Analyse automatique de données par Support Vector Machines non supervisés

par

Vincent D'Orangeville

Thèse présentée au Département d'informatique
en vue de l'obtention du grade de docteur ès sciences (Ph. D.)

FACULTÉ DES SCIENCES
UNIVERSITÉ DE SHERBROOKE

Sherbrooke, Québec, Canada, 20 mai 2012

Library and Archives Canada

Published Heritage Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et Archives Canada

Direction du Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-89639-6

Our file Notre référence
ISBN: 978-0-494-89639-6

Le 4 juin 2012

le jury a accepté la thèse de Monsieur Vincent D'Orangeville
dans sa version finale.

Membres du jury

Professeur André Mayers
Directeur de recherche
Département d'informatique

Professeur Ernest Monga
Codirecteur de recherche
Département de mathématiques

Professeur Jean-Pierre Dussault
Évaluateur interne
Département d'informatique

Professeur Denis Larocque
Évaluateur externe
Service de l'enseignement des méthodes quantitatives de gestion
HEC Montréal

Professeur Shengrui Wang
Président rapporteur
Département d'informatique

Sommaire

Cette dissertation présente un ensemble d'algorithmes visant à en permettre un usage rapide, robuste et automatique des « *Support Vector Machines* » (SVM) non supervisés dans un contexte d'analyse de données. Les SVM non supervisés se déclinent sous deux types algorithmes prometteurs, le « *Support Vector Clustering* » (SVC) et le « *Support Vector Domain Description* » (SVDD), offrant respectivement une solution à deux problèmes importants en analyse de données, soit la recherche de groupements homogènes (« *clustering* »), ainsi que la reconnaissance d'éléments atypiques (« *novelty/abnomaly detection* ») à partir d'un ensemble de données.

Cette recherche propose des solutions concrètes à trois limitations fondamentales inhérentes à ces deux algorithmes, notamment 1) l'absence d'algorithme d'optimisation efficace permettant d'exécuter la phase d'entrainement des SVDD et SVC sur des ensembles de données volumineux dans un délai acceptable, 2) le manque d'efficacité et de robustesse des algorithmes existants de partitionnement des données pour SVC, ainsi que 3) l'absence de stratégies de sélection automatique des hyperparamètres pour SVDD et SVC contrôlant la complexité et la tolérance au bruit des modèles générés.

La résolution individuelle des trois limitations mentionnées précédemment constitue les trois axes principaux de cette thèse doctorale, chacun faisant l'objet d'un article scientifique proposant des stratégies et algorithmes permettant un usage rapide, robuste et exempt de paramètres d'entrée des SVDD et SVC sur des ensembles de données arbitraires.

Remerciements

Je tiens à remercier mes directeurs et codirecteurs de thèse, André Mayers et Ernest Monga, pour leur confiance à mon égard et la rigueur scientifique qu'il on su me transmettre au cours des dernières années.

Merci infiniment à mes parents, Annie et Christian, pour leur support indéfectible, leur compréhension à l'égard de mes absences répétées lors de mon exil à Sherbrooke, et pour toute l'aide et l'inspiration qu'ils ont su m'apporter. Je remercie également ma belle-famille Monique et Michel pour leurs petites attentions délicates et le simple fait d'avoir une fille aussi formidable.

Je tiens à remercier mon frère Loïc et ma sœur Akané pour avoir veillé à ce que je maintienne un semblant de santé mentale et de vie sociale, et à petit mon garçon, Alexandre, pour ses grands sourires qui éclairent chacune de mes journées.

Je remercie mon comité examinateur pour leurs critiques pertinentes et commentaires constructifs.

Je dédicace cette thèse à ma formidable femme, Elizabeth, qui par sa patience infinie, son éternel soutien, ses constants encouragements et son caractère unique, a su me faire garder le sourire et me garder motivé au cours des dernières années.

Table des matières

Liste des abréviations

SVM	Support Vector Machine
SVC	Support Vector Clustering
SVDD	Support Vector Domain Description
AL	Active Learning
SV	Support Vector
SMO	Sequential Minimal Optimization
F-SMO	Fast SMO
KKT	Karush-Kuhn-Tucker

Introduction

Contexte

Les « *Support Vector Machines* » (SVM) sont une classe d'algorithmes d'analyse de données dérivées des fondements théoriques sur l'apprentissage statistique formalisés par Vapnik dans son ouvrage *The Nature of Statistical Learning Theory* [5]. Les SVM se déclinent sous deux catégories d'algorithmes d'apprentissage : les algorithmes dits *supervisés*, adaptés aux contextes de classification (« *Support Vector Classifier* » - SVM) et de régression (« *Support Vector Regression* » - SVR), et ceux dits *non supervisés*, objets de cette thèse doctorale, adaptés à la détection d'éléments atypiques (« *Support Vector Data Description* » - SVDD) et à la recherche de groupements homogènes (« *Support Vector Clustering* » - SVC).

Les SVM non supervisés sont caractérisés par un processus d'induction estimant une courbe de niveau de la fonction de densité sous-jacente à un ensemble de données, englobant de façon compacte les observations les plus représentatives. Ces contours sont estimés par la méthode SVDD, en générant une hypersphère de rayon minimal renfermant une proportion contrôlée de points dans un référentiel de projection non linéaire. La projection est réalisée implicitement par l'usage de noyaux gaussiens et permet de générer, dans le référentiel des données, un ensemble de courbes de formes arbitraires dont la complexité est contrôlée par le paramètre σ définissant l'étendue du noyau gaussien, et dont la tolérance au bruit est contrôlée par le paramètre ρ définissant la proportion de points exclus des contours.

Le SVDD produit et exploite ces contours afin de différencier les instances normales des instances anormales d'une classe d'observations, et l'algorithme SVC utilise ces mêmes contours afin d'identifier des groupements homogènes d'observations (« *clusters* ») associés à des zones de densités élevées.

Les SVDD ont été utilisés avec succès dans des contextes tels que la détection de visages [10], la reconnaissance vocale [3], la détection d'ombres mouvantes en télésurveillance [11], le diagnostic de pathologies cardiaques rares [4] et l'identification de dysfonction dans les réseaux informatiques [5]. Les SVC ont été employés en segmentation de clientèle en marketing [7] et en gestion de relation à la clientèle [16], la détection de règles sémantiques [14], en groupement des courbes de charges électriques [2] et d'images rétiniennes biométriques [12], et en segmentation d'images [6].

Les SVDD et les SVC bénéficient des qualités fondamentales suivantes :

- La surface estimant le domaine jouit d'une grande flexibilité lui permettant de s'adapter à des distributions complexes. La complexité de la surface est contrôlée via un seul paramètre σ définissant l'étendue du noyau gaussien ;
- La surface bénéficie d'une tolérance explicite au bruit contrôlée par un paramètre de pénalisation ρ permettant de définir la proportion de points exclus des contours.

En contrepartie, les SVDD et SVC sont affligés des trois limitations fondamentales suivantes restreignant leur usage dans des contextes concrets d'analyse de données :

- L'absence d'algorithme d'optimisation efficace permettant d'exécuter la phase d'entrainement des SVDD et SVC sur des ensembles de données volumineux dans un délai acceptable ;
- L'absence de stratégies de sélection automatique des hyperparamètres (σ,ρ) pour SVDD et SVC contrôlant respectivement la complexité et la tolérance au bruit des modèles générés ;
- Le manque d'efficacité et de robustesse des algorithmes existants de partitionnement des données pour SVC en présence de groupements aux formes complexes.

La résolution individuelle des trois limitations mentionnées ci-haut constitue les trois axes principaux de cette thèse doctorale, chacun faisant l'objet d'un article scientifique proposant des stratégies et algorithmes permettant un usage rapide, robuste et exempt de paramètres d'entrée des SVDD et SVC sur des ensembles de données arbitraires.

Objectifs

Les trois limitations précédemment énumérées sont individuellement résolues via l'atteinte des objectifs suivants :

1. Créer un algorithme d'optimisation exécutant la phase d'entrainement des SVDD sur des données volumineuses dans un délai acceptable. L'algorithme développé doit traiter des observations séquentiellement, afin d'être compatible avec une stratégie d'apprentissage actif.

2. Développer un mécanisme d'apprentissage actif (« *active-learning* ») identifiant les candidats les plus informatifs dont l'optimisation par l'algorithme développé en (1) minimise le nombre total d'étapes d'optimisation tout en produisant une solution de qualité comparable à celle d'un modèle entrainé sur la totalité des observations.

3. Développer un algorithme pour SVC permettant un partitionnement robuste et efficace des données en groupes homogènes distincts, à partir d'une solution d'un modèle SVDD préalablement entrainé par l'algorithme développé en (1). L'algorithme proposé doit produire une segmentation exacte en présence de groupements aux formes complexes ainsi qu'en présence de données bruitées.

4. Mettre au point un mécanisme non supervisé de sélection automatique des hyperparamètres pour SVDD, résultant en une représentation robuste et compacte du domaine d'un ensemble de données bruité. La stratégie proposée doit être indépendante d'un ensemble de validation comportant des instances négatives/anormales de la classe cible.

Méthodologie

Nous avons développé « *Fast-SMO* » (F-SMO), un algorithme d'optimisation permettant d'accomplir efficacement la phase d'entrainement d'un SVM non supervisé (objectif 1) sur un flux d'observations sélectionnées par notre stratégie d'apprentissage actif (objectif 2). Cette stratégie est basée sur une mesure hybride offrant un compromis entre un critère de diversité spatiale ainsi qu'un critère d'ambiguïté, et permet de concentrer la phase d'entrainement de l'algorithme F-SMO sur un sous-ensemble d'observations les plus pertinentes.

Nous avons mis au point L-CRITICAL, un algorithme efficace de partitionnement de données (objectif 3) pour SVC, basé sur un nouveau test d'interconnexion robuste permettant un partitionnement précis et rapide des données en présence de groupements aux formes complexes. Ce test d'interconnexion est basé sur une analyse des chemins d'interconnexions entre les points critiques de la fonction $d(x)$ définissant les contours. À cet effet, un algorithme efficace de recherche des points critiques a été mis au point, jumelant un processus d'optimisation de Quasi-Newton avec un mécanisme de fusion des trajectoires similaires.

Nous avons créé une méthode de sélection automatique des hyperparamètres pour SVDD (objectif 4) dans un contexte non supervisé. La méthode intègre une mesure de surgénéralisation, permettant de rejeter les hyperparamètres résultant en une représentation trop complexe d'un ensemble de données (« *overfitting* »), et intègre à la fois une mesure robuste en présence de bruit, permettant d'identifier des représentations compactes offrant une estimation juste du domaine d'un ensemble de données quelconque.

Résultats

Tel que discuté dans l'article 1, les expérimentations révèlent que l'algorithme F-SMO permet d'exécuter la phase d'entrainement 7 fois plus rapidement que l'algorithme usuel «*Sequential Minimal Optimization* » (SMO) [9], tout en générant une solution pratiquement identique à la solution exacte. L'intégration du mécanisme d'apprentissage actif à l'algorithme F-SMO permet de résoudre en moyenne la phase d'optimisation 13 fois plus rapidement que l'algorithme SMO, tout en produisant une solution compacte composée de seulement du quart du nombre de supports vectoriels de la solution exacte. L'algorithme F-SMO couplé à la stratégie d'apprentissage actif rend conséquemment possible l'apprentissage d'ensembles de données volumineux dans un délai raisonnable sans détériorer la qualité de la solution SVDD résultante.

Les expérimentations décrites dans l'article 2, réalisées sur des ensembles de données artificiels représentant des structures complexes de groupements, mènent à deux conclusions. En premier lieu, la méthode proposée, L-CRITICAL, affiche un temps d'exécution largement plus compétitif que les méthodes compétitives [8] [1]. En second lieu, L-CRITICAL génère un partitionnement parfait sur l'ensemble des simulations réalisées, alors que les algorithmes compétitifs affichent une proportion moyenne d'erreurs de partitionnement importante sur des groupements de formes complexes.

Les résultats présentés dans l'article 3 démontrent que la méthode proposée affiche une excellente tolérance au bruit, et permet de discerner efficacement les données normales des observations atypiques. L'algorithme SVDD implémentant notre stratégie de sélection des paramètres a été comparée à l'algorithme « *abnomaly detection* » implémenté dans le logiciel SPSS Clementine 12.0. Les résultats démontrent la supériorité de la méthode proposée sur la vaste majorité des ensembles de données et démontrent son efficacité pour un usage pratique et automatique en analyse de données réelles.

Structure de la thèse

Cette thèse doctorale est structurée sous forme de trois articles proposant des solutions à chacun des objectifs précédemment énumérés.

