A Comparative Analysis of One-class Structural Risk Minimization by Support Vector Machines and Nearest Neighbor Rule

George G. Cabral and Adriano L. I. Oliveira

Department of Computing and Systems,Polytechnic School of Pernambuco, University of Pernambuco, Rua Benfica, 455, Madalena, 50.750-410, Recife-PE, Brazil{ggc,adriano}@dsc.upe.br

One-class classification is an important problem with applications in several different areas such as outlier detection and machine monitoring. In this paper we propose a novel method for one-class classification, referred to as kernel k-NNDDSRM. This is a modification of an earlier algorithm, the kNNDDSRM, which aims to make the method able to build more flexible descriptions with the use of the kernel trick. This modification does not affect the algorithm's main feature which is the significant reduction in the number of stored prototypes in comparison to NNDD. Aiming to assess the results, we carried out experiments with synthetic and real data to compare the method with the support vector data description (SVDD) method. The experimental results show that our one-class classification approach outperformed SVDD in terms of the area under the receiver operating characteristic (ROC) curve in six out of eight data sets. The results also show that the kernel kNNDDSRM remarkably outperformed kNNDDSRM.

1 Introduction

One-class classification differs from normal classification because in the training phase there are data samples from only one class available to build the model [5][9][10][11]. The term one-class classification originates from Moya [12], but also outlier detection [13], novelty detection [2] or concept learning [7] are used.

Outlier detection is the task of learning what is normal and determining when an event occurs that differs significantly from expected normal behavior. The approach that outlier detection takes is the opposite of signature detection (which can be implemented using multi-class classification). Signature detection is explicitly given information on what is novelty, and simply attempts to detect it when it happens. False alarms are rare when using signature detection because the algorithm has been programmed to know exactly what to look for to detect the known novelty conditions. However, signature detection is unable to detect *new* unknown events. Although outlier detection systems produce more false alarms than signature detection systems, they have the significant advantage that they are able to detect new, previously unknown, novelty behavior [14].

Structural risk minimization (SRM) [16] aims to find the function that for a fixed amount of data achieves the minimum of guaranteed risk. In our approach we do not search for a function that best fit the data, we try to find the more representative and smaller amount of data in the training set according with the empirical risk minimization principle (ERM). Many other approaches for multi-class classification have a similar goal. An example is a method to prune neurons from a neural network which have similar outputs given the same input aiming to reduce the complexity of the network.

In a recent paper, we proposed to implement one-class classification with the SRM principle using a nearest neighbor (NN) rule, referred to as k-NNDDSRM [4]. One of the objectives of k-NNDDSRM is to reduce the number of instances in an NNDD like one-class classifier while improving its classification performance. Analysis has shown that this new method had a lower complexity in comparison with the NNDD [15] with an improved performance in almost all data sets considered in the experiments [3, 4].

In this paper we propose a modification in the original k-NNDDSRM to make the one-class classifier able to work in a non-Euclidean space through the use of kernel operators. The novel method introduced in this paper is referred to as *kernel k-NNDDSRM.* The idea is to map the original input space into an n-dimensional hyperspace. By doing this we establish a connection between SVM classification and our NN rule. We also make a structural changing in the original algorithm by eliminating the concept of *center of mass*, proposed in [3], thereby introducing a more general form to build the data description.

To evaluate the effectiveness of our proposed method we conducted some experiments using both artificial and real-world data sets and compared it with both the SVDD [15], Support Vector Data Description, and the original k-NNDDSRM [4]. In this paper we have chosen the SVDD by its SVM nature which means we are dealing with one of the more sophisticated and powerful methods available today. Performance is assessed by calculating the receiver operating characteristics (ROC) curves and computing the AUCs (Areas Under the Curves).

