Proceedings of AICS-2007: 18th Irish Conference on Artificial Intelligence and Cognitive Science, Dublin, August 2007.

One-Class Support Vector Machine Calibration Using Particle Swarm Optimisation

Yang Liu, Michael G. Madden
Department of Information Technology,
National University of Ireland, Galway, Ireland
sharkyangliu@yahoo.co.uk; michael.madden@nuigalway.ie

Abstract. Population-based search methods such as evolutionary algorithms, shuffled complex algorithms, simulated annealing and ant colony search are increasingly used as automatic calibration methods for a wide range of numerical models. This paper proposes the use of particle swarm optimisation to calibrate the parameters a one-class support vector machine. This approach is developed and tested in the calibration of a one-class SVM, applied to several data sets. The results indicate that the proposed method is able to match or surpass the performance of a one-class SVM with parameters optimized using a standard grid search method, with much lower CPU time required.

1 Introduction

During the past two decades, a great deal of research has been devoted to the development of traditional classification methods (binary and multi-class classification) for pattern recognition [1]. Such research has focused primarily on four issues: (1) Determination of appropriate quantity and most informative kind of data (feature selection); (2) Dimension reduction for high dimensional features (feature selection); (3) Search for a classifier that can reliably solve linear or non-linear classification problems; and (4) Validation for the classifier. Non-linear classification methods such as Support Vector Machines (SVMs), K-Nearest Neighbour (KNN) classifier, Decision Tree and Artificial Neural Networks (ANNs) are widely used traditional classification problems.

However, for a significant number of practical problems, traditional *discriminating* classifiers that are trained using positive and negative examples are not directly applicable, because negative examples may be either rare, entirely unavailable or statistically unrepresentative. Such problems include industrial process control, text classification and image analysis. One-Class Classification (OCC) is emerging as a solution, which characterizes the target class, seeking to distinguish one class from the universal set of multiple classes.

One class classification (OCC) algorithms are receiving increasing interest both in the academia and industry [2, 3]. In some real-world applications, negative examples are hard or expensive to collect and label. Either a negative doesn't exist, or collection and label of the negative is computationally very expensive. In an example of diagnosis of a disease, positive data are easy to access (e.g., all patients who have disease) and unlabeled data are abundant (e.g., all patient), but negative data are expensive if detection tests for the disease are expensive since all patients in the

database cannot be assumed to be negative samples if they have never been tested. The second example is system intrusion data. Historical data of system intrusion cannot be used to recognise new kinds of assault. An effective security tool would be one designed to recognise assaults as they occur through the understanding and comparison of the current behaviour against nominal systems activity. Another example is the tool break detection problem. The challenge of the tool break detection problem lies in the break data is relative scarcity compared with normal cutting data since it is difficult and costly to obtain especially for new tool types and cutting tasks. In both cases, it is necessary to estimate the test example by constructing a new classifier which does not depend on such negative examples would be especially desirable.

The One-Class Support Vector Machine is a general purpose learning method designed to handle the various one-class classification problems [4, 5]. The algorithm maps the data into the feature space corresponding to the kernel, with the outliers mapped to a small region enclosing the origin, and target class instances are separated from the origin with maximum margin. For a new point x, the value $f(x)$ is determined by evaluating which side of the hyperplane it falls on, in feature space.

To turn the one class SVM algorithm into an easy-to-use black-box method for practitioners, questions about the selection of parameters (such as the width of a Gaussian kernel, and the upper bound on the fraction of training errors and the lower bound of the fraction of support vectors ν) must to be tackled [4, 5]. The method proposed in this paper investigates the use of automatic calibration of one-class SVM to find the optimal parameters. Calibration is the process of modifying the input parameters to a numerical model until the output from the model matches an observed set of data [6]. In automatic calibration, parameters are adjusted automatically according to a specified search scheme and numerical measures of the goodness-of-fit. Compared to manual calibration, automatic calibration is faster while being less dependent on individual skill and effort, and relatively easy to implement. Previous work has involved the development and application of optimization algorithms for automatic model calibration, with the proposed methodology being demonstrated on numerical model calibration applications [6].