Chapitre 1

Optimisation rapide de SVDD avec mécanisme d'apprentissage actif

Nous proposons F-SMO, un algorithme rapide permettant d'effectuer la phase d'entrainement d'un modèle SVDD sur des ensembles de données volumineux et de dimensions élevées. L'algorithme F-SMO a la particularité de pouvoir traiter séquentiellement les observations, et est par conséquent compatible avec les stratégies d'apprentissage actif. Une nouvelle méthode d'apprentissage actif est proposée, permettant d'accélérer la vitesse de convergence de l'algorithme d'optimisation tout en ne requérant qu'un nombre restreint d'observations. Cette stratégie est la première stratégie d'apprentissage actif proposée dans le contexte des SVM non supervisés. Les résultats expérimentaux confirment que la méthode d'optimisation proposée surclasse significativement l'algorithme « *Sequential Minimal Optimization* » [9] en terme de temps d'entrainement, et que l'intégration du mécanisme d'apprentissage actif décuple la vitesse d'entrainement de F-SMO, rendant possible l'entrainement d'un modèle SVDD sur des ensembles de données massifs au coût d'une erreur d'approximation fonctionnelle négligeable.

La contribution de l'auteur (V. D'Orangeville) à cet article représente 90% de la charge de travail globale liée au développement des algorithmes et de la rédaction de l'article.

Fast Optimization of Support Vector Data Description with Active Learning

V. D'Orangeville, A. Mayers, E. Monga and S. Wang

Abstract — We propose F-SMO, a fast algorithm for solving the Support Vector Domain Description (SVDD) optimization problem that implements a new active learning strategy that accelerates its learning rate by focusing only on the most informative instances of the dataset. The proposed active learning strategy integrates spatial-diversity and distance-based strategies reduce by more than 90% the training time and 70% the model complexity without affecting the solution accuracy. We investigate the computational efficiency of the F-SMO algorithm with active learning on synthetic and real-world datasets of various sizes and dimensions and show that it significantly outperforms the well-established Sequential Minimal Optimization (SMO) algorithm in terms of training time and solution complexity.

1 Introduction

SUPPORT Vector Machine (SVM) refers to a group of machine learning algorithms derived from concepts of statistical learning formalized by Vapnik in his book *The Nature of Statistical Learning Theory* [18]. The SVMs were introduced in 1995 by Cortes and Vapnik [7] as a binary classifier algorithm and then extended to the regression problem, providing exceptional generalization performance on many difficult learning tasks. While the literature reveals a high degree of interest in new efficient SVM optimization algorithms in the supervised context over the past decade, few work has been reported on unsupervised SVM counterpart. In fact, to our knowledge, there is only one adaptation of the SVM, known as Support Vector Domain Description (SVDD) [17] for unsupervised learning. Although the SVDD has been successfully applied to perform anomaly detection and cluster analysis, it is not effective on large-scale datasets.

In this paper, we aim to propose an efficient and effective method, named F-SMO, for solving the nonlinear optimization problem associated with unsupervised SVM learning for SVDD for large-scale datasets. This objective will be reached in two stages. First, we develop a fast online algorithm for SVDD. Our approach is inspired by recent advancements proposed by Bordes in 2005 [4] in the context of SVM classifiers. Bordes's online optimization algorithm for SVM classifiers was derived from the Sequential Minimal Optimization (SMO) [15]. It allows learning sequentially from individual instances as opposed to the conventional SVM algorithm, which requires the prior availability of the entire training dataset. We propose an extension of the SMO to solve SVDD by redefining the KKT optimality conditions that allow defining the selection criterion for KKT violating pairs for joint optimization and Lagrangian updating rules. The new algorithm, named Fast-SMO or F-SMO for short, allows solving efficiently the SVDD optimization problem from a stream of individual patterns.

In the second stage, we develop an active learning strategy [6] that selects individual patterns for the F-SMO algorithm. In fact, most of the individual patterns analysed by the F-SMO process do not contain significant information about the borders of clusters. At the same time, they also cause a reduction in the efficiency of the algorithm, especially if we need to deal with very large datasets. The new active learning strategy is designed for selecting the most informative instances for optimization by F-SMO while reducing significantly the number of training patterns involved for obtaining a good approximation of the SVDD exact solution. The proposed selection scheme is based on a combination of a spatial diversity and distance-based criteria. It allows F-SMO to generate an approximation of the exact solution with a very small error, while dramatically reducing the complexity of the solution and the computational time requirement by an order of magnitude compared to the LIBSVM implementation of SMO for SVDD. The proposed active learning strategy is the first selection strategy of its kind in an unsupervised SVM learning context, and allows large scale datasets to be learned within reasonable training time.

In the follows, Section 2 presents adaption of SMO to solving SVDD. Section 3 describes the new active learning strategy for SVDD. Section 4 presents experimental evaluations of the proposed algorithms on real and synthetic datasets. Note that to allow a fair comparison between LIBSVM[1] and F-SMO for SVDD optimization, we have chosen to disable all heuristics such as shrinking and kernel caching. All algorithms are implemented in C++ and are available upon request to the authors.

2 SVDD sequential optimization

The SVDD is designed to characterize the support of the unknown distribution function of an

[1] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

input dataset by computing a set of contours that rejects a controlled proportion ρ of patterns. These contours provide an estimate of a specific level set associated with the probability $1-\rho$ of the distribution function and allow unseen patterns to be classified as normal or abnormal. This section details the SVDD optimization problem, the optimal candidate selection strategy for optimization and the F-SMO algorithm for solving efficiently the SVDD optimization problem.

2.1 SVDD optimization problem

Given a set X of training vectors $x_i \in \mathbb{R}^d, i=1,\dots,n$ and a nonlinear mapping ϕ from X to some high-dimensional nonlinear feature space Φ, we seek a hypersphere of center a and minimal radius R that encloses most data points and rejects a proportion ρ of the less representative patterns. This requires the solution of the following quadratic problem:

$$\begin{gathered}\min_{R^2,\xi_i,a}\left\{R^2+C\sum_{i=1}^{n}\xi_i\right\}\\ \left\|\phi_i-a\right\|^2\le R^2+\xi_i,\\ \xi_i\ge 0,\quad i=1,\dots,n.\end{gathered}\tag{1}$$

Slack variables ξ_i are added to the constraints to allow soft boundaries, and ϕ_i denotes the coordinate $\phi(x_i)$ of x_i in the feature space. Points associated with $\xi_i>0$ are excluded from the contours and penalized by a regularization constant C which controls a proportion ρ of points lying outside (and on the surface of) the hypersphere.

The optimization problem (1) can be solved by introducing the Lagrangian L as a function of primal variables R^2, ξ_i and a and dual variables α and β referred as Lagrange multipliers enforcing the two constraints in (1).

$$\begin{gathered}L\left(R^2,\xi,a,\alpha,\beta\right)=R^2+C\sum_{i=1}^{n}\xi_i-\sum_{i=1}^{n}\alpha_i\left(R^2+\xi_i-\left\|\phi_i-a\right\|^2\right)-\sum_{i=1}^{n}\beta_i\xi_i\\ \alpha_i,\beta_i\ge 0,\quad i=1,\dots,n.\end{gathered}\tag{3}$$

Define p^* as the optimal value of the object function (1), we can verify that

$$p^* = \min_{R^2,\xi,a}\left(\max_{\alpha\geq 0,\beta\geq 0} L\left(R^2,\xi,a,\alpha,\beta\right)\right) \tag{4}$$

Moreover, we can define the dual optimal value of the dual objective function $D(\alpha,\beta) = \min_{R^2,\xi,a} L\left(R^2,\xi,a,\alpha,\beta\right)$ as

$$p^* = \max_{\alpha\geq 0,\beta\geq 0} D(\alpha,\beta) = \max_{\alpha\geq 0,\beta\geq 0}\left(\min_{R^2,\xi,a} L\left(R^2,\xi,a,\alpha,\beta\right)\right) \tag{4}$$

Setting to zero the partial derivatives of formula (3) with respect to primal variables R^2, ξ_i and a at the optimal point leads to:

$$\begin{aligned}
&\frac{\partial L_P}{\partial R^2} : 1 - \sum_{i=1}^{n}\alpha_i = 0 \rightarrow \sum_{i=1}^{n}\alpha_i = 1 \\
&\frac{\partial L_P}{\partial \xi_i} : C - \alpha_i - \beta_i = 0 \rightarrow C = \alpha_i + \beta_i \\
&\frac{\partial L_P}{\partial a} : -2\sum_{i=1}^{n}\alpha_i\phi_i + 2a\sum_{i=1}^{n}\alpha_i = 0 \rightarrow a = \sum_{i=1}^{n}\alpha_i\phi_i
\end{aligned} \tag{4}$$

We can deduce from the constraints $C = \alpha_i + \beta_i$ in (4) and $\alpha_i, \beta_i \geq 0$ in (3) that $\alpha_i \leq C$. The Karush-Kuhn-Tucker (KKT) complementary slackness conditions [REF] results in:

$$\begin{aligned}
\beta_i\xi_i &= 0 \\
\alpha_i\left(R^2 + \xi_i - \|\phi_i - a\|^2\right) &= 0
\end{aligned} \tag{5}$$

It follows from constraints (5) that the image ϕ_i of a point x_i with $\varepsilon_i > 0$ and $\alpha_i > 0$ lies outside (or on the surface) the feature-space sphere, and that a point x_i with $\varepsilon_i = 0$ and $\alpha_i = 0$ lies within the sphere. This indicates that the solution is sparse, only training vectors excluded from the decision surface with $\alpha_i > 0$ contributes to the SVDD solution. These vectors are referred to as *support vectors*.

By substituting eq. (4) into the primal Lagrangian (3) allows eliminating references to primal variables

R^2, ξ_i and a, turning the Lagrangian into the Wolfe dual form L_d where $\langle\cdot,\cdot\rangle$ is the inner product of two possibly infinite vectors.

$$\begin{aligned} L_d:\quad & \max_{\alpha_i}\left\{\sum_{i=1}^{n}\alpha_i\langle\phi_i,\phi_i\rangle-\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\langle\phi_i,\phi_j\rangle\right\} \\ & \sum_{i=1}^{n}\alpha_i=1, \\ & 0\le\alpha_i\le C,\quad i=1,\dots,n. \end{aligned} \tag{6}$$

Details of the derivation of the Lagrangian into to Wolfe dual is provided below:

$$\begin{aligned} L_P &= R^2+C\sum_{i=1}^{n}\xi_i-\sum_{i=1}^{n}\alpha_i\left(R^2+\xi_i-\|\phi_i-a\|^2\right)-\sum_{i=1}^{n}\beta_i\xi_i \\ &\rightarrow R^2+C\sum_{i=1}^{n}\xi_i-R^2\underbrace{\sum_{i=1}^{n}\alpha_i}_{1}-\sum_{i=1}^{n}\underbrace{(\alpha_i+\beta_i)}_{C}\xi_i+\sum_{i=1}^{n}\alpha_i\|\phi_i-a\|^2 \\ &\rightarrow \sum_{i=1}^{n}\alpha_i\left(\langle\phi_i,\phi_i\rangle-2\langle a,\phi_i\rangle+\langle a,a\rangle\right) \\ &\rightarrow \sum_{i=1}^{n}\alpha_i\langle\phi_i,\phi_i\rangle-2\sum_{i=1}^{n}\alpha_i\langle a,\phi_i\rangle+\underbrace{\sum_{i=1}^{n}\alpha_i}_{1}\langle a,a\rangle \\ &\rightarrow \sum_{i=1}^{n}\alpha_i\langle\phi_i,\phi_i\rangle-\langle a,a\rangle \\ &\rightarrow \sum_{i=1}^{n}\alpha_i\langle\phi_i,\phi_i\rangle-\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\langle\phi_i,\phi_j\rangle \end{aligned} \tag{7}$$

The dot product $\langle\phi_i,\phi_j\rangle$ in eq. (7) is replaced by an appropriate Mercer [REF] kernel $k(x_i,x_j)$, referred to as $k_{i,j}$ for notation simplicity, overcoming the explicit reference to ϕ_i of possible infinite dimension. The Gaussian kernel is used in this context, adjusting the complexity of the cluster contours with a single parameter σ controlling the kernel bandwidth.

$$k_{i,j}=e^{-\frac{1}{2\sigma^2}\|x_i-x_j\|^2} \tag{8}$$

The Wolfe dual is simplified by replacing the dot products $\langle \phi_i, \phi_j \rangle$ by the kernel $k_{i,j}$:

$$
\begin{aligned}
L_d: \quad & \max_{\alpha_i} \left\{ \sum_{i=1}^{n} \alpha_i k_{i,i} - \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k_{i,j} \right\} \\
& \sum_{i=1}^{n} \alpha_i = 1, \\
& 0 \leq \alpha_i \leq C, \quad i = 1, \dots, n.
\end{aligned}
\tag{9}
$$

The SVDD solution can consequently be optimized by maximizing the dual equation (9). Note that the problem remains convex since the kernel matrix K with i, j th entry $K_{ij} = k_{i,j}$ is positive definite.

