of positive errors made in the training were much greater than the percentages of negative errors made, as expected from equation (7). This resulted in the machines having a bias towards a lower probability of false alarm. By adjusting the ratio of C_+ to C_-, we obtained different percentages of training and test errors, as expected from equation (9).

With the error weighting ratio at 20:1, the SVMs consistently provide a good compromise between P_d and P_f. The linear SVM trained with the positive error weighting (C_+) set to 20×10^{-6} and the negative error weighting (C_-) set to 1×10^{-6}, works well on the problem.

Private discussions with other researchers using traditional classifiers have indicated that the performance of SVMs compares well with traditional classifiers.

5.1 Performance Metrics

Receiver Operating Characteristic (ROC) curves are widely used to view the performance of a range of classifiers, and to select a suitable classifier. In this target detection problem, the ROC plots P_d against P_f. P_f increases when P_d is increased. The probability of detection and probability of false alarm are defined as

$$P_d = \frac{N_+ - F_+}{N_+} \tag{16}$$

$$P_f = \frac{F_-}{N_-} \tag{17}$$

where N_+ is the number of target images, N_- is the number of non-target images, F_+ is the number of target images not detected by the machine, and F_- is the number of non-target images detected as targets.

Since a trained SVM will only supply a pair of P_d and P_f values, this results in only one point on the ROC curve. We need a way of creating a series of machines with varying P_d-P_f pairs. The decision function is generally expanded to

$$f(\mathbf{z}) = \mathrm{sgn}\left[\left(\sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{z}) + b\right) - \theta\right] \tag{18}$$

where θ is the threshold constant, to adjust the performance of the machine towards a higher P_d or lower P_f. Increasing the θ decreases both P_d and P_f (moving the performance towards the lower left), while

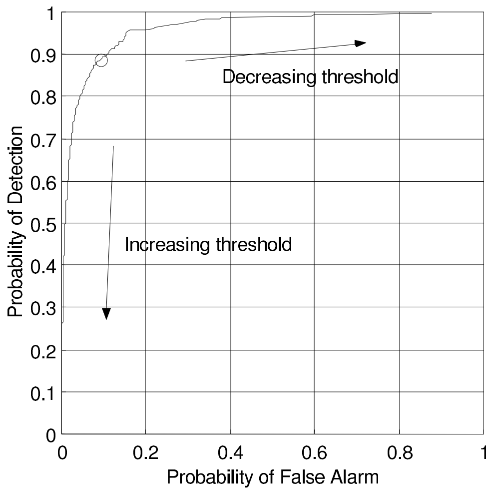


Figure 3 Result of changing the threshold θ. The circle indicates the point where $\theta = 0$.

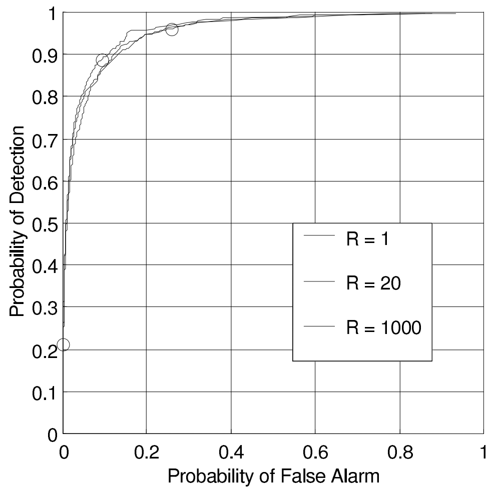


Figure 4 ROC curves for SVMs with $C_- = 10^{-6}$ and $R = C_+/C_-$.

decreasing the θ increases both P_d and P_f (moving the performance towards the upper right). Figure 3 shows the effects of adjusting θ.

The ROC curves, calculated from the trained SVMs, are plotted in Figure 4, and the P_d of each SVM at 10% P_f is recorded in Table 1. We are interested in 10% P_f as it provides a point where we can compare the different SVMs and is also a good compromise between the probabilities of detection and false alarm.

