

# Radar Pulse Classification using Support Vector Machines

Gregory P. Noone

Electronic Warfare Division, Defence Science and Technology Organisation

PO Box 1500 Salisbury, South Australia 5108

email: greg.noone@dsto.defence.gov.au

## Abstract

A high dimensional pulse characteristic is used in order to classify real radar signals as belonging to various similar, but distinct, categories. The pulse characteristic needs to be reasonably time and location invariant to allow robust future classifications. We used a supervised learning process known as the Support Vector Machine algorithm for the classification problem. The advantages of this approach is that the algorithm is well known for its generalising ability, its solution is global and unique, and a data *feature selection* step is *not* required. Our results indicate that there exists non-conventional pulse characteristics suitable for rigorous pulse classification.

## Introduction and Background

Pulse classification involves determining the categories of signals which are being transmitted in a given environment. In particular, we wish to be able to distinguish between various categories which may contain common signal patterns or characteristics. For instance, differing pulse categories can have signals with very similar radio frequency (RF) and pulse duration (PD) values. In addition, RF and PD values may not be so stable over time and location.

Hence we require other pulse characteristics, that would be reasonably time and location invariant, that could robustly categorise the signal. That is to say, that data collected at one time and location with a certain signal to noise ratio (SNR) could be used to classify pulses at another time and location with a possibly differing SNR. We also require the classification algorithm itself to be reasonably insensitive to noise effects and any small changes in the data. This allows the algorithm to generalise well by successfully classifying not only on seen *training* data, but also on unseen *testing* data.

## Support Vector Machines

For the classification process, we used the *Support Vector Machine* (SVM) algorithm. This is a recent approach for solving supervised classification problems, or “learning from examples”. It is quickly growing in popularity due to its generalising ability. In essence, such an approach maximises the margin between the training data (where the category of each data

point is known) and the decision boundary between two competing categories. This allows robust classification of future unseen data whose categories are unknown. Maximising the margin casts the problem as a *Quadratic Optimisation* one. The subset of patterns that are closest to the decision boundary are called *supporting vectors* and are automatically determined by the algorithm. Although SVMs are only binary classifiers, it is simple to generalise them to a classifier for multiple classes, as is discussed later.

We give a very brief overview of the mathematics of SVMs. For a more detailed presentation, the reader is referred to [1] and [2]. Suppose for the linearly separable binary classification problem, we have a set of examples, or training data,  $\mathbf{x}_i, y_i$ , where  $y_i \in -1, 1$  and defines class membership,  $\mathbf{x}_i \in \mathbb{R}^N$ ,  $i = 1, \dots, l$ . We wish to construct a hyperplane  $\mathbf{w} \cdot \mathbf{x} + b = 0$  so that the margin between the hyperplane and the nearest point is maximised. It can be shown that this leads to the following quadratic optimisation problem:

$$\min_{\mathbf{w}} \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) \quad (1)$$

subject to

$$y_i((\mathbf{w} \cdot \mathbf{x}_i) + b) \geq 1, \quad i = 1, \dots, l. \quad (2)$$

Note that (2) forces a rescaling on  $(\mathbf{w}, b)$  so that the point closest to the hyperplane has a distance of  $1/\|\mathbf{w}\|$ . Often in practice a separating hyperplane does not exist. Hence we need to relax the constraints (2) by introducing slack variables  $\xi_i \geq 0$ ,  $i = 1, \dots, l$ . The optimisation problem now becomes, for a user defined positive finite constant  $C$ :

$$\min_{\mathbf{w}, \xi} \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) + C \sum_{i=1}^l \xi_i \quad (3)$$

subject to

$$y_i((\mathbf{w} \cdot \mathbf{x}_i) + b) \geq 1 - \xi_i, \quad i = 1, \dots, l \quad (4)$$

$$\xi_i \geq 0, \quad i = 1, \dots, l. \quad (5)$$

Note that for a classification error to occur, we require  $\xi_i \geq 1$ . Hence  $\sum_{i=1}^l \xi_i$  is an upper bound on the number of misclassifications on the training set. The additional term of Equation (3), with  $0 < C < \infty$ , balances the contribution of minimising  $(\mathbf{w} \cdot \mathbf{w})$  (i.e. maximising the margin) with penalising solutions for which the  $\xi_i$  get too large.