As described in eq. (4), the center a of the hypersphere is described as a linear combination of the feature space vectors ϕ_i.

$$
a = \sum_i \alpha_i \phi_i \tag{10}
$$

The square distance $r^2(x_t)$ from an image ϕ_t of x_t to the sphere center a is defined as:

$$
\begin{aligned}
r^2(x_t) &= \|\phi_t - a\|^2 \\
&= \langle \phi_t, \phi_t \rangle - 2\langle a, \phi_t \rangle + \langle a, a \rangle \\
&= \langle \phi_t, \phi_t \rangle - 2\sum_{i=1}^{n} \alpha_i \langle \phi_i, \phi_t \rangle + \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j \langle \phi_i, \phi_j \rangle \\
&= k_{t,t} - 2\sum_{i=1}^{n} \alpha_i k_{i,t} + \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k_{i,j}
\end{aligned}
\tag{11}
$$

Based on eq. (11), the square radius R^2 defined in (12) of the hypersphere can be calculated as the average of distances to center of $r^2(x_u)$ and $r^2(x_v)$ of two feature-space vectors ϕ_u and ϕ_v both located the closest of the hypersphere surface and respectively outside and inside the sphere. Theses vectors are identified during the optimization phase of SVDD described in Section 2.3.

$$R^2 = \tfrac{1}{2}\left(r^2(x_u)+r^2(x_v)\right)$$
$$\text{where } \begin{cases} u \leftarrow \arg\min_s r^2(x_s) \quad s.t. \quad \alpha_s > 0 \\ v \leftarrow \arg\max_s r^2(x_s) \quad s.t. \quad \alpha_s < C \end{cases} \tag{12}$$

Eq. (11) and (12) allows defining the function $d(x_t)$ for evaluating the relative position from any image $\phi(x_t)$ to the surface of the hypersphere by comparing its distance $r^2(x_t)$ to center a to the sphere radius R^2.

$$\begin{aligned} d(x_t) &= R^2 - r^2(x_t) \\ &= \tfrac{1}{2}\left(r^2(x_u)+r^2(x_v)\right) - r^2(x_t) \\ &= 2\sum_{i=1}^{n}\alpha_i k_{i,t} - \sum_{i=1}^{n}\alpha_i k_{i,u} - \sum_{i=1}^{n}\alpha_i k_{i,v} + \tfrac{1}{2}\left(k_{u,u}+k_{v,v}\right) - k_{t,t} \\ &= 2O_t - O_s \qquad \text{where } O_j = \sum_{i=1}^{n}\alpha_i k_{i,j} \text{ and } O_s = \tfrac{1}{2}\left(O_u + O_v\right) \end{aligned} \tag{13}$$

The function $d(x_t)$ classifies a point x_t inside the contours if $d(x_t) < 0$, on its surface if $d(x_t) = 0$ and outside otherwise. The decision surface is defined as the implicit surface $\{x : d(x) = 0\}$. Note that the Gaussian kernel property $k_{t,t} = 1$ allowed simplifications to be made in eq. (13). Also, values of O_u and O_v are calculated in the optimization process described in Section 2.3.

2.2 Optimal candidate selection

We describe here the notions of KKT optimality and τ-violating pair that will be used to select candidates for joint optimization during the F-SMO learning phase, as well as a stopping criterion during its optimization process.

For a trained SVDD solution, points x_i associated with a Lagrange multiplier $0 < \alpha_i < C$ lie on the surface of the sphere described by the iso-surface $d(x_i) = 0$. Points such as $\alpha_i = C$ are excluded from the contours $(d(x_i) > 0)$ and those associated with $\alpha_i = 0$ are enclosed by the hypersphere $(d(x_i) < 0)$ and do not contribute to the description of the contours. The maximization of the Wolfe dual eq. (9) produces a sparse Lagrangian vector α, where a proportion $1-\rho$ of data points lies inside the hypersphere and only a small fraction ρ of points with $\alpha_i > 0$ and $d(x_i) \geq 0$ contributes to the definition of the hypersphere surface. Based on the Karush-Kuhn-Tucker optimality conditions [14], a SVDD solution eq. (9) is optimal if each of the following conditions are fulfilled for each point x of the training set X.

$$\begin{aligned} \alpha_i = 0 \quad &\wedge \quad d(x_i) < 0 \\ 0 < \alpha_i < C \quad &\wedge \quad d(x_i) = 0 \\ \alpha_i = C \quad &\wedge \quad d(x_i) > 0 \end{aligned} \tag{14}$$

Conversely, we can state that a point x_i violates the KKT conditions in either of the following two cases:

$$\begin{aligned} \alpha_i > 0 \quad &\wedge \quad d(x_i) < 0 \\ \alpha_i < C \quad &\wedge \quad d(x_i) > 0 \end{aligned} \tag{15}$$

The KKT violation test of formula (15) allows defining a criterion to test for simultaneous violation of the KKT conditions by a pair of points (x_i, x_j) referred to as a τ-violating pair.

$$\begin{aligned} &(\alpha_i > 0 \wedge \alpha_j < C) \quad \wedge \quad (d(x_i) - d(x_j) > \tau) \\ \equiv &(\alpha_i > 0 \wedge \alpha_j < C) \quad \wedge \quad (O(x_i) - O(x_j) < \tau) \end{aligned} \tag{16}$$

A τ-violating pair (x_i, x_j) is a pair of points with $\alpha_i > 0$ and $\alpha_j < C$ which are respectively misclassified by the decision function $d(x)$ as inside and outside of the hypersphere, within a tolerance factor of τ. The absence of any such pair in the training set indicates the convergence of F-SMO and τ-optimality of the solution within a tolerance factor τ.

The F-SMO implements an efficient selection scheme inspired from Keerthi's improved selection strategy [11] for SVM classifiers and optimizes successively τ-violating pairs of formula (16) that locally maximize the gradient of the objective function (9) and induce a maximal step in the objective function's value at each iteration.

The gradient of the objective function (9) is maximized by selecting a τ-violating pair (x_i,x_j) for joint optimization according to $\max_{i,j}\left|O_i - O_j\right|$. The optimal selection strategy for the τ-violating pair in the SVDD context can be stated as follows:

$$\left.\begin{array}{lll} i \leftarrow \arg\min_k O_k & s.t. & \alpha_k < C \\ j \leftarrow \arg\max_k O_k & s.t. & \alpha_k > 0 \end{array}\right\} \text{ with } \max_{i,j}\left|O_i - O_j\right| > \tau \qquad (17)$$

The selection of a τ-violating pairs is achieved by maintaining a cache of O for all active support vectors and by keeping track of $O_{\min}$, $O_{\max}$, $\alpha_{\min}$ and $\alpha_{\max}$ during the optimization process in order to allow an immediate identification of the optimal τ-violating pair according to formula (17).

2.3 Fast sequential optimization

This section describes the algorithm F-SMO, inspired from the algorithm proposed by Bordes [4] for SVM classification. The F-SMO algorithm offers two important advantages over SMO. First, F-SMO allows the sequential processing of individual training examples, as opposed to the SMO algorithm, which treat patterns in pair and cannot treat them separately. This property is essential, as it makes it possible to implement our active learning strategy for selecting the most informative individual training patterns for optimization by the **INSERT** function of the F-SMO algorithm described below. Moreover, F-SMO allows solving the SVDD optimization problem in a single sequential pass over all patterns of the training set, while the SMO requires multiple passes over the dataset. The F-SMO method works by alternating the two following steps.

The first step, **INSERT** (x_{new}), reads an unseen input pattern x_{new} and seeks an existing support vector to form a τ -violating pair $(x_{new}, x_{\max})$ according to the optimal selection strategy eq. (14). It then performs a joint optimization of the pair by updating both multipliers $(\alpha_{new}, \alpha_{\max})$ as stated in the F-SMO subroutines in Table 1.The second step, **UPDATE**, aims at minimizing the imbalance produced by the recent inclusion of α_{new} and update of $\alpha_{\max}$ in the solution, by performing a single optimization step on a τ -violating pair selected according to [14]. It then proceeds to a pass to remove all inactive support vectors $(\alpha_i = 0)$ fulfilling the KKT optimality condition$(O_i + \tau > O_{\max})$. The purpose of this removal pass is to enforce sparseness in the solution during the optimization process by removing inactive SVs. These two steps are repeated in alternation until all points x_i of the training set X have been evaluated once by the function **INSERT** or until no more candidates are selected by the active learning strategy. A finalizing pass is then performed by iterating the function **UPDATE** over the set of active support vectors, until no more τ -violating pair can be identified indicating the convergence of the SVDD solution.

The sequence of **INSERT** and **UPDATE** in step 4 in Algorithm 1 can be considered as a filtering pass over the training set that identifies and optimizes potential support vectors within a single pass through the training set, while step 6 ensures the stabilization of the solution over the selected set of support vectors. It is worth mentioning that the single pass over the training set could fail to select important support vectors during the learning phase on very small datasets. However, the likelihood of this is minimal as the F-SMO algorithm is designed for solving large-scale datasets. Segments of code highlighted in blue in Algorithm 1 and Table 1 represent instruction sequences that benefit from multithreaded implementation.

Algorithm 1 - F-SMO.

```
1. Input parameters
    • $X \subset \mathbb{R}^d$ : input dataset of size $N$ and dimension $d$
    • $\gamma$: RBF bandwidth $\left(\gamma \in \mathbb{R}^+\right)$
    • $\rho$: rejection rate $\left(\rho \in [0,1]\right)$
    • $\tau$: KKT tolerance factor $\left(\tau \approx 0.001\right)$
2. Initialization:
    $C = \frac{1}{\rho' N}$ with $\rho' = \min\left(\frac{N-1}{N}, \max\left(\frac{1}{N}, \rho\right)\right)$
    $n_{sv} = \frac{1}{C}$
    $\vec{\alpha} = \left\{\alpha_0 = \cdots = \alpha_{n_{sv}-1} = C, \quad \alpha_{n_{sv}} = \cdots = \alpha_{N-1} = 0\right\}$
    $\vec{O} = \left\{O_0, \cdots, O_{n_{sv}-1}\right\}$ with $O_i = \sum_{j=1\cdots n_{sv}} \alpha_j k_{i,j}$
3. Selection:
    Select an unseen training example $x_t \in X$
    go to (5) if no unseen pattern remains.
4. Optimization:
    a. INSERT $(x_t)$
    b. UPDATE( )
5. Return to (3).
6. Finish:
    Repeat UPDATE( ) until $\tau$-convergence.
```

Table 1 - F-SMO subroutines.

```
INSERT $(x_{new})$
    1. Initialization
        Set $\alpha_{new} = 0$
        and compute $O_{new} = \sum_{i=1..n_{sv}} \alpha_i k_{i,new}$
    2. Check $\tau$ -optimality of $x_{new}$
        Exit if $O_{new} + \tau > O_{\max}$
    3. Optimize Pair$(x_{new}, x_{\max})$
        $n_{sv} \leftarrow n_{sv} + 1$
```

```
UPDATE ( )
    1. Check $\tau$ -optimality of solution
        Exit if $O_{\min} + \tau > O_{\max}$
    2. Optimize Pair$(x_{\min}, x_{\max})$
    3. Inactive SV removal
        Remove any $x_i$ such as $\alpha_i = 0$ and
        $O_i + \tau > O_{\max}$
    4. Update MinMax( )
    5. $R^2 = (O_{\min} + O_{\max})/2$
```

```
Optimize Pair$(x_s, x_t)$
    1. Joint optimization
        $\Delta_\alpha \leftarrow \min\left\{\frac{O_s - O_t}{k_{s,s} + k_{t,t} - 2k_{s,t}}, C - \alpha_s, -\alpha_t\right\}$
        $\alpha_t \leftarrow -\Delta_\alpha \qquad \alpha_s \leftarrow \alpha_s + \Delta_\alpha$
    2. Update O (for all active SVs)
        a) $O_i \leftarrow O_i - \Delta_\alpha (k_{i,s} - k_{i,t}) \quad \forall i \in \{1 \ldots n_{sv}\}$
        b) Update Min Max( )
```

```
Update MinMax( )
    $i \leftarrow \min_{k \in [1, n_{sv}]} O_k \quad s.t. \quad \alpha_k < C$
    $j \leftarrow \max_{k \in [1, n_{sv}]} O_k \quad s.t. \quad \alpha_k > 0$
    $O_{\min} = O_i, O_{\max} = O_j, x_{\min} = x_i, x_{\max} = x_j$
```

3 Active learning for SVDD

Active learning is the process of actively selecting the most informative patterns during the learning phase according to a sample selection criterion that accelerates the learning rate and minimizes the number of training examples required to achieve a good solution approximation. Active learning has been successfully implemented in the context of classification to enhance the learning rate of neural networks [1], support vector classifiers [8][10] [12][18] and statistical models [5][6][16].