Next section briefly reviews the Support Vector Data Description method for one-class classification. Section 3 details the proposed modification in the k-NNDDSRM, named kernel k-NNDDSRM. Section 4 presents the experiments and the results, including a comparison with SVDD and the original k-NNDDSRM. Finally in section 5 conclusions and suggestions for further research are presented

2 Support Vector Data Description - SVDD

Support vector machines (SVMs) comprise state of the art machine learning methods based on the principle of structural risk minimization (SRM) [16]. SVMs can be applied, for instance, for classification and regression. SVM is one of the most sophisticated nonparametric supervised classifiers available.

One-class SVM works by mapping the data onto the surface of a hyper sphere in the feature space. The goal is to maximize the margin of separation from the origin. This is equivalent to Support Vector Data Description (SVDD)[15] which finds the smallest sphere enclosing the data. As in multi-class SVMs, slack variables, denoted by $\boldsymbol{\xi}_i$, are associated to each data sample. This allows the possibility that some of the training data samples fall outside the description (i.e. are misclassified as outliers) when the minimum radius is found.

Fig. 1 shows an example in which a data description is built and 3 objects reside in the boundary of the description and one, with $\boldsymbol{\xi}_i > 0$, falls outside of the description. These 4 objects are called support vectors.

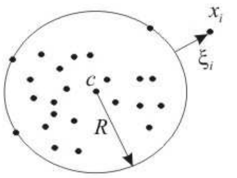

Fig. 1 Hypersphere Generated by SVDD

Let $\phi : X \rightarrow H$ be a kernel map which transforms the training sample from a space X to another space H. To separate the data from the origin with maximum margin one needs to solve the following quadratic problem:

$$\mathbf{min}\frac{1}{2}\|\omega\|^2 - \rho + \frac{1}{\nu\ell}\sum_{i=1}^{l}\boldsymbol{\xi}_i \tag{1}$$

where ω is the normal vector to the separating hyper plane, ℓ is the number of training samples and ρ is the offset, subject to $(\omega \bullet \Phi(x_i)) \geq \rho - \xi_i \quad i = 1, 2, ..., \ell \quad \xi_i \geq 0$.

If ω and ρ solve this problem, then we have found a function $f(x) = sign\left((\omega \bullet \Phi(x))) - \rho\right)$ such that if $f(x) > 0$, the object x is classified as normal. Otherwise, x is classified as novelty.

When $\rho > 0$ then the parameter $\upsilon \in (0, 1)$ is an upper bound on the fraction of outliers (i.e. training error) and also a lower bound on the fraction of support vectors. The dual problem is: $\mathbf{min}_\alpha \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j k(\Phi(x_i), \Phi(x_j))$, subject to $0 \leq \alpha_i \leq \frac{1}{\nu\ell} \quad and \quad \sum_i \alpha_i = 1$. Now the decision function is

$$f(x) = sign\left(\sum_{i=1}^{\ell} \alpha_i k(\Phi(x_i), \Phi(z)) - \rho\right) \tag{2}$$

and ρ can be recovered by

$$\rho = \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j k(\Phi(x_i), \Phi(x_j)) \tag{3}$$

where $0 \leq \alpha_i, \alpha_j \leq \frac{1}{\nu\ell}$.

To carry out simulations using SVDD in our research, we have used the DD_Tools (Data Description toolbox) version 1.6.1. This is an integrated tool for one-class classification which can handle a number of one-class classification algorithms. The DD_Tools 1.6.1 is available at http://www-ict.ewi.tudelft.nl/~davidt/dd_tools.html. In DD_Tools the parameter ν is replaced by the *fracrej* parameter, which gives the fraction of the training set which will be rejected. Therefore, in the experiments we will only refer to the parameter *fracrej*. The parameters used in our experiments will be explained in Section 4.

3 Kernel *k*-Nearest Neighbor Data Description with Structural Risk Minimization - kernel k-NNDDSRM

In this Section we first explain how the training phase of the kernel NNDDSRM is performed and then we show how the kernel *k*-NNDDSRM uses the kernel NNDDSRM to classify objects taking into account the *k* nearest neighbors.

3.1 Kernel NNDDSRM

The main feature of the NNDDSRM [3] consists of reducing the number of stored prototypes. This reduction produces at least two improvements. The first improvement is a reduction in the search time for neighbors in the classification phase. The second improvement is the reduction in memory space for data storage.