This paper proposes a novel approach that combines particle swarm optimization (PSO) with one-class SVM, called the PSO-One-Class SVM (POCS) hybrid algorithm. The proposed methodology is demonstrated on several applications, showing that the proposed POCS method possesses better ability to find good parameters (λ and ν) using one-class SVM in some applications. Performance comparison between PSO and a basic grid search approach is then presented.

2 One-Class Support Vector Machine

Schölkopf et al. have proposed a strategy of mapping the target-class data into the feature space corresponding to the kernel and to separate them from the origin using a boundary with maximum margin [4, 5]. The algorithm is an extension of the binary support vector algorithm to the case of one-class data. As described by Manevitz and Yousef [3], it is supposed that there is a dataset drawn from an underlying probability

distribution *P*, and one needs to estimate a "simple" subset *S* of the input space such that the probability that a test point from *P* lies outside of *S* is bounded by some a prior specified as $v \in (0,1)$. The solution for this problem is obtained by estimating a function *f* which is positive on *S* and negative on the complement S^C [4, 5]. This is illustrated in Figure 1 in which target class data are labelled as +1 and outliers are labelled as -1. The origin is the only original point that is not a member of the target class, but the algorithm relaxes this constraint to return a function *f* that has the value -1 in a restricted region around the origin and +1 elsewhere.

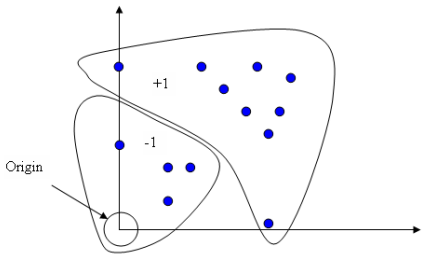

Figure 1: One-class SVM Classifier (Source: Manevitz and Yousef [3]).

To separate the data from the origin, we solve the following quadratic program:

$$\min \frac{1}{2}\|\omega\|^2 + \frac{1}{vl}\sum_{i=1}^{l}\xi_i - \rho \tag{1}$$

subject to

$$(\omega \bullet \Phi(x)) \geq \rho - \xi_i \qquad i = 1,2,\ldots,l \quad \xi_i \geq 0 \tag{2}$$

where ω and ρ are hyperplane parameters, Φ is the map from input space to feature space, v is the asymptotic fraction of outliers allowed, l is the number of training instances, and ξ is a slack variable. For solutions to this problem, ω and ρ, the decision function

$$f(x) = sign((\omega \bullet \Phi(x)) - \rho) \tag{3}$$

specifies labels for test examples, e.g., -1 for outliers.

Two commonly used kernel functions are the Gaussian Radial Basis Function (RBF) kernel $k(x_1, x_2) = \exp(-\lambda\|x - y\|^2)$ and the polynomial kernel $k(x_1, x_2) = (x \cdot y + c)^d$, where the free parameter *d* is the degree of the polynomial kernel.

3 Particle Swarm Optimisation

Kennedy and Eberhart developed particle swarm optimisation based on the analogy of swarming animals, such as a flock of birds or school of fish [7]. . In each iteration,

each agent is updated with reference to two "best" values: *pbest* is the best solution (in terms of fitness) the individual particle has achieved so far, while *gbest* is the best obtained globally so far by any particle in the population. Each agent seeks to modify its position using the current positions, the current velocities, the distance between the current position and *pbest*, and the distance between the current position and *gbest.*

The velocity of each agent is modified by the following equation:

$$v_i^{k+1} = K[v_i^k + c_1 \times rand() \times (pbest_i - s_i^k) + c_2 \times rand() \times (gbest - s_i^k)] \quad (4)$$

$$K = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|} \quad \text{where} \quad \varphi = c_1 + c_2 \qquad \varphi > 4 \quad (5)$$

A modification, the *constriction factor* approach, can generate higher quality solutions than the conventional PSO approach [8]. Here, the current position can be modified by the following equation:

$$s_i^{k+1} = s_i^k + v_i^{k+1} \quad (6)$$

Compared to genetic algorithm optimisation, there are not many parameters that need to be tuned in PSO. The parameters are: the number of particles; weighting factors; and the maximum change for a particle. It is generally found that operation is not very sensitive to parameter settings. For the number of particles, the typical range is 20 – 40 [9]. The weighting factors, c_1 and c_2, are often to 2 [9], though other settings are used in different papers, typically with $c_1 = c_2$ and in the range [0, 4] [9].