Despite its strong theoretical foundations and encouraging results in a classification context, no active learning strategy adapted to unsupervised SVM has yet appeared in the literature, for accelerating the learning phases of SVDD. For this purpose, we propose a new active learning strategy intended to concentrate the learning phase of F-SMO on a small set of the most informative patterns, in order to improve its learning rate and reduce its solution complexity at the cost of a minimal loss of functional accuracy (compared to a full model trained on the whole training set). The proposed method is a hybrid sampling method which combines a *spatial diversity* and *distance-based* criteria to guide the selection of new candidates x_{new} within small subsets of potential learning candidates X_{AL}, to be optimized by the function **INSERT**(x_{new}) of F-SMO. This sequence of active learning selection and optimization is repeated until every training pattern has been evaluated once by the active learning selection procedure.

3.1 Spatial diversity

The *spatial diversity* criterion enforces the selection of candidates dissimilar to the current support vectors set X_{sv}, in order to minimize redundancy among support vectors and focus on the most informative candidates. The diversity fitness score $S_{div}(x_t)$ of a potential candidate $x_t \in X_{AL}$ is assessed as the minimal dissimilarity from x_t to any support vector $x_j \in X_{SV}$.

$$S_{div}(x_t) = \min_{x_j \in X_{sv}} \left(1 - k_{t,j}\right) \tag{18}$$

According to the *spatial diversity* criterion, the best candidate $x_{div}^{*} \in X_{AL}$ is the one which maximizes the ***minimal*** distance to any support vector of the expansion set X_{sv}.

$$\begin{aligned} x_{div}^{*} &= \arg\max_{x_t \in X_{AL}} S_{div}\left(x_t\right) \\ &= \arg\max_{x_t \in X_{AL}} \left[\min_{x_j \in X_{sv}} \left(1 - k_{t,j}\right) \right] \end{aligned} \tag{19}$$

This strategy is analogous to the *angle diversity strategy* tested in SVM classification [18], where the authors considers the maximal angle between the induced hyperplane $h(x_t)$ of a candidate $x_t \in X_{AL}$ in feature space and each hyperplane $h(x_j)$ associated with each support vector $x_j \in X_{sv}$. The function $h(x_i)$ defines a hyperplane passing through the image ϕ_t of x_t in feature space and the center a of the hypersphere. The angle diversity fitness score $S_{ang}(x_t)$ of a candidate x_t is evaluated as the minimal angle between $h(x_t)$ for $x_t \in X_{AL}$ and any $h(x_j)$ for $x_j \in X_{sv}$.

$$\begin{aligned} & S_{ang}\left(x_t\right) = \min_{x_j \in X_{sv}} \left|\cos\left(\angle\left(h(x_t), h(x_j)\right)\right)\right| \\ & \text{where} \quad \left|\cos\left(\angle\left(h(x_t), h(x_j)\right)\right)\right| = \frac{\left|\phi_t \cdot \phi_j\right|}{\left\|\phi_t\right\| \left\|\phi_j\right\|} = \frac{\left|k_{t,j}\right|}{\sqrt{k_{t,t} k_{j,j}}} = \left|k_{t,j}\right| = k_{t,j} \end{aligned} \tag{20}$$

According to the angle diversity fitness score of eq. (20), a candidate is chosen according to:

$$\begin{aligned} x_{ang}^{*} &= \arg\min_{x_t \in X_{AL}} S_{ang}\left(x_t\right) \\ &= \arg\min_{x_t \in X_{AL}} \left[\min_{x_j \in X_{sv}} \left(k_{t,j}\right) \right] \end{aligned} \tag{21}$$

In the context of SVDD, the angle diversity strategy enforces a uniform coverage of the hypersphere surface with support vectors images ϕ_j, and is equivalent at encouraging spatial diversity in the primal space among support vectors.

$$\begin{aligned} x^*_{div} &= \underset{x_t \in X_{AL}}{\arg\max}\left[\min_{x_j \in X_{sv}}\left(1 - k_{t,j}\right)\right] \\ &= \underset{x_t \in X_{AL}}{\arg\min}\left[\min_{x_j \in X_{sv}}\left(k_{t,j}\right)\right] \\ &= x^*_{ang} \end{aligned} \tag{22}$$

3.2 Distance-based strategy

In the SVM classification context, the *distance-based strategy* aims at selecting the misclassified candidates located the nearest of the separating plane, which corresponds at choosing ambiguous candidates in order to fine-tune the separating plane. This strategy translates in the SVDD context into focusing on the most ambiguous training patterns located immediately outside the hypersphere, or equivalently, finding the closest candidate to the contours which is excluded from the contours. Recall that only data points located outside the cluster surface contribute to the definition of the contours described by the isosurface of the decision function $d(x)$ of eq. (13).

The *distance-based* fitness score of a potential candidate x_t is calculated as:

$$S_{dist}\left(x_t\right) = d'\left(x_t\right) \tag{23}$$

where $d'\left(x_t\right)$ is the relative position of the surface of hypersphere ϕ_t, normalized by the value O_s in order to constrain its range between 0 and 1, and defined as follows:

$$d'\left(x_t\right) = \tfrac{1}{O_s} d\left(x_t\right) = 1 - \tfrac{1}{O_s} \sum_{j=1..N} \alpha_j k_{t,j} \tag{24}$$

The normalization of $d(x_t)$ is intended to control the magnitude of $S_{dist}(x_t)$. A training example $x^*_{dist} \in X_{AL}$ is then selected according to:

$$x^*_{dist} = \arg\min_{x_t \in X_{AL}} \left[1 - \tfrac{1}{O_s} \sum_{j=1..N} \alpha_j k_{t,j} \right] \tag{25}$$

A more naïve approach would favor selecting an input pattern $x^*_{naive} \in X_{AL}$ located the farthest away from the contours as $x^*_{naive} = \arg\max_{x_t \in X_{AL}} d'(x_t)$, resulting in a model more sensitive to outliers.

3.3 Hybrid selection criteria

We propose a hybrid active learning selection strategy which combines the *spatial diversity* score (18) and the *distance-based* score (23) seeking a candidate x_t excluded from the contours which simultaneously has simultaneously a minimal (positive) distance $d'(x_t) > 0$ to the hypersphere surface and maximal distance to all existing support vectors.

The existing hybrid selection strategies for SVM classification described in [10] and [18] combine these two selection criteria by defining the following convex combination:

$$\begin{aligned} &S_{convex}(x_t) = w \cdot S_{div}(x_t) + (1-w) \cdot \frac{1}{S_{dist}(x_t)} \quad \text{with } w \in [0,1] \\ &x^*_{convex} = \arg\max_{x_t \in X_{AL}} S_{convex}(x_t) \end{aligned} \tag{26}$$

One major drawback of this convex combination of fitness scores stems from the fact that the efficiency of a linear combination of fitness scores depends on the appropriate choice of the weighting parameter w which is data dependant and depends on the relative values of $S_{div}(x_t)$ to $1/S_{dist}(x_t)$. To avoid the unintuitive choice of w, we defined a hybrid score $S_{hybrid}(x_t)$ (27) computed as the ratio of the two fitness scores, allowing simultaneous maximization of the diversity score $S_{div}(x_t)$ as numerator and minimization of distance score $S_{dist}(x_t)$ as denominator.

$$S_{hybrid}\left(x_t\right)=\frac{S_{div}\left(x_t\right)}{S_{dist}\left(x_t\right)} \tag{27}$$

Combining the two selection criteria, the hybrid selection strategy selects training points $x_{al}^{*} \in X_{AL}$ according to the following criterion:

$$x_{al}^{*}=\underset{x_t \in X_{AL}}{\arg\max}\left(\mathrm{I}\left(d'(x_t)>0\right)\frac{\min\limits_{j\in X_{sv}}\left(1-k_{t,j}\right)}{1-\frac{1}{O_t}\sum\limits_{j\in X_{sv}}\alpha_j k_{t,j}}\right) \tag{28}$$

The indicator function $\mathrm{I}\left(d'(x_t)>0\right)$ returns a value of 1 if $d'\left(x_t\right)>0$ and 0 otherwise, and enforces the selection of a candidate x_{al}^{*} excluded from the sphere. A candidate x_{al}^{*} is selected according to eq. (28) from a small subset of potential candidates X_{AL} (20 candidates in our implementation), then optimized by the F-SMO procedure **INSERT**$\left(x_{al}^{*}\right)$. The selection and optimization sequence is repeated until each training pattern has been evaluated once.

4 Experiments and results

Experiments have been performed on synthetic and real-world datasets in order to compare the computational efficiency of the F-SMO optimization method with and without active learning, to the standard LIBSVM SMO algorithm, for solving the SVDD training phase. All algorithms are evaluated on 11 well-known UCI benchmark datasets with dimensions ranging from 2 to 60, in order to compare their respective training times, numbers of support vectors of the solutions and functional approximation errors ξ_{fn}, in comparison to a reference exact solution Θ_{ref}. The reference model (referred to as REF) is generated by training a SVDD model with SMO using a highly restrictive KKT tolerance factor of $\tau=10^{-7}$.

The functional approximation error ξ_{fn} is assessed by training a model with a looser factor of $\tau = 10^{-4}$ on the same training set as Θ_{ref}, and then evaluating the proportion of points misclassified by the "approximate" SVDD solution Θ_{approx} as:

$$\xi_{fn}\left(\Theta_{ref};\Theta_{approx}\right) = \frac{1}{N}\sum_{i=1..N} I\left(d^{ref}\left(x_i\right)\cdot d^{approx}\left(x_i\right)<0\right) \tag{29}$$

The procedure $\mathrm{I}(y)$ is the indicator function returning a value of 1 for any negative value of y and 0 otherwise. The function (29) evaluates the proportion of points that are (mis)classified by the approximate model (SMO or F-SMO) to the opposite side of the hypersphere as compared with the reference model.

Two variants of the proposed active learning scheme were tested. F-AL1 refers to the F-SMO method with active learning trained with rejection rate ρ. F-AL2 is trained with an adjusted $\rho_{AL} \geq \rho$ to compensate for a phenomenon involving the expansion of the generated contours obtained with active learning, in comparison to SVDD contours obtained without active learning. Values of ρ_{obs} displayed in Table 2 measure the observed proportion of points excluded from the contours generated by the models F-AL1 and F-AL2, the expected values of ρ_{obs} are $\rho = 20\%$. All the experiments reported in Table 2 are performed on a 3.6 GHz Intel quad-core CPU, with each test repeated 20 times and the results averaged. Note that the symbols +1 and -1 identify positive and negative class instances of each dataset.

Table 2 - Comparison of SMO, F-SMO, F-AL1 and F-AL2: training times, number of support vectors, functional approximation error ξ_{fn} in comparison to a reference solution and proportion of points ρ_{obs} rejected by the trained contours. Rejection rates are set to $\rho = 20\%$ (for REF, SMO, F-SMO and F-AL1) and to ρ_{AL2} for F-AL2.