By introducing Lagrange multipliers  $\alpha_i$  and using the Kuhn-Tucker theorem of optimisation theory, we can pose the equivalent dual optimisation problem:

$$\max_{\alpha} \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \quad (6)$$

subject to

$$0 \leq \alpha_i \leq C, \quad i = 1, \dots, l \quad (7)$$

$$\sum_{i=1}^l \alpha_i y_i = 0. \quad (8)$$

The solution is given by

$$\mathbf{w} = \sum_{i=1}^l y_i \alpha_i \mathbf{x}_i. \quad (9)$$

The non-zero  $\alpha_i$ 's correspond to the so-called *support vectors*  $\mathbf{x}_i$  that help define the boundary between the two classes. All other training examples with corresponding zero  $\alpha_i$  values are now rendered irrelevant and automatically satisfy constraint (4) with  $\xi_i = 0$ . We can now write down the hyperplane decision function, for the vector  $\mathbf{x}$ , as:

$$f(\mathbf{x}) = \text{sgn} \left( \sum_{i=1}^l y_i \alpha_i \cdot (\mathbf{x} \cdot \mathbf{x}_i) + b \right). \quad (10)$$

Note that in the objective function (6) and decision function (10), the input examples only appear as dot products. This leads to two interesting properties. First, the complexity of the algorithm is not strongly dependent on the dimension of the input data, but rather the *number* of data examples. Secondly, to allow for more general decision surfaces, the dot product can simply be replaced by a suitable kernel function  $k$  [1]. Hence the objective function to be maximised can now be written as

$$\sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j), \quad (11)$$

with the constraint equations (7) and (8) unchanged. The decision function then becomes, for the vector  $\mathbf{x}$ :

$$f(\mathbf{x}) = \text{sgn} \left( \sum_{i=1}^l y_i \alpha_i \cdot k(\mathbf{x}, \mathbf{x}_i) + b \right). \quad (12)$$

The  $\alpha_i$ 's are determined from the solution to the Quadratic Program (11), (7) and (8). The bias parameter,  $b$ , is determined from (10) by using two arbitrary support vectors from known but opposing classes.

By replacing the dot products with kernel functions, we have effectively mapped the input data to a higher dimensional space. It is then in this higher dimensional space that we still attempt to construct a data separating hyperplane so as to maximise the margin. In the lower dimensional data space, this hyperplane becomes a non-linear separating function. An example of a commonly used kernel function is  $k(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^q$ , where  $q \in \mathbb{Z}^+$ .

Although the above analysis only applies to the binary classification problem, it is easily extended to the multiclass problem. For  $K$  classes, we simply train  $\binom{K}{2} = K(K-1)/2$  binary classifiers to take into account all combination pairs of classes ("One-against-One classifiers"). To classify a new data vector, all  $\binom{K}{2}$  binary classifiers are applied to it, and a simple voting strategy is employed. The label of the class with the most votes is chosen.

## Radar Pulse Classification Problem

We thereby chose an appropriate high dimensional pulse characteristic in order to classify the signals as belonging to various distinct categories. We were careful to ensure that the dimension of this pulse characteristic was reduced to the value of the smallest dimension across all

categories considered. In this way, the classification process would only use the intrinsic pulse characteristic information, rather than any categorical differences in dimensions.

In this study we have chosen five different categories of pulse signals, all with similar RF and PD values. Each category was further subdivided into three PD modes of low, medium and high values. Each mode was considered separately in the classification problem as their PD values were widely separated. For each of the five categories, we have two data sets of signals digitally recorded at distances of 3 nmi and 6 nmi.

One of the main purposes of the study is to determine if the proposed high dimensional pulse characteristics recorded at 3 nmi of known categories can be used to classify pulses recorded at 6 nmi at a differing time, and vice-versa. Hence, we trained the SVM on 50 pulses from each of the categories of the 3 nmi data and tested it on all the 6 nmi data and vice-versa. This is a good test for the robustness of the pulse characteristic used as well as the generalisation abilities of SVMs. We also trained the SVM on a small amount of data recorded at both 3 nmi and 6 nmi (25 pulses each) and tested it on the remaining data at both these distances.

Finally, we calculated the sample mean pulse characteristic for each category at each distance. We then classified each pulse at the other distance by choosing that category which minimised the Euclidean distance between the sample means and the pulse being classified. This is a standard classification procedure and is used as a means of comparison with the SVM algorithm. Note that both methods are “data driven” and use no prior information (apart from category membership), so that a comparison is meaningful and valid.

## Results and Discussion

There is so far no theoretical approach to choosing an optimal kernel function and user defined  $C$  value [2]. When using the SVM algorithm for this problem, we chose the kernel function  $k(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^2$  which seemed to work well across all the data. We found that  $C = 100$  worked well for all cases, except when training the SVM with the 3 nmi data and testing on the 6 nmi data for modes 2 and 3, where  $C = 1$  was found to be more appropriate.

It was found that the data for mode 1 at both distances were comparatively noisy compared to modes 2 and 3. The data for mode 3 had a slightly higher SNR than for mode 2. The data at 3 nmi for modes 2 and 3 was, as expected, less noisy than the data at 6 nmi. We found that when the SVM was trained on high SNR data and then tested on data with a significantly lower SNR, a smaller value of  $C$ , which penalised classification errors less, was more appropriate.