4 Automatic Calibration Scheme

The general flow chart for the calibration process using PSO is presented below and illustrated in Figure 2:

Step 1: Generation of initial condition of each agent
We begin with initial population and velocities sampled randomly from the feasible space.

Step 2: Evaluation of search point for each agent
The objective function value is calculated by running one-class SVM model.

Step 3: Modification of each searching point
The current searching point of each agent is changed using (4) and (5). If the value is better than the current *pbest* of the agent, *pbest* is replaced by the current value. If the best value of *pbest* is better than the current *gbest*, *gbest* is replaced by the best value.

Step 4: Stop.
As the standard PSO's search progresses, the entire population tends to converge towards the global optimum. This process is continued until a satisfactory condition is

met. The termination criterion is determined according to whether the maximum number of generations or a designated value of fitness is reached.

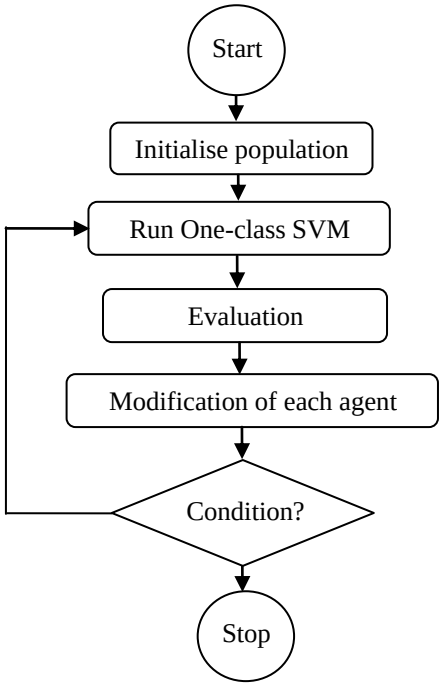

Figure 2: Outline of PSO for Optimisation Problems

5 Evaluation

5.1 Fitness Function

In order to obtain successful calibration by using automatic optimisation routines, it is necessary to formulate the calibration objective. The fitness function is formulated as follows (minimisation of fitness is assumed):

$$f(\lambda,\nu) = \frac{1}{1+PositiveRate} \tag{7}$$

$$PositiveRate = \frac{\text{Targets correctly classified}}{\text{Total targets in test set}} \tag{8}$$

For maximization problems, the fitness can be calculated as the reciprocal of the objective function value so that solutions with larger objective function value get smaller fitness.

5.2 Datasets

In order to test the validity of the proposed methodologies, the POCS method was applied to the following datasets:

Tremor Dataset: (The Tremor dataset from Exeter University, UK)[1].
For the calibration, 89 positive examples and 90 negative examples were used for training. In order to evaluate the performance of the calibrated models, test data (45 positive examples and 46 negative examples) and validation data (44 positive examples and 43 negative examples) were used.

Diabetes dataset (The Pima Indians Diabetes Database from UCI) [2] .
For the calibration, 185 positive examples and 327 negative examples were used for training. In order to evaluate the performance of the calibrated models, test data (34 positive examples and 95 negative examples) and validation data (49 positive examples and 78 negative examples) were used.

Vehicle dataset (The vehicle dataset from Statlog, to recognize a vehicle from its silhouette). [2] For the calibration, 219 positive examples and 63 negative examples were used for training. In order to evaluate the performance of the calibrated models, test data (202 positive examples and 80 negative examples) and validation data (213 positive examples and 69 negative examples) were used.

When negative examples (objects which should be rejected) are available, they can be used during the training to improve the performance [2].

5.3 Experiment Setup

In our research we used the OSU SVM (version 3.0). The OSU SVM Support Vector Machine Toolbox for MATLAB uses the LIBSVM packag[3]. The relevant experiment parameters using the PSO for one-class SVM calibration are listed in Table 1.