					Training time(s)				Number of SVs			ξ_{fn}			ρ_{obs}	
DATASET	N	D	σ^2	ρ_{AL2}	SMO	F-SMO	F-AL1	F-AL2	REF	F-AL1	F-AL2	F-SMO	F-AL1	F-AL2	F-AL1	F-AL2
Banana (+1)	2376	2	0.75	0.304	0.125	0.023	0.011	0.010	482	104.4	151.2	0.90%	11.42%	5.24%	11.74%	22.10%
banana (-1)	2924	2	0.25	0.31	0.108	0.033	0.016	0.017	585	144.9	184.4	0.70%	13.90%	5.17%	13.40%	20.81%
image (+1)	1188	18	7.5	0.32	0.057	0.010	0.005	0.005	247	66.0	83.7	0.76%	10.60%	5.67%	9.62%	22.78%
ringnorm (+1)	3664	20	13.75	0.236	0.474	0.059	0.028	0.034	751	167.0	198.6	0.42%	6.03%	2.40%	14.04%	20.45%
ringnorm (-1)	3736	20	13.75	0.23	0.512	0.064	0.031	0.035	765	173.6	192.0	0.44%	5.18%	2.41%	15.03%	19.25%
splice (+1)	[illegible]	60	44.5	0.2	0.092	0.013	0.009	0.009	280	76.6	76.5	0.94%	3.46%	3.47%	20.01%	20.02%
splice (-1)	1647	60	44.65	0.2	0.139	0.018	0.013	0.013	352	98.7	98.7	0.96%	5.78%	5.81%	24.31%	24.34%
twonorm (+1)	[illegible]	20	13.75	0.256	0.486	0.062	0.028	0.033	759	180.2	181.4	0.30%	4.57%	1.92%	15.44%	19.52%
twonorm (-1)	3697	20	13.75	0.25	0.495	0.063	0.029	0.034	752	170.0	204.0	0.27%	6.98%	2.44%	13.01%	21.63%
waveform (+1)	1647	21	14.25	0.22	0.124	0.016	0.010	0.010	344	84.6	98.7	0.50%	3.60%	4.40%	16.40%	21.77%
waveform (-1)	3353	21	11.5	0.25	0.505	0.059	0.029	0.030	689	153.4	188.2	0.46%	5.78%	4.31%	14.26%	23.14%
			AVERAGE		0.291	0.038	0.019	0.021	547.2	128.0	153.3	0.65%	7.16%	3.89%	15.30%	21.42%

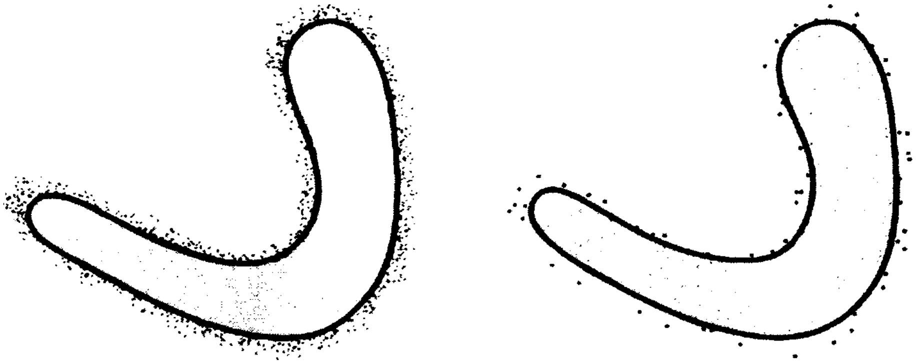
As shown in Table 2, F-AL1 and F-AL2 significantly outperform both SMO and F-SMO in terms of average training times, at the cost of increased functional approximation error ξ_{fn} over SMO and F-SMO. The models F-AL1 and F-AL2 exhibit averaged observed rejection rates ρ_{obs} of 15.39% and 21.42%, respectively, which suggests that a SVDD model trained with active learning requires an adjusted ρ_{AL2} in order to minimize the absolute difference between ρ_{obs} and ρ.

The increased functional approximation error ξ_{fn} of F-AL1 is caused by the choice of the regularization factor $C = 1/(\rho \cdot N)$ with $\rho = 20\%$ kept constant for both F-SMO and F-AL1. Because the active learning selection strategy enforces the selection of candidates located near the outer part of the hypersphere, it alters the distribution of training patterns optimized by the **INSERT**(x) procedure and results in contours of slightly expanded shapes compared to the F-SMO trained on the whole training set for the same value of ρ.

Figures 3, 4 and 5 summarizes the relative running times, numbers of support vectors, and functional approximation errors ξ_{fn} of the algorithms discussed in this paper. Based on the experimental results, F-SMO is far more competitive than SMO for optimizing a SVDD solution (not using an active learning strategy), and F-AL2 is superior to F-AL1 in terms of functional accuracy.

As illustrated in Table 2, the functional approximation error ξ_{fn} can be minimized efficiently by setting an increased value of $\rho_{AL2} \geq \rho$ to compensate for the contour expansion, also reducing ξ_{fn} for F-AL2 compared to F-AL1 in all tests performed.

Figures 1 and 2 illustrate two SVDD models trained with F-SMO and F-AL2 on a same training set of 5,000 points, F-SMO trained with $\rho = 20\%$ and F-AL2 trained with equal kernel bandwidth while increasing ρ to $\rho_{AL2} = 27.5\%$. The two methods produce comparable contours shapes. The theoretical arguments guiding the optimal adjustment of ρ_{AL2} in F-AL2 in order to minimize the functional approximation error ξ_{fn} remain to be explored in further research.

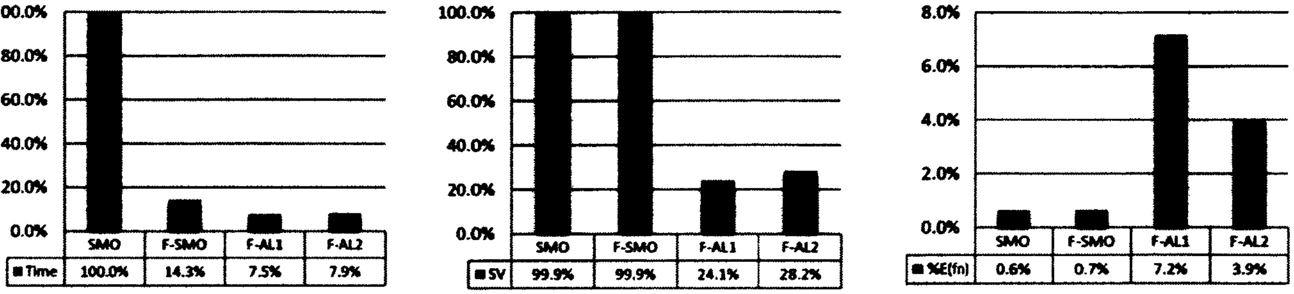


Figures 1 and 2 (from left to right) - (Figure 1) F-SMO trained with $\rho = 20\%$. (Figure 2) F-AL2 trained with $\rho_{AL2} = 27.5\%$ and $\xi_{fn} = 4.95\%$.

Table 3 - Comparison of F-SMO, F-AL1 and F-AL2 versus SMO: relative training times and number of support vectors compared to SMO.

	Training time (s)			Number of SVs	
DATASET	F-SMO	F-AL1	F-AL2	F-AL1	F-AL2
banana (+1)	18.00%	8.73%	7.96%	21.66%	31.37%
banana (-1)	17.56%	8.27%	9.15%	24.36%	32.68%
image (+1)	17.45%	8.74%	9.27%	26.70%	33.87%
ringnorm (+1)	12.41%	6.01%	7.11%	22.24%	25.78%
ringnorm (-1)	12.58%	5.98%	6.76%	22.70%	25.10%
splice (+1)	14.20%	10.12%	9.79%	27.06%	27.02%
splice (-1)	13.09%	9.48%	9.65%	28.05%	28.04%
twonorm (+1)	12.76%	6.06%	6.79%	22.30%	25.22%
twonorm (-1)	12.80%	5.81%	6.83%	22.61%	27.12%
waveform (+1)	14.16%	8.12%	8.09%	24.99%	[illegible]
waveform (-1)	11.69%	5.65%	5.98%	22.26%	27.31%
AVERAGE	14.25%	7.54%	[illegible]	24.05%	[illegible]

Table 3 reports the relative training times of F-SMO, F-AL1 and F-AL2 in reference to SMO (with $\tau = 10^{-4}$), computed from values in Table 2. F-SMO's computing time represents 14.25% of the time required by SMO to optimize the solution, and F-AL1 and F-AL2 benefit from reduced average training times and numbers of support vectors in comparison to F-SMO. The reduced number of support vectors is responsible for the increase in functional approximation error ξ_{fn}, while significantly reducing at the same time the complexity of the SVDD solution, which in turn allows new data points to be classified far more rapidly.



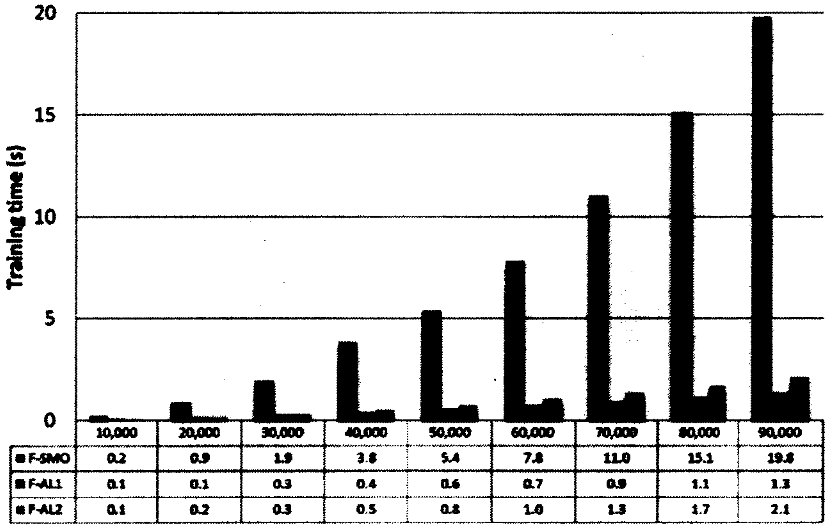
	SMO	F-SMO	F-AL1	F-AL2
■ Time	100.0%	14.3%	7.5%	7.9%

	SMO	F-SMO	F-AL1	F-AL2
■ SV	99.9%	99.9%	24.1%	28.2%

	SMO	F-SMO	F-AL1	F-AL2
■ %E(fn)	0.6%	0.7%	7.2%	3.9%

Figures 3,4 and 5 (from left to right) - Comparison of SMO, F-SMO, F-AL1 and F-AL2. (Figure 3) Relative training time compared to SMO with $\tau = 10^{-4}$, (Figure 4) Relative number of SVs compared to the exact solution, (Figure 5) Functional approximation error ξ_{fn}.

The hybrid selection strategy was evaluated on synthetic 2D datasets of sizes ranging from 10,000 to 90,000 training patterns, in order to assess the asymptotic behavior of the training times of the F-SMO algorithm implementing an active learning strategy (F- AL1 and F-AL2) relative to training set size.



	10,000	20,000	30,000	40,000	50,000	60,000	70,000	80,000	90,000
■ F-SMO	0.2	0.9	1.9	3.8	5.4	7.8	11.0	15.1	19.8
■ F-AL1	0.1	0.1	0.3	0.4	0.6	0.7	0.9	1.1	1.3
■ F-AL2	0.1	0.2	0.3	0.5	0.8	1.0	1.3	1.7	2.1

Figure 6 - Comparison of training times (s) for F-SMO, F-AL1 and F-AL2, as a function of the training set size (horizontal axis).

Table 4 - Training times, number of support vectors and functional approximation errors ξ_{fn} of F-SMO $(\rho = 10\%)$, F-AL1 $(\rho = 10\%)$ and F-AL2 $(\rho_{AL2} = 23\%)$ for increasing training set size (N) with fixed kernel bandwidth $\sigma = 0.05$.

	Time (s)			SVs			ξ_{fn}	
N	F-SMO	F-AL2	F-AL1	F-SMO	F-AL1	F-AL2	F-AL1	F-AL2
10,000	0.23	0.07	0.06	1,010	64	124	8.41%	3.28%
20,000	0.87	0.15	0.17	2,007	117	239	8.64%	1.89%
30,000	1.92	0.27	0.31	3,008	169	355	8.76%	1.76%
40,000	3.83	0.40	0.50	4,007	221	470	8.78%	1.45%
50,000	5.38	0.55	0.75	5,007	272	584	8.80%	1.31%
60,000	7.82	0.72	1.02	6,006	324	700	8.82%	1.23%
70,000	11.02	0.91	1.35	7,005	373	814	8.77%	1.13%
80,000	15.12	1.11	1.72	8,006	423	929	8.83%	1.07%
90,000	19.81	1.34	2.14	9,006	475	1,044	8.83%	1.11%

As expected, the proposed hybrid selection schemes (F-AL1 and F-AL2) show dramatically improved training times compared with the F-SMO algorithm: indeed, the asymptotic relationship of their training times to training set size is almost linear ($R^2 = 0.9821$ for AL1 and $R^2 = 0.9632$ for AL2).

Note that F-SMO with active learning can be effectively used in an online context on a continuous flow of training points, by dynamically adapting the number of candidates $|X_{AL}|$ evaluated in each active learning pass according to the availability of processing power and the speed of data acquisition.

5 Conclusion

We have proposed F-SMO, an efficient algorithm for SVDD that optimizes a stream of individual patterns during its learning phase. The development of F-SMO requires defining the KKT optimality conditions, the selection criterion for KKT-violating pairs for joint optimization and the Lagrangian updating rules in the unsupervised SVM context.