NNDDSRM is based on NNSRM [8], a classification algorithm based on NN (Nearest Neighbor) and SRM. The idea of NNSRM for the case of one-class classification is to include in the prototype set only the training samples which are in the harder region for classification. The training samples are included in the prototype set until the training error becomes zero.

The first step of the *kernel NNDDSRM* consists of computing a matrix $n \times n$, where n is the number of input patterns in the training set, with the results of the kernel function for each two input patterns. After computing the matrix we compute an array containing a sum, *Si*, of each row as shown in Eq. 4.

$$\begin{array}{rcl} \sum_{i=1}^{\ell} k(x_i, x_1) & = & s_1 \\ \sum_{i=1}^{\ell} k(x_i, x_2) & = & s_2 \\ \dots & & \dots \dots \\ \sum_{i=1}^{\ell} k(x_i, x_l) & = & s_l \end{array} \tag{4}$$

For this work we have used the RBF kernel (Eq. 5).

$$K(x_i, x_j) = \exp\left(\frac{- \parallel x_i - x_j \parallel^2}{\sigma^2}\right) \tag{5}$$

In Eq. 5, the σ value is not a crucial parameter to obtain a good kernel kNNDDSRM classifier. We have performed several experiments and varied σ; the results have shown that σ has no significant influence on performance.

After computing the array *S*, containing the s_i's (Eq. 4), it must be sorted in ascending order.

In the training phase, the kernel NNDDSRM will compute two different sets of samples, namely, the rejected set (*RS*) and the prototype set (*PS*). *RS* contains the *fracrej* patterns with smallest s_i. The idea is that a fraction of the training set (*fracrej*) should be considered outliers. On the other hand, *PS* is a set which stores prototypes that delimit the region of normal patterns. The inner training samples, that is, those with greatest sum s_i, will not be included in *PS*. The number of samples to be stored in *PS* is determined as in NNSRM, that is, training samples are included in *PS* as needed to make the training error equal to zero.

After training, we have two sets of training samples, namely, *PS* (Prototype Set) and *RS* (Rejected Set). Both sets are used in the test phase of the algorithm, therefore the total number of prototypes stored by the algorithm is the number of samples in *PS* plus the number of samples in *RS*.

The following pseudo-code shows the training phase of NNDDSRM.

```
1. Load data of the training set ($TS$)
2. Compute the array ($S$) containing all the summing of each RBF between
   each input sample and the rest of the samples
3. Sort $TS$ in increasing order, according with $S$.
4. Remove $fracrej$% of the samples from the beginning of $TS$ and add them to
   $RS$
5. Remove the two first samples in $TS$ and add to $PS$.
6. FOR ALL training pattern ($p$)
       d1 = max(K($p$,$q$) | $q \in RS$)
       d2 = max(K($p$,$j$) | $j \in PS$)
           IF (d2/d1) < 1
           errorCounter++
7. IF errorCounter > 0
```

```
        //Remove the 2 first patterns from TS, add into PS, reset errorCounter
        //and go back to (7)
    ELSE
        //End
```

The test phase, for a prototype p, is performed using the following pseudo-code:

```
r1 = max(K(p,RS))
r2 = max(K(p,PS))
if (r2/r1) < th
    return NOVELTY
else
    return NORMAL
```

3.2 Kernel k*-NNDDSRM*

The kernel k-NNDDSRM method consists solely of an extension of the kernel NNDDSRM involving the k members from PS and RS with highest kernel outputs for given test object. The kernel output of the first prototype with highest kernel output in PS is compared to the kernel output of the first prototype in RS with highest kernel output to a test object. The comparison is repeated for the next $k-1$ prototypes in PS and RS with highest kernel outputs for a given test object.