Table 1: Experimental Parameters

Parameter	Description	Range
λ	Kernel parameter	[0.0001 100]
ν	Fraction of outliers and support vectors	[0.01 0.3]
C_1	Weighting factor 1	2.5
C_2	Weighting factor 2	2.5
G	The total iterations	30
P	The number of particles	60

[1] The dataset is available at http://www.dcs.ex.ac.uk/studyRes.
[2] The dataset is available at http://www-it.et.tudelft.nl/~davidt/occ/index.html.
[3] OSU SVM 3.0 is available at http://svm.sourceforge.net/download.shtml.

The results, including the optimal parameters, positive rate, best and worst calibration results, average values, and the standard deviations using PSO for the objective function after 30 generations with a population size 60, are listed in Table 2. Table 2 also shows the validation result results of applying the calibrated parameter set to the validation dataset. From Table 2, it can be seen that PSO is able to find optimal calibration parameters of one-class SVM with good positive rates of the 10 random runs, and all the negative examples can be classified (negative rate =100%) in the ten random runs. The negative rate is formulated as follows:

$$NegativeRate = \frac{\text{Outliers correctly classified}}{\text{Total outliers in test set}} \tag{9}$$

The small standard deviations of fitness by the PSO imply that the method POCS is stable.

Table 2: One-Class SVM Calibration and Validation Results Using PSO

Calibration and validation results using PSO for the tremor dataset				
Trial	Optimal Parameter λ	Optimal Parameter ν	Calibration Result Positive Rate (%)	Validation Result Positive Rate (%)
Best	0.0054	0.02	100	88.64
Worst	0.1371	0.01	91.11	88.64
Mean			97.111	87.048
STD			3.4831	2.6368
Calibration and validation results using PSO for the diabetes dataset				
Trial	Optimal Parameter λ	Optimal Parameter ν	Calibration Result Positive Rate (%)	Validation Result Positive Rate (%)
Best	0.0001	0.03	100	100
Worst	0.0001	0.04	100	100
Mean			100	100
STD			0	0
Calibration and validation results using PSO for the vehicle dataset				
Trial	Optimal Parameter λ	Optimal Parameter ν	Calibration Result Positive Rate (%)	Validation Result Positive Rate (%)
Best	4.0090	0.01	99.50	99.06
Worst	0.0285	0.03	99.01	98.12
Mean			99.402	98.872
STD			0.2066	0.3963

Figure 3 shows an example that entire population converged around the global optimum after 3 generations using PSO with the population size of 60 for the tremor dataset, so a fixed number of iterations or generations (G=30) has been suggested as a stopping criterion in the calibration process for the three test datasets.

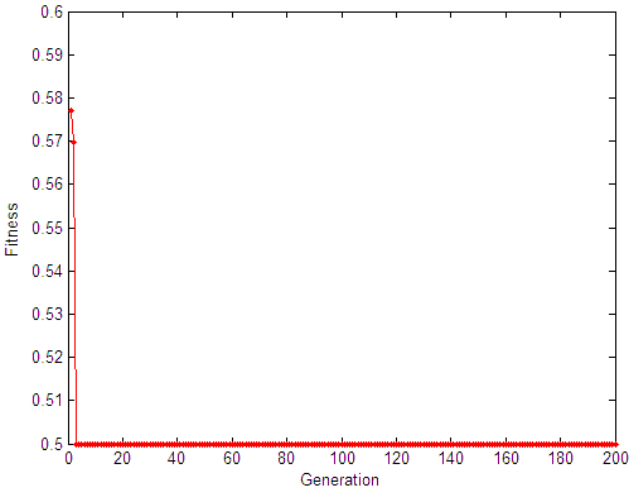

Figure 3: Iteration process using PSO

5.4 Comparison with Grid Search

For the purposes of comparison with PSO, we perform an standard grid search over the same range of parameters and with the same increments; i.e. $\lambda = 0.0001, 0.1001,...,$ 100 and $\nu = 0.01,0.03,\ldots, 0.3$. Thus, for the grid search, the number of model evaluations equal to 15×1000. The grid search method is simply an exhaustive search to determine the global optimum among those at each point on the grid of parameter values. Clearly, it is not very efficient, but is deterministic and reliable. If *n* is the number of parameters, the method employs a moving *n*-dimensional grid with spacing determined by the increment specified. The algorithm tries to centre the grid around the minimum point for each dimension (parameter), moving in an appropriate direction during each iteration. The optimization is successful when the grid becomes centered on a minimum point across all dimensions. Table 3 also shows the results of using the basic grid search to find that the best (λ, ν) and calibration and validation results.