We have proposed a new active learning strategy that identifies the most informative instances for optimization by F-SMO, and reduces the overall number of training patterns required to obtain a good approximation of the SVDD solution. The hybrid candidate fitness measure is based on diversity and ambiguity criteria that allow F-SMO to generate an approximation of the exact solution with small approximation error, while dramatically reducing the complexity of the solution and the computational burden – by more than 10 times compared to SMO. The proposed active learning strategy is the first selection strategy adapted to SVDD learning, and makes it possible to optimize large-scale datasets within reasonable training time.

We have compared the effectiveness of the proposed method F-SMO with active learning to the standard LIBSVM SMO implementation on several synthetic and real-world datasets. Experiments suggest that F-SMO solves the same problem in less than 15% of the time spent by SMO and that F-SMO with active-learning in less than 8%, proving their vast superiority in terms of computational cost on all experiments performed.

References

[1] A. Adejumo, A. Engelbrecht, A Comparative study of neural networks active learning algorithms, Proceedings of the International Conference on Artificial Intelligence. (1999) 32- 35.

[2] A. Ben-Hur, D. Horn, H.T. Siegelmann, V. Vapnik, A support vector clustering method, Proceedings 15th International Conference on Pattern Recognition. 2 (2000) 724-727.

[3] Benchmark repository. [Online] [Cited: 11 11, 2009.] http://ida.first.fraunhofer.de/projects/bench/.

[4] A. Bordes, S. Estekin, J. Weston, L. Bottou, Fast kernel classifiers with online and active learning, Journal of Machine Learning Research. 6 (2005) 1579-1619.

[5] D. A. Cohn, Minimizaing statistical bias with queries, Advances in Neural Information Processing Systems. 6 (1997).

[6] D. A. Cohn, Z. Ghahramani, M. I. Jordan, Active learning with statistical models, Journal of Artificial Intelligence Research. 4 (1996) 129-145.

[7] C. Cortes, V. Vapnik, Support-vector networks, Machine Learning. 20 (1995) 273-297.

[8] C. K. Dagli, S. Rajaram, T. S. Huang, Utilizing information theoric diversity for SVM active learning, 18th International Conference on Pattern Recognition. (2006) 506-511.

[9] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, R. C. Williamson, Estimating the support of a high-dimensional distribution, Microsoft Research. (1999) 30.

[10] J. Jiang, H. H. S. Io, Dynamic distance-based active learning with SVM, Proceedings of the 5th international conference on Machine Learning and Data Mining in Pattern Recognition. (2007) 13.

[11] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, K. R. K. Murthy, Improvements to Platt's SMO algorithm for SVM classifier, Neural Computation. 13 (2001).

[12] M. I. Mandel, G. E. Poliner, D. P. W. Ellis, Support vector machine active learning for music retrieval, Multimedia Systems. 12 (2006) 3-13.

[13] W. Karush, Minima of functions of several variables with inequalities as side constraints, M.Sc. Dissertation, Dept. of Mathematics, Univ. of Chicago. (1939).

[14] H. W. Kuhn, A. W. Tucker, Nonlinear programming, Proceedings of 2nd Berkeley Symposium (1951) 481-492.

[15] J. C. Platt, Fast training of support vector machines using sequential minimal optimization, Advance in Kernel Methods. (1999).

[16] N. Roy, A. McCallum, Toward optimal active learning through sampling estimation of error reduction, Proceeding 18th International Conference on Machine Learning. (2001) 441-448.

[17] D.M.J. Tax, R.P.W. Duin, Support vector domain description, Pattern Recognition Letters. 20 (1999) 1191-1199.

[18] V. Vapnik, The Nature of Statistical Learning Theory, Springer, New York, (1995).

[19] Q. Wang, Y. Guan, X. Wang, SVM-Based spam filter with active and online learning, Proceedings of TREC. (2006).

Chapitre 2

Partitionnement efficace des données pour SVC

Cet article propose L-CRITICAL, un algorithme de partitionnement des données en sous-groupes homogènes disjoints pour la méthode « *Support Vector Clustering* ». L'objectif de cet algorithme est d'identifier l'ensemble de groupements intrinsèques à un ensemble de données arbitraire, et de produire un partitionnement robuste et précis des observations en fonction des sous-groupes détectés. L'algorithme repose sur une analyse topologique fonctionnelle de la solution d'un SVDD décrivant les contours des segments, et cherche à caractériser les chemins d'interconnexion entre les points critiques situés à l'intérieur des contours, permettant ainsi de distinguer les segments. Les résultats expérimentaux confirment que l'algorithme proposé améliore significativement la précision du processus de partitionnement des données dans un contexte de SVC comparativement aux compétitifs, tout en minimisant significativement le temps de calcul nécessaire sur tous les ensembles de données analysés.

La contribution de l'auteur (V. D'Orangeville) à cet article représente 90% de la charge de travail globale liée au développement des algorithmes et de la rédaction de l'article.

Efficient Cluster Labeling for Support Vector Clustering

V. D'Orangeville, A. Mayers, E. Monga and S. Wang

Abstract — We propose a new efficient algorithm for solving the cluster labeling problem in Support Vector Clustering (SVC). The proposed algorithm analyzes the topology of the function describing the SVC cluster contours and explores interconnection paths between critical points separating distinct cluster contours. This process allows distinguishing disjoint clusters and associating each point to its respective one. The proposed algorithm implements a new fast method for detecting and classifying critical points while analyzing the interconnection patterns between them. Experiments indicate that the proposed algorithm significantly improves the accuracy of the SVC labeling process in the presence of clusters of complex shape, while reducing the processing time required by existing SVC labeling algorithms by orders of magnitude.

1 Introduction

CLUSTER analysis is a learning procedure aimed at discovering intrinsic group structure in unlabeled patterns in order to organize them into homogeneous groups. Clustering analysis is a key area of data mining for which computationally efficient and accurate methods are needed to deal with very large-scale datasets in terms of data volume, data dimensionality and clusters complexity.

Support Vector Clustering (SVC) is a clustering algorithm proposed in 2000 by Ben-Hur [1] that uses the solution of the Support Vector Domain Description (SVDD) [2] model to group data points into clusters. While the SVDD algorithm produces contours that estimate a level set of the unknown distribution function of a dataset, the SVC method interprets these contours as cluster cores and assigns each data point to its nearest core to generate the final clusters.

The SVDD generates cluster boundaries by projecting a dataset into a nonlinear feature space via the use of Gaussian kernels, and by defining a sphere of minimal radius which encloses most data points. In the input space, the hypersphere surface defines a set of contours that can be regarded as an estimate of the dataset domain exploited by the SVC algorithm. While providing a description of the cluster cores, the SVDD method lacks information that connects each individual point to its membership cluster, hereby necessitating algorithms such as the one proposed in this paper to solve the cluster labeling process.

From a cluster analysis perspective, the SVC method has attractive properties. It allows controlling the number of clusters and their shape complexity by simply varying the Gaussian kernel bandwidth σ. It also allows controlling the sensitivity to outliers with a single parameter ρ representing the rejection rate for cluster boundaries. Finally, it defines clusters based on the structural risk minimization principles that are more robust to outliers.

Ben-Hur proposed a simple labeling algorithm [1] (referred to as BENHUR in this paper) based on an interconnection test that assumes that a pair of patterns belongs to the same cluster if both can be connected by a virtual segment located within a common contour. This test verifies the inclusion of test points along the connecting segments, and is repeated for every combinations of pairs of points. This exhaustive test allows creating an adjacency matrix that is used to partition data points into distinct clusters. As described in Section 5, experiments show that the method suffers from intractable processing time on moderately sized datasets. Moreover, the interconnection test is inaccurate when dealing with high rejection rates ρ as it results in data points being excluded from the contours and thus considered wrongly as singleton clusters as they cannot be interconnected internally.

Lee partially addressed the high processing requirements of Ben-Hur's method by proposing an algorithm referred to here as LEE [3]. It simplifies the labeling process by first grouping together data points distributed aroung a same local minimum of the function describing the cluster contours. It then tests the interconnection between each pairs of local minima (similarly to BENHUR interconnection test) to deduce the inner partitioning of the dataset. Although less time consuming than BENHUR, Lee's method still suffers from high computational complexity due to the repetition of gradient descents starting from each point of the dataset. In addition, experiments presented in Section 5 show that Lee's method produces high labeling error rates when dealing with complex datasets displaying narrow or curved cluster contours.

We should mention that Lee has presented in [4] an evolution of his previous method [3] that extends his interconnection tests between saddle-points and minima instead of minima only. The method is shown to be accurate in presence of complex clusters. This method came to our attention while submitting this paper for review, and shares some similarities with the algorithm proposed in this paper. It is not evaluated in our experiments as few implementation details are discussed in Lee's paper. We do include a discussion comparing the two methods in Section 4.4.

Jung proposed in [8] an extension to Lee's algorithm, grouping training points distributed around identical local minima, then by checking inteconnections between pairs of minima by performing linear interconnection tests. It enhances Lee's implementation by adding a process where similar descent trajectories are merged together during the minimization process toward local minima, in order to reduce the time complexity of the algorithm on large-scale datasets. As discussed in Section 5, Jung's algorithm exhibits similar labeling accuracy to Lee's method, while reducing significantly the labeling time.

In this paper, we propose a new labeling method, named L-CRITICAL, that provides an exceptionally high labeling accuracy even in the presence of complex cluster shapes, while reducing the required processing time by orders of magnitude in comparison to BENHUR and LEE. Our approach is based on the analysis of the topology of the function describing the cluster contours, and analyzes interconnection paths between all critical points (minima and saddle points) to ensure a more accurate and flexible interconnection test than the one implemented in BENHUR, LEE and JUNG. In particular, this interconnection test is performed by exploring Quasi-Newton descent trajectories toward local minima and saddle-points. As reported in Section 5, L-CRITICAL provides a state-of-the-art labeling accuracy in all our experiments, outperforming significantly BENHUR and LEE while dramatically reducing the labeling time. Although slower than JUNG on small datasets, the proposed method proved far more accurate and robust on all experiments and propose a pratical way of selecting the optimal merging parameter for accelerating the search for critical points without deteriorating accuracy.

2 Support Vector Domain Description

The SVDD is designed to characterize the support of the unknown distribution function of an input target class by computing a set of contours that rejects a controlled proportion ρ of patterns. These contours provide an estimate of a specific level set associated with the probability $1-\rho$ of the distribution function and allow unseen patterns to be classified as normal or abnormal.

2.1 Optimization Problem

Given a set X of training vectors $x_i \in \mathbb{R}^d$, $i=1,...,n$ and a nonlinear mapping ϕ from X to some high-dimensional nonlinear feature space Φ, we seek a hypersphere of center a and minimal radius R that encloses most data points and rejects a proportion ρ of the less representative patterns. This requires the solution of the following quadratic problem:

$$
\begin{aligned}
&\min_{R^2,\xi_i,a}\left\{R^2+C\sum_{i=1}^{n}\xi_i\right\}\\
&\left\|\phi_i-a\right\|^2 \le R^2+\xi_i,\\
&\xi_i \ge 0,\quad i=1,...,n.
\end{aligned}
\tag{1}
$$

Slack variables ξ_i are added to the constraints to allow soft boundaries, and ϕ_i denotes the coordinate $\phi(x_i)$ of x_i in the feature space. Points associated with $\xi_i > 0$ are excluded from the contours and penalized by a regularization constant C which controls a proportion ρ of points lying outside (and on the surface of) the hypersphere.

The optimization problem (1) can be solved by introducing the Lagrangian L as a function of primal variables R^2, ξ_i and a and dual variables α and β referred as Lagrange multipliers enforcing the two constraints in (1).

$$
\begin{aligned}
&L\left(R^2,\xi,a,\alpha,\beta\right)=R^2+C\sum_{i=1}^{n}\xi_i-\sum_{i=1}^{n}\alpha_i\left(R^2+\xi_i-\left\|\phi_i-a\right\|^2\right)-\sum_{i=1}^{n}\beta_i\xi_i\\
&\alpha_i,\beta_i \ge 0,\quad i=1,...,n.
\end{aligned}
\tag{2}
$$

Setting to zero the partial derivatives of formula (2) with respect to primal variables R^2, ξ_i and

a at the optimal point leads to:

$$\begin{aligned} &\frac{\partial L_P}{\partial R^2}: 1-\sum_{i=1}^{n}\alpha_i = 0 \rightarrow \sum_{i=1}^{n}\alpha_i = 1 \\ &\frac{\partial L_P}{\partial \xi_i}: C-\alpha_i-\beta_i = 0 \rightarrow C = \alpha_i + \beta_i \\ &\frac{\partial L_P}{\partial a}: -2\sum_{i=1}^{n}\alpha_i\phi_i + 2a\sum_{i=1}^{n}\alpha_i = 0 \rightarrow a = \sum_{i=1}^{n}\alpha_i\phi_i \end{aligned} \tag{3}$$

We can deduce from the constraints $C = \alpha_i + \beta_i$ in (3) and $\alpha_i, \beta_i \geq 0$ in (2) that $\alpha_i \leq C$. The Karush-Kuhn-Tucker (KKT) complementary slackness conditions results in:

$$\begin{gathered} \beta_i\xi_i = 0 \\ \alpha_i\left(R^2 + \xi_i - \|\phi_i - a\|^2\right) = 0 \end{gathered} \tag{4}$$

It follows from constraints (4) that the image ϕ_i of a point x_i with $\varepsilon_i > 0$ and $\alpha_i > 0$ lies outside (or on the surface) the feature-space sphere, and that a point x_i with $\varepsilon_i = 0$ and $\alpha_i = 0$ lies within the sphere. This indicates that the solution is sparse, only training vectors excluded from the decision surface with $\alpha_i > 0$ contributes to the SVDD solution. These vectors are referred to as *support vectors*.