The following pseudo-code shows how the algorithm takes a decision on a pattern z to be classified:

```
1. kRS // set with the k prototypes with highest kernel outputs to z in
       //RS, increasing order
2. kPS // set with the k prototypes with highest kernel outputs to z in
       // PS, increasing order
   NOVELTIES = 0 // number of patterns classified as novelties
   NORMAL = 0 // number of patterns classified as normal
3. for ( i = 1 to i ≤ k)
       d1 = K(z,kRS[i])
       d2 = K(z,kPS[i])
           IF (d1/d2) ≤ th
               normal++;
           ELSE
               novelties++;
   end for
4. IF(novelties ≤ normal)
       //the pattern z is classified as novelty
   ELSE
       //the pattern z is classified as normal
```

4 Experiments

This section reports on experiments carried out to evaluate the performance of the kernel k-NNDDSRM method and to compare it to SVDD and k-NNDDSRM. For the experiments with the three methods we considered a range of 5% to 25% for the *fracrej* parameter. The parameter k of the kernel k-NNDDSRM and k-NNDDSRM were varied from 1 to 5 and the parameter σ of the SVDD method was varied with values [5, 10, 15, 20].

To evaluate the methods we have used the area under the curve (AUC) produced by the receiver operating characteristic curves (ROC) which is frequently used to evaluate one-class classifiers and methods for novelty detection [15], [5], [14]. In the ROC curve, the x-axis represents the PFA (Probability of False Alarm), which identifies normal patterns wrongly classified as novelties; the y-axis represents the PD (Probability of Detection), which identifies the probability that patterns of the novelty class be recognized correctly. The ROC curve depicts several operating points where each one of these operating point consist of a different classifier.

Aiming to obtain the most accurate points to build the ROC curve we have generated an array having length = #(test dataset), containing all the results values of testing the model in each sample from the test dataset. After creating the array, we sorted this array in increasing order and applied the same approach used for Tax [15] for building the ROC curve; this approach achieves the most accurate points with a low computational cost. With this approach we do not need to vary any parameter for building the ROC curves.

The experiments were conducted using six data sets, three of them from the UCI repository [1]. We have used two artificial data sets and four real world data sets in the experiments.

The first artificial data set was generated from two Gaussian Distributions and was also used in [3][4]. In the Gaussian Distributions data set the samples belonging to normal class were generated by a Gaussian distribution with mean 0 and covariance 4 and the samples belonging to novel class by one with mean 4 and covariance 4. This data set is particularly important because it is visually possible to analyze the behavior of the algorithm and to validate it.

The Banana Shaped data set, as the previous data set, is an artificial bi-dimensional data set which was also used in [15]. This data set was generated with the prtools Matlab toolbox [6].

Fig. 2 shows a small, but representative, fraction of the samples of the bi-dimensional Gaussian Distributions data set and of the Banana Shaped data set.

Three of the real-world data sets were obtained from the UCI Repository [1]: (1) Iris, (2) Winscounsin Breast Cancer and (3) Pima Indian Diabetes. The breast cancer and diabetes are two classes data sets. The Iris data set has three different classes, thus we generated three different data sets from it for novelty detection experiments. In each data set, a different class was selected to represent novelties whereas patterns from the remaining classes represented

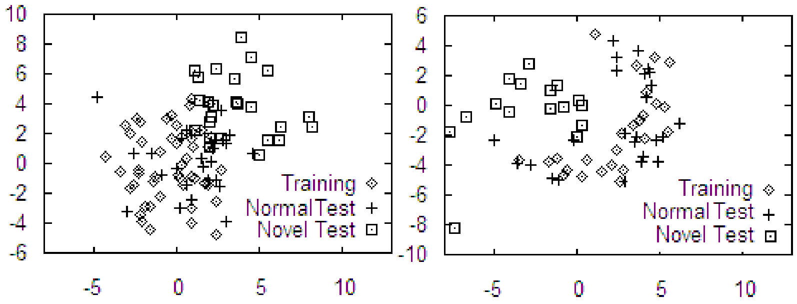

Fig. 2 Synthetic Gaussian Distributions and Banana data sets distribution

the normal class. For simplicity we labeled class Iris-setosa as 1, Iris-versicolor as 2 and Iris-virginica as 3. Thus we generated three different data sets, named *Iris class 1*, *Iris class 2*, and *Iris class 3*. The Biomed data set, available in the StatLib (http://lib.stat.cmu.edu/datasets) archive, was also used in our experiments. This data set was also used in [5]. Table 1 shows the partitioning of the data sets used in the experiments.