For all five categories across three PD modes, there were a total of 4340 pulses, of which 2533 were recorded at a distance of 6 nmi and the remaining on another day at 3 nmi. The first two columns of Table 1 show the classification results when an SVM was trained on the data at one distance and tested on data from the other distance. The third column shows the classification results when an SVM was trained on a small amount of data from each distance and tested on all remaining data. The fourth and fifth columns are the classification results using the minimum Euclidean distance procedure based on the mean pulse characteristics as described earlier.

We quickly note that the SVM algorithm produced significantly better classification results than the traditional minimum Euclidean distance approach. Other traditional approaches were considered, such as correlation methods, but these fared even worse and are not reported here. In particular, for modes 1 and 2 where the amount of noise was significant, the SVM performed

	3 nmi train	6 nmi train	mixed train	3 nmi mean	6 nmi mean
Cat.	6 nmi test	3 nmi test	mixed test	6 nmi test	3 nmi test
a1	199/200	91/91	241/241	197/200	91/91
b1	71/193	71/123	255/266	62/193	33/123
c1	100/100	100/100	150/150	100/100	100/100
d1	190/193	137/141	276/284	190/193	136/141
e1	148/200	126/149	234/299	133/200	145/149
a2	97/97	60/60	107/107	97/97	60/60
b2	89/95	42/50	93/95	47/95	29/50
c2	190/190	140/140	280/280	190/190	140/140
d2	248/248	187/187	385/385	248/248	187/187
e2	165/200	133/143	280/293	153/200	110/143
a3	149/149	50/50	149/149	149/149	50/50
b3	176/180	42/46	176/176	180/180	44/46
c3	100/100	173/173	223/223	100/100	173/173
d3	190/190	175/175	315/315	190/190	175/175
e3	191/198	171/179	312/327	189/198	154/179
mean	90.9%	94.0%	96.8%	87.8%	90.0%

Table 1: Pulse classification results for each of the three PD modes. The first three columns are classification results for the SVM algorithm trained on 1) 3 nmi data and tested on 6 nmi data 2) 6 nmi data and tested on 3 nmi data 3) a mixture of 3 nmi and 6 nmi data and tested on unseen mixed data. Columns 4 and 5 reflect pulse classification results for the minimum squared error method for the sample mean calculated from 1) data at 3 nmi and tested on data at 6 nmi 2) data at 6 nmi and tested on data at 3 nmi. The results are given as the number of successfully classified pulses/the total number of pulses we attempted to classify.

much better than the minimum Euclidean distance approach. For mode 3, where the noise was less significant, both approaches gave comparable results.

Although the amount of pulse data was limited, applying the SVM algorithm to the pulse classification problem described above was generally very successful. It was found that categories *b* and *e* were the most difficult to separate, especially in mode 1 where there were substantial misclassifications. A misclassified *b* pulse was nearly always classified as an *e* category pulse and vice-versa.

All other categories were very close to being 100% separable in the examples we encountered. Note that it was easier to classify the 3 nmi data after training on the 6 nmi data than vice-versa. This is because the data at 3 nmi would be expected to be less noisy than the data at 6 nmi. However, the best performance was achieved when the SVM was trained on a small amount of data recorded from both 3 nmi and 6 nmi and tested on all of the remaining data. The issue of the noise spread of the training and testing data and how it affects the performance of an SVM, and the optimal choice of its user defined parameter *C*, deserves extra attention. This is especially important in view of the fact that the SNR of future signals to be classified will be generally unknown.

## Conclusion

Our results suggest that there exists non-conventional pulse characteristics that can help classify accurately and robustly pulses of distinct, but similar, categories. It also shows that SVMs trained on small amounts of noisy data can be successful in discriminating between very similar high dimensional data that are from differing categories. In addition, the SVM algorithm alleviates the problem of using high dimensional data so that a feature selection step is *not* required. Indeed, the complexity of the SVM algorithm is independent of the data dimension, but rather depends on the number of training examples. Further, because the SVM operates as a Quadratic Optimisation problem, the solution is global and almost certainly unique. This differs vastly from other well known learning machines, such as *Multilayer Perceptrons* (MLP), which can be very sensitive to its own initial configuration. Extra work on the effect of noise in the training and testing data and corresponding optimal  $C$  value is required.

## References

- B. Boser, I. Guyon, V. Vapnik, "A training algorithm for optimal margin classifiers", *Fifth Annual Workshop on Computational Learning Theory*, pp. 144–152, ACM Press, 1992.
- C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, Vol 2(2), pp. 1–43, 1998.