By substituting eq. (3) into the primal Lagrangian (2) allows eliminating references to primal variables R^2, ξ_i and a, turning the Lagrangian into the Wolfe dual form L_d where $\langle \cdot,\cdot \rangle$ is the inner product of two possibly infinite vectors.

$$\begin{gathered} L_d: \ \max_{\alpha_i}\left\{\sum_{i=1}^{n}\alpha_i\langle\phi_i,\phi_i\rangle - \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\langle\phi_i,\phi_j\rangle\right\} \\ \sum_{i=1}^{n}\alpha_i = 1, \\ 0 \leq \alpha_i \leq C, \quad i = 1,\ldots,n. \end{gathered} \tag{5}$$

The dot product $\langle \phi_i, \phi_j \rangle$ in eq. (5) is replaced by an appropriate Mercer kernel $k(x_i, x_j)$, referred to as $k_{i,j}$ for notation simplicity, overcoming the explicit reference to ϕ_i of possible infinite dimension. The Gaussian kernel is used in this context, adjusting the complexity of the cluster contours with a single parameter σ controlling the kernel bandwidth.

$$k_{i,j} = e^{-\frac{1}{2\sigma^2}\|x_i - x_j\|^2} \tag{6}$$

The Wolfe dual is simplified by replacing the dot products $\langle \phi_i, \phi_j \rangle$ by the kernel $k_{i,j}$:

$$\begin{aligned} L_d:\ & \max_{\alpha_i} \left\{ \sum_{i=1}^{n} \alpha_i k_{i,i} - \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j k_{i,j} \right\} \\ & \sum_{i=1}^{n} \alpha_i = 1, \\ & 0 \le \alpha_i \le C, \quad i = 1, \ldots, n. \end{aligned} \tag{7}$$

The SVDD solution can consequently be optimized by maximizing the dual equation (7). Note that the problem remains convex since the kernel matrix K with i, j th entry $K_{ij} = k_{i,j}$ is positive definite.

2.2 Decision Function

As described in eq. (3), the center a of the hypersphere is described as a linear combination of the feature space vectors ϕ_i.

$$a = \sum_{i} \alpha_i \phi_i \tag{8}$$

The square Euclidian distance from an image ϕ_t of x_t to the sphere center a is defined as:

$$
\begin{aligned}
r^2(x_t) &= \|\phi_t - a\|^2 \\
&= k_{t,t} - 2\sum_i \alpha_i k_{i,t} + \sum_i \sum_j \alpha_i \alpha_j k_{i,j}
\end{aligned}
\tag{9}
$$

The decision surface is defined as the implicit surface $\{x : d(x) = 0\}$ of the function $d(x)$ described belowthat evaluates the relative position from the image $\phi(x)$ to the surface of the hypersphere. The function $d(x)$ classifies a point x inside the contours if $d(x) < 0$, on its surface if $d(x) = 0$ and outside otherwise.

$$
\begin{aligned}
&d(x_i) = O_S - O_i \quad \text{where} \quad O_i = \sum_j \alpha_j k_{i,j} \\
&O_S = \tfrac{1}{2}(O_i + O_j) \quad \text{where} \quad \begin{cases} i \leftarrow \arg\min_k O_k & s.t. \quad \alpha_k < C \\ j \leftarrow \arg\max_k O_k & s.t. \quad \alpha_k > 0 \end{cases}
\end{aligned}
\tag{10}
$$

The SVC cluster labeling process consists of separating the contour level associated with $d(x) = 0$ into a set of disjoint connected contours, and assigning each point to its nearest cluster contour.

3 Overview of cluster labeling algorithms

In this section, we describe three algorithms designed by Ben-Hur and Lee for SVC cluster labeling. Their underlying principles, qualities and limitations will serve as the basis from which to define the desired characteristics of the new SVC labeling algorithm proposed in this paper.

3.1 BENHUR

Ben-Hur proposed a method (BENHUR) that considers two points (x_s, x_t) as belonging to the same cluster if both can be connected by a straight path $\Gamma(\omega) = (1-\omega)\cdot x_s + \omega \cdot x_t$ entirely located within a same connected cluster contour characterized by the isosurface $\{x : d(x) = 0\}$. The path Γ is discretized and represented by a set of test points uniformly distributed along its way, and the interconnection test consists in verifying the inclusion of each point within the cluster contours. BENHUR performs an exhaustive evaluation of all interconnection paths between each pair of points of the dataset and then partitions all patterns into distinct groups based on the resulting adjacency matrix.

This method has the advantage of being simple from an implementation perspective, and relatively accurate with sufficiently large and low-dimensional datasets which provide an adequate distribution over the inner cluster volume. Smaller or high-dimensional datasets may result in high labeling error rates due to an insufficient coverage of data points that cause some crucial linear connection tests to fail. Also, the exhaustive interconnection test between all pairs of points constrains this method from being used on large datasets. In addition, Ben-Hur's algorithm is only suitable for models trained with a near zero rejection rate ρ. Choosing a higher ρ may result in many points being excluded from the contours and consequently being considered wrongly as outliers, as a consequence of the inability to connect them to any other points within the contours.

3.2 LEE

Lee's labeling algorithm (LEE) exploits a topological property of the SVDD solution illustrated in Figure 1 by which all points distributed around a common local minimum belong to a same cluster. First, the association of each point with its nearest converging local minimum allows grouping of the dataset into four groups. Then, each pair of local minima is tested using the linear interconnection test as depicted in Figure 2. Lee's labeling algorithm reduces the overall number of interconnection tests by proceeding to a gradient descent starting from each point of the dataset toward the nearest converging attractive minimum of $d(x)$. The patterns are then grouped and represented by their corresponding stable equilibrium, and the linear interconnection is evaluated exhaustively between each pair of minima using the same interconnection test implemented in BENHUR.

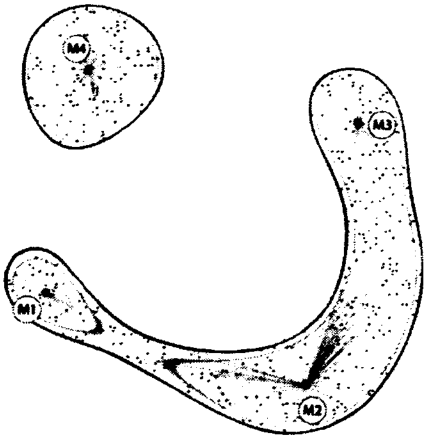


Figure 1 - Association of each point with its nearest converging local minimum M_i.

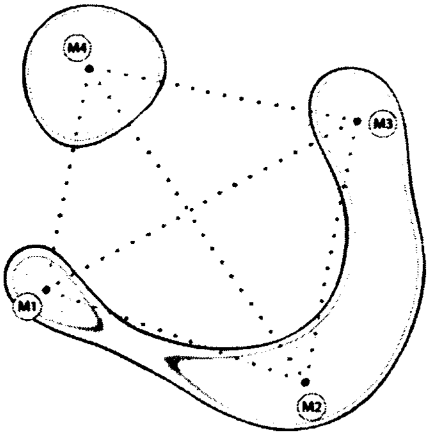


Figure 2 - Linear interconnection tests between each pair of local minima M_i. Green and red points are respectively included in and excluded from the cluster contours.

Although LEE provides a significant reduction in the number of interconnection tests compared to BENHUR, by testing interconnections between pairs of minima rather than all pairs of data points, it relies on the linear interconnection test that proves inaccurate in the presence of curved cluster contours as observed in Figure 2. LEE considers local minima that cannot be connected by any straight internal path as belonging to different and results in individual clusters of points being detected as multiple distinct clusters, as shown in Figure 3. Moreover, Lee's exhaustive gradient descent from each point of the dataset is computationally intense and makes this algorithm inefficient on large datasets.

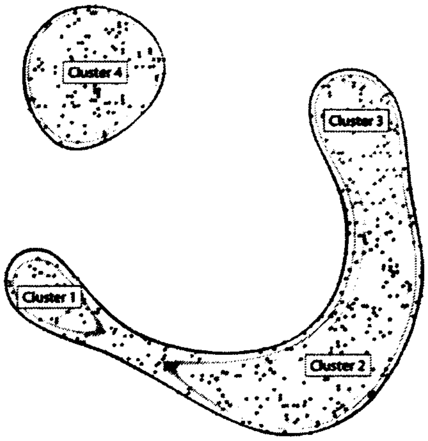


Figure 3 - Incorrect cluster labeling produced by LEE, detecting 4 clusters instead of 2.

3.3 DYNAMIC

Lee proposed a second labeling algorithm (DYNAMIC) in [4] which performs the interconnection test between saddle-points and minima instead of just minima (as implemented in LEE) to improve the accuracy of the labeling step.

DYNAMIC starts similarly to LEE, by carrying out a series of gradient descents over $d(x)$, starting from each point toward the nearest converging local minimum. The dataset is then partitioned into disjoint groups of points, each being represented by a distinct local minimum. The input space is then divided into grids to sample one data point per grid region. Each sample point is then used as starting point by a root detection algorithm to locate the nearest critical point (local minimum or saddle point) of the decision function $d(x)$. Eigenvalues and eigenvectors are then solved at the location of each critical point to allow classifying critical points as local minima or saddle points.

For each saddle point detected, the algorithm generates test points in the vicinity of the saddle point along the eigenvectors associated with negative eigenvalues. A gradient descent starting from each candidate is then performed, to detect the local minima adjacent to each saddle point. This process produces an adjacency matrix that summarizes the interconnections between local minima and saddle points, which is used to group data points into disjoint clusters.

3.4 JUNG

Jung proposed in [8] an extension (referred to as JUNG) to Lee's algorithm, solving the labeling phase by grouping training points converging toward identical local minima, then by assessing interconnections between pairs of minima by performing linear internconnection tests. JUNG differs from Lee's implementation by a process of merging similair descent trajectories during the minimization process toward local minima, in order to reduce the time complexity of the algorithm on large-scale datasets. JUNG exhibits similar labeling accuracy to Lee's method, while reducing significantly the labeling time. Although significantly faster than LEE, JUNG's method relies on a merging radius as input parameter for defining at which viscinity descent trajectories should be merged together. The drawback to this approach is it's lack of criteria for selecting the merging radius. Such a criteria will be proposed in the following section.

4 Efficient SVC labeling

We first present in the following subsection our approach for locating the nearest attractor and/or saddle point of the function $d(x)$ for every point of the dataset. This algorithm plays a central role in the proposed labeling algorithm in Section 4.3, as the labeling algorithm exploits this information to partition data points into distinct clusters.

4.1 Detection of critical points

We present a new Quasi-Newton (QN) optimization scheme that we used in two distinct contexts, for detecting either all minima or all minima and saddle-points. The method acts by minimizing, respectively, the function $d(x)$ or the squared norm $N(x)$ of the gradient of $d(x)$, with each point $x \in X$ acting as a starting location (see Table 1). Minimizing $d(x)$ is equivalent to minimizing the distance separating a point from the hypersphere center, and minimizing $N(x)$ is equivalent to finding the roots of the gradient $\nabla d(x)$.

Table 1 - Functions Minimized when Searching for Minima or Critical Points.

Search for	**Optimization problem**
All minima	$\arg\min_x d(x)$ with $d(x) = O_s - O(x)$
All critical points	$\arg\min_x N(x)$ with $N(x) = \lVert \nabla d(x) \rVert^2$

Detecting efficiently all of the nearest local minima or critical points (minima and saddle points) for all points in the dataset is a challenging task, for two reasons. First, the efficiency and accuracy of this process depends on the appropriate choice of the QN maximal step length λ and stopping criterion η that ensure convergence into the nearest local minimum within the minimal number of steps without jumping over the minimum. Secondly, the QN optimzation processes initiated from different points could generate redundant descent trajectories that can be merged together by using a quantization process which will be described shortly.