Table 1 Data sets Patterns Partitioning

Data set	♯Training Patterns	Test Patterns	
		♯normal	♯novelty
Gaussian Distributions	300	150	150
Banana	80	80	80
Iris class 1	50	50	50
Iris class 2	50	50	50
Iris class 3	50	50	50
Diabetes	250	250	268
Breast Cancer	184	269	239
Biomed	80	54	75

Table 2 shows the results of the comparison of both methods, kernel k-NNDDSRM and SVDD. The best AUC results are shown in boldface.

Table 2 kernel k-NNDDSRM and SVDD results

Data set	Kernel k-NNDDSRM					SVDD				
	fracrej%	k	#prot	%Total	AUC	fracrej%	σ	#SV	%Total	AUC
Gaussian Distributions	17	2	85	28.33	0.9144	8	10	226	75.33	**0.9351**
Banana	24	4	28	35	0.9309	10	5	11	13.75	**0.9864**
Iris Class 1	20	2	17	34	**1.0**	5	5	4	8	0.9800
Iris Class 2	6	1	5	10	**0.5910**	8	5	6	12	0.1296
Iris Class 3	16	3	19	38	**0.9848**	6	5	6	12	0.9736
Biomed	13	4	21	26.25	**0.9080**	5	5	80	100	0.8725
Diabetes	19	2	157	62.8	**0.7017**	7	20	167	66.8	0.6548
Breast Cancer	20	2	86	46.73	**0.9974**	7	5	61	33.15	0.7781

For both synthetic data sets the SVDD slightly outperformed our proposed method. For the synthetic Gaussian Distributions data set the best result when using kernel k-NNDDSRM was achieved using the parameter fracrej set to 17% and $k = 2$. In this case, we observed a performance loss, relative to SVDD, of 2.07%, on other hand only 28.33% of the entire training set was used for classification whereas the best SVDD used 75.33%.

Our proposed method outperformed the SVDD in all four real world data sets of Table 2. In the Iris data set, when the class 1 was elected as novelty, we have achieved the best possible result, AUC = 1. With class 2 as novelty we achieved a poor result with both methods. In the Diabetes data set, even achieving a considerably better result than the SVDD, the AUC of 0.7017 was not satisfactory. In the Biomed data set the kernel k-NNDDSRM has achieved a better AUC than the SVDD storing 73.75% less prototypes. A great performance was also achieved in the Breast Cancer data set. An AUC of 0.9974 was achieved by our proposed method storing only 46.73% of the entire training set.

Finally, we compare the performance of the kernel k-NNDDSRM with our earlier method, the k-NNDDSRM [4]. Table 3 shows the best results obtained in this paper and in [4], considering the same data sets. Once more, the boldface AUCs show the bests results. The results show that the kernel k-NNDDSRM remarkably outperformed the original k-NNDDSRM in the first three data set and obtained similar result in the last one.

Table 3 kernel k-NNDDSRM and standard k-NNDDSRM results [4]

Data set	Kernel k-NNDDSRM					k-NNDDSRM				
	fracrej%	k	#prot	%Total	AUC	fracrej%	k	#prot	%Total	AUC
Gaussian Distributions	17	2	85	28.33	**0.9144**	5	1	40	13.3	0.7640
Biomed	13	4	21	26.25	**0.9080**	15	3	21	26.25	0.8500
Diabetes	19	2	157	62.8	**0.7017**	25	9	157	62.8	0.6470
Breast Cancer	20	2	86	46.73	**0.9974**	15	3	125	50	0.9950

5 Conclusion

In this paper we proposed a novel method for one-class classification named kernel k-NNDDSRM. It is a modification of an early method that we developed, the k-NNDDSRM. The new method aims to obtain more flexible descriptions than a sphere shaped description, achieved by the original k-NNDDSRM. This was done by using the kernel trick in our method and also by eliminating the concept of *center of mass* [3, 4]. Both methods have a parameter k which makes the final result more dependent on the neighborhood [4].