SVMs with an error weighting ratio at 20:1 consistently produce one of the highest P_d at 10% P_f when compared with SVMs with an error weighting ratio at 1:1. This trend follows until the number of bounded support vectors falls to 30% of support vectors, as shown in Figure 5.

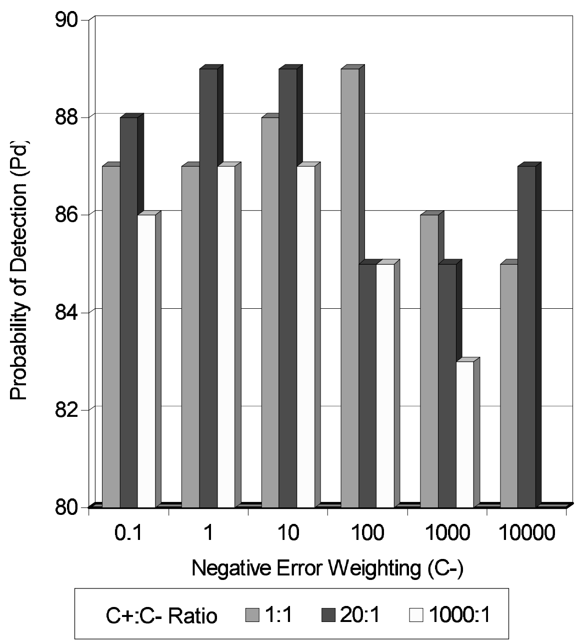


Figure 5 Probability of Detection with 10% Probability of False Alarm.

6 CONCLUSION

We have shown that the use of SVMs in the detection of targets was successful and compares well with other researchers' results using traditional classifiers. The ability of SVMs to use raw image data, and perform well is remarkable and eliminates the need for pre-processing. However, it should be noted that while data pre-processing should improve performance and training times, the time involved in determining the pre-processing required could outweigh the small improvements.

The proposed strategy, of adjusting the error weightings for the positive and negative classes using $C_+/C_- = N_-/N_+$, was shown to improve the performance of the SVM by removing the bias towards fewer errors on the negative class. The resulting SVM has an even percentage of errors. We have verified that the best performing SVM has the ratio of error weightings ($C_+ : C_-$) equal to the inverse ratio of training class size ($N_+ : N_-$), and at least 80% of the support vectors are bounded.

Although linear SVMs are the least complex form of SVMs, our results show the impressive performance of these SVMs. We can improve on the results by using a different kernel, such as the radial basis function kernel, but at the expense of increased computational load.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the Defence Science and Technology Organisation, Australia, for providing the synthetic aperture radar images.

REFERENCES

[1] Burges, C.J.C., "A tutorial on Support Vector Machines for Pattern Recognition", Data Mining and Knowledge Discovery, Vol. 2, No. 2, 1998.

[2] Cortes, C. and Vapnik, V., "Support vector networks", Machine Learning, Vol. 20, pp.1-25, 1995.

[3] Platt, J. C., "Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines" Technical Report MSR-TR-98-14, Microsoft Research, 1998.

[4] Osuna, E.E., Freund R. and Girosi, F., "Support Vector Machines: Training and Applications", Technical Report AIM-1602, MIT AI Lab, 1997.

[5] Osuna, E.E. and Girosi, F., "Reducing the run-time complexity of Support Vector Machines" Proceedings of the 14th International Conference on Pattern Recognition, Brisbane, Australia, 1998.

[6] Schölkopf, B., Sung, K., Burges, C.J.C., Girosi, F., Niyogi, P., Poggio, T. and Vapnik, V., "Comparing Support Vector Machines with Gaussian Kernels to Radial Basis Function Classifiers" IEEE Transactions of Signal Processing, Vol. 45, pp.2758-2765, 1997.

[7] Vapnik, V., "The Nature of Statistical Learning Theory", Springer-Verlag, New York, 1995.