From Tables 2 to 3, it is seen that the results of PSO and grid search are very close. Thus, it is clear that the PSO optimisation framework considered here is capable of searching more efficiently than the standard grid method on the objective function under a limited computational budget (60×30 evaluations). The above results indicate the number of function evaluations using PSO is around 88 percent less than grid search. This implies that we get a considerable advantage by using PSO.

Table 3: One-class SVM Calibration and Validation Results Using grid search

Calibration and validation results using grid search method for the tremor data				
Evaluations	Optimal	Optimal	Calibration Result	Validation Result

	Parameter λ	Parameter ν	Positive Rate (%)	Positive Rate (%)
15×1000	0.0001	0.2	100	88.64

Calibration and validation results using grid search method for the diabetes data				
Evaluations	Optimal Parameter λ	Optimal Parameter ν	Calibration Result Positive Rate (%)	Validation Result Positive Rate (%)
15×1000	0.1001	0.1	85.29	85.71

Calibration and validation results using grid search method for the vehicle dataset				
Evaluations	Optimal Parameter λ	Optimal Parameter ν	Calibration Result Positive Rate (%)	Validation Result Positive Rate (%)
15×1000	3.7001	0.01	99.06	99.50

6 Concluding Remarks

When using one-class SVM for classification problems, it is difficult to decide the width of a Gaussian kernel λ, and the upper bound on the fraction of training errors and the lower bound of the fraction of support vectors ν. To tackle this problem, an automatic calibration scheme has been formulated that considers the calibration problem in a general single objective framework. The scheme seeks to optimise the true positive rate. The hybrid method POCS was presented that can find good parameters for a one-class SVM. It has been shown that the proposed method performed more efficiently when compared with traditional grid search method. The simulation results indicated that the proposed method was able to reduce the required simulation runs to 12% of grid search while achieving comparable calibration and validation results. The results provide us with confidence that the proposed method is indeed a viable method to reduce the computation effort required in calibrating one-class SVM model.

Acknowledgments. The authors are grateful for the support of Enterprise Ireland under Project CFTD/05/222a. The second author also acknowledges the support of a Marie Curie Transfer of Knowledge Fellowship of the European Community's Sixth Framework Programme, Contract MTKD-CT-2005-029611. Both authors thank Shehroz Khan for his comments and input.

References

1. Bishop C.M.: Neural Networks for Pattern Recognition, Clarendon Press, Oxford, 1995.
2. Tax D.M.J. and Duin R.P.W.: Support vector domain description, Pattern Recognition Letters, 20 (1999), 1191-1199.
3. Manevitz L. M. and Yousef M.: One-class SVMs for Document Classification, Journal of Machine Learning Research, 2(2001), 139-154.

4. Schölkopf B., Williamson R., Smola A., Shawe-Taylor J., Platt J.: Support Vector Method for Novelty Detection. Advances in Neural Information Processing Systems. 12(2000) 582-588.
5. Schölkopf B., Williamson R., Smola A., Shawe-Taylor J., Platt J.: Estimating the Support of a High-dimensional Distribution, Neural Computation, 13 (2001) 1443-1471.
6. Liu Y., Khu S.T.:, Automatic Calibration of Numerical Models Using Fast Optimisation by Fitness Approximation. 2007 International Joint Conference on Neural Networks (IJCNN). (2007).
7. Kennedy J., and Eberhart R.: Particle Swarm Optimisation, in Proc. of the IEEE Int. Conf. on Neural Networks, (1995) 1942–1945.
8. Eberhart R., Shi Y.: Comparing inertia weights and constriction factors in particle swarm optimization. In: Proceedings of the 2000 Congress on Evolutionary Computation. Washington, DC, (2000) 84-88.
9. Hu X.H.: Particle Swarm Optimisation tutorial, www.swarmintelligence.org/tutorials.php, available online on 26th, Jun. 2007.
10. Hsu C.W, Chang C.C and Lin C, J.: A Practical Guide to Support Vector Classification, Department of Computer Science, National Taiwan University.