The novel method was able to achieve a significant reduction in the number of stored prototypes in comparison to NNDD, which stores all training patterns.

This reduction is directly related to the parameter *fracrej* which indicates the fraction of prototypes in training set that should fall outside the description boundary.

Our simulations using real and synthetic data sets have shown that the proposed method has achieved a good performance in comparison with the SVDD method. In six out of eight data sets our method outperformed SVDD. In comparison with the original k-NNDDSRM our method has obtained much better results in all data sets.

Our future work will include the use of other kernels besides the RBF kernel. We also aim to adapt our method for training with examples of the novelty class as well as of the normal class, as in [15].

References

1. Asuncion, A., Newman, D.: UCI machine learning repository (2007). URL `www.ics.uci.edu/~mlearn/MLRepository.html`
2. Bishop, C.M.: Novelty detection and neural network validation. IEE Proceedings — Vision, Image and Signal Processing **141**(4), 217–222 (1994). Document No. 19941330
3. Cabral, G.G., Oliveira, A.L.I., Cahú, C.B.G.: A novel method for one-class classification based on the nearest neighbor rule and structural risk minimization. In: Proc. IJCNN'2007, International Joint Conference on Neural Networks, pp. 1976–1981 (2007)
4. Cabral, G.G., Oliveira, A.L.I., Cahú, C.B.G.: Combining nearest neighbor data description and structural risk minimization for one-class classification. Neural Computing & Applications (2008). Accepted for publication
5. Cao, L., Lee, H.P., Chong, W.K.: Modified support vector novelty detector using training data with outliers. Pattern Recognition Letters **24**(14), 2479–2487 (2003)
6. Duin, R.P.W., Juszczak, P., de Ridder, D., Paclík, P., Pekalska, E., Tax, D.M.J.: PR-Tools 4.0, a Matlab toolbox for pattern recognition. http://www.prtools.org (2004)
7. Hanson, S.J., new Brunswick, G.S., Kulikowski, C., Japkowicz, N.: Concept-learning in the absence of counter-examples: An autoassociation-based approach to classification. Tech. rep. (1999). URL `http://citeseer.ist.psu.edu/222433.html`
8. Karacali, B., Krim, H.: Fast minimization of structural risk by nearest neighbor rule. IEEE Trans. on Neural Networks **14**, 127–137 (2003)
9. Markou, M., Singh, S.: Novelty detection: a review - part 1: statistical approaches. Signal Processing **83**(12), 2481–2497 (2003)
10. Markou, M., Singh, S.: Novelty detection: a review-part 2: neural network based approaches. Signal Processing. Dec. 2003; 83(12): 2499-521 (2003)
11. Marsland, S., Nehmzow, U., Shapiro, J.: On-line novelty detection for autonomous mobile robots. Robotics and Autonomous Systems **51**(2-3), 191–206 (2005)
12. Moya, M.M., Koch, M.W., Hostetler, L.D.: One-class classifier networks for target recognition applications. In: Proc. WCNN'93, World Congress on Neural Networks, vol. III, pp. 797–801. INNS, Lawrence Erlbaum, Hillsdale, NJ (1993)
13. Ritter, G., Gallegos, M.T.: Outliers in statistical pattern recognition and an application to automatic chromosome classification. Pattern Recognition Letters **18**(6), 525–539 (1997)
14. Salvador, S.W.: Learning states for detecting anomalies in time series. Master's thesis, Florida Institute of Technology (2004)
15. Tax, D.M.J.: One-class classificationconcept-learning in the absence of counter-examples. Ph.D. thesis, Technische Universiteit Delft (2001)
16. Vapnik, V.: Statistical Learning Theory. Wiley (1998)