Our contribution to this problem is two-fold. We first design a method for calculating the optimal Quasi-Newton step length λ and stopping criterion η for each context stated in Table 1. We then design a method for discarding redundant descent paths that trades a negligible potential loss in accuracy for a massive gain in total processing time.

4.1.1 Quasi-Newton step length and stopping criterion

The value of the QN maximal step length λ depends on the minimal distance d_{crit} between two critical points, which is equal to the minimal distance between a local minimum and a saddle point on the surface of $d(x)$. As any pair of adjacent minima are necessarily separated by a saddle point, the minimal distance between two minima is $d_{\min} = 2d_{crit}$. Setting respectively $\lambda_{\min} = \frac{1}{2}d_{\min}$ and $\lambda_{crit} = \frac{1}{2}d_{crit}$ ensures the maximal step length to be always smaller than the distance $d_{\min}$ or d_{crit} between two adjacent stable equilibria of $d(x)$ or $N(x)$, and eliminates the possibility of jumping over a stable local equilibrium during a QN descent. The QN stopping criteria $\eta_{\min}$ and η_{crit} are evaluated as the norm of the gradient of $d(x)$ or $N(x)$ in the vicinity of a local minimum or saddle point.

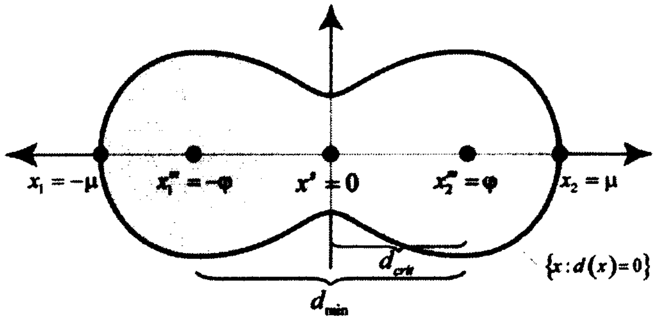


Figure 4 - Configuration of critical points on a 2D trained SVDD solution. Green and red points represent minima and saddle points, respectively.

Let's consider the two-dimensional SVDD model illustrated in Figure 4 trained on two points of coordinates $x_1 = -\mu$ and $x_2 = \mu$. For a kernel bandwidth σ with x_1 and x_2 located sufficiently close to each other, the SVDD solution exhibits a single connected contour with a local minimum at $x^m = 0$, and x_1 and x_2 lying on each opposite side of x^m on the cluster contour. Increasing the distance between x_1 and x_2 will at some point induce the split of x^m into two local minima $x_1^m = -\varphi$ and $x_2^m = \varphi$ separated by a saddle point $x^s = 0$ as illustrated in Figure 4.

Let's refer as $x_1 = -\mu_{\min}$ and $x_2 = \mu_{\min}$ the coordinates of the two points at which the cluster does not exhibit yet two local minima, and $x_1 = -\mu_{\max}$ and $x_2 = \mu_{\max}$ the pair of points at maximum distance that still forms a single cluster. Figure 5 illustrates these two extreme points configuration, the gray curves displaying the values of $d(x)$ for two SVDD models respectively trained on $\{x_1 = -\mu_{\min}, x_2 = \mu_{\min}\}$ and $\{x_1 = -\mu_{\max}, x_2 = \mu_{\max}\}$. Due to the symmetric configuration of x_1 and x_2, and given the dual constraint $\sum_i \alpha_i = 1$, both Lagrangian multiplier values are set to $\alpha_1 = \alpha_2 = \frac{1}{2}$.

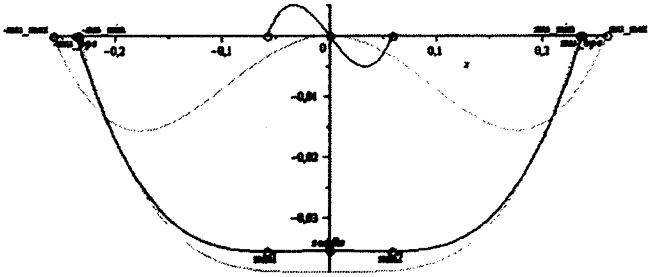


Figure 5 - First derivative $d'(x)$ (green curve), local minima (green circles) and saddle points (red circles) associated with the function $d(x)$ (black curve).

The first configuration (coordinates $\pm\mu_{\min}$) shows a SVDD model exhibiting a single local minimum at $x = 0$, and the second configuration (coordinates $\pm\mu_{\max}$) illustrates the extreme case where x_1 and x_2 are moved apart to the point where the curve exhibits a saddle point at the origin and two symmetrical local minima. The black curve represents an intermediate model trained on a pair of coordinates $x_1 = -\mu_\varepsilon$ and $x_2 = \mu_\varepsilon$ with $\mu_\varepsilon = (1-\upsilon)\mu_{\min} + \upsilon\mu_{\max}$ and υ a small constant. The latter curve illustrates the contour configuration at which a local minimum splits into two local minima and a saddle point as μ is increased, describing the minimal distance between pairs of adjacent local minima and a saddle point controlled by the parameter $\upsilon \approx 0.01$. The factor υ allows controlling the tradeoff between the accuracy of the critical points set detected, and the computational cost associated to their detection. In our implementation, υ is set a small value ($\upsilon \approx 0.01$) to ensure a conservative minimal distance between critical points that will lead to an accurate critical points detection while reducing the mislabeling rate, at the cost of an increased computational burden.

Based on the kernel bandwidth σ, the values of $\mu_{\min}$ and $\mu_{\max}$ can be analytically derived as $\mu_{\min} = \sqrt{\sigma}$ and $\mu_{\max} = 0.6094/\sqrt{0.6094/2\sigma}$. The coordinates of the local minima (referred to as *min1* and *min2* in Figure 5) are evaluated as $x_2^m = \mu_{\min} + 0.1761\sqrt{\upsilon}\left(\mu_{\max} - \mu_{\min}\right)$ and $x_1^m = -x_2^m$, and the coordinate of the saddle point as $x^s = 0$. The minimal distances $\lambda_{\min}$ and λ_{crit} between a pair of local minima and between a minimum and a saddle point are estimated as $\lambda_{\min} = 2x_2^m$ and $\lambda_{crit} = x_2^m$ based on the values of σ and υ.

The value of $\mu_{\max}$ is computed by solving the value of μ for which the first derivative of d_t equals to zero at the origin $x_t = 0$, as illustrated in eq. eq. (11).

$$d(x=0)=0 \quad \Rightarrow \quad \mu_{\max} = \frac{0.6094}{\sqrt{0.6094\rho}} \tag{11}$$

The value of $\mu_{\min}$ is calculated by solving for which value of μ the curvature (second derivative) of the function d_t equals to zero at the origin $x_t = 0$, as illustrated in (12).

$$
\begin{aligned}
&\frac{\partial^2 d_t}{\partial^2 x_t} = -\rho\left(\left(1+2\rho\left(x_t+\mu_{\min}\right)^2\right)e^{\rho(x_t+\mu_{\min})^2}+\left(1+2\rho\left(x_t-\mu_{\min}\right)^2\right)e^{\rho(x_t-\mu_{\min})^2}\right)\\
&\frac{\partial^2 d(x_t=0)}{\partial^2 x_t} = 0 \;\Rightarrow \quad \Rightarrow\; \mu_{\min} = \sqrt{\sigma}
\end{aligned}
\tag{12}
$$

The minimal QN stopping criterion η is calculated as the norm of the gradient at a small distance ε from a stable equilibrium. In the context of a search for minima, the value of η is computed as the norm $\eta_{\min} = \left\|\nabla d\left(x_2^m - \varepsilon\right)\right\|^2$ of the gradient of $d(x)$ in the neighborhood of the minimum x_2^m (with $\varepsilon = 10^{-3}$). The value of η is calculated as the norm $\eta_{crit} = \left\|\nabla N(\varepsilon)\right\|^2$ of the gradient of $N(x)$, when searching for critical points. The estimation of $\eta_{\min}$ is based on the coordinate $x_2^m - \varepsilon$ rather than $x_2^m + \varepsilon$, as the norm of the first coordinate is more restrictive.

Table 2 - Maximum Quasi-Newton step lengths and stopping criteria.

	Maximum step length	**Stopping criterion**
Search for critical points	$\lambda_{crit} = \sqrt{\sigma} + 0.1761\sqrt{\upsilon}\left(0.6094/\sqrt{0.6094/2\sigma} - \sqrt{\sigma}\right)$	$\eta_{crit} = \left\|\nabla N(\varepsilon)\right\|^2$
Search for minima	$\lambda_{\min} = 2\lambda_{crit}$	$\eta_{\min} = \left\|\nabla d\left(\lambda_{crit} - \varepsilon\right)\right\|^2$

Table 2 summarizes the maximum step lengths $\lambda_{\min}$ and λ_{crit} and the stopping criteria $\eta_{\min}$ and η_{crit} calculation for both optimization problems.

Note that the minimal distances $d_{\min}$ or d_{crit} holds for any dimensions greater than 2, as the coordinates of $x_1 = -\mu$ and $x_2 = \mu$ would simply be extended by adding zeros for all other coordinates.

4.1.2 Critical points detection

This section describes our method for assigning data points to their nearest local minimum or critical point.

The algorithm works by splitting the set of QN minimizations starting from each point into a sequence of quick QN iterations using loose stopping criterion η. Once the QN steps converge, a quantization step merges together descent trajectories within a small radius δ as they will likely converge toward the same stable equilibrium. Each redundant group of points are discarded and represented by a single candidate that will follows QN steps, dramatically reducing the overall number of QN iterations.

The stopping criterion η^t is adjusted at each optimization iteration t, going from a loose value $\eta^{t=0}$ to a restrictive value $\eta^{t=end} = \eta_{\min}$ or η_{crit} to ensure a convergence on the stable equilibrium and to also allow eliminating the most redundant descent trajectories early in the process. The stopping criterion is computed as $\eta^t = 10^{t_{\max}-t}\eta$, where $t_{\max} = 3$ is the total number of QN iterations and $\eta = \eta_{\min}$ or η_{crit}.

The quantization radius value δ is calculated from the minimal distances $\lambda_{\min}$ or λ_{crit} between pairs of minima or critical points. As all points within a radius $\frac{1}{2}\lambda_{\min}$ or $\frac{1}{2}\lambda_{crit}$ converge toward the same attractive minimum of $d(x)$ or $N(x)$, our quantization steps uses $\delta_{\min} = \frac{1}{2}\lambda_{\min}$ and $\delta_{crit} = \frac{1}{2}\lambda_{crit}$ as merging radii without adversely affecting the final set of stable equilibria detected.

The algorithm for assigning all patterns to their nearest converging minima or critical point is described in Algorithm 1.

Algorithm 1 - Quasi-Newton with adaptative quantization.

Input parameters:

	Search for minima	**Search for critical points**
Function to minimize	$f(x)=d(x)$	$f(x)=N(x)$
QN max step length	$\lambda=\lambda_{\min}$	$\lambda=\lambda_{crit}$
Quantization radius	$\delta=\lambda_{crit}$	$\delta=\frac{1}{2}\lambda_{crit}$
Convergence tolerance	$\eta=\eta_{\min}$	$\eta=\eta_{crit}$

$t_{\max}$: number of optimization iterations

Q_{in} : dataset (possibly previously quantized)

ID : parent indices for $x \in Q_{in}$

Main process:

```
1. Initialization
   $t = 0$
   $Q^0 \leftarrow Q_{in}$

2. Initial quantization
   $\forall (x_u, x_v) \subset Q^t,\ \|x_u - x_v\| \le \delta$
       $\Rightarrow Q^t \leftarrow Q^t / \{x_v\},\ ID_v = ID_u$

3. Optimization / quantization

   For $t = 1$ to $t_{\max}$ do
       QN convergence criterion
       $\eta^t = 10^{t_{\max}-t}\eta$
       Partial QN minimization
       For each $x \in Q^t$ do
           $x \leftarrow$ Quasi-Newton Routine$(x; f(x), \lambda, \eta^t)$
           Quantization
           $\forall (x_u, x_v) \subset Q^t,\ \|x_u - x_v\| \le \delta$
               $\Rightarrow Q^t \leftarrow Q^t / \{x_v\},\ ID_v = ID_u$

4. Return $(Q^{t_{\max}}, ID)$
```

The quantization step in Algorithm 1 should be understood as follows. If a pair of points (x_u, x_v) exists in the set Q^t at iteration t, such that their relative Euclidian distance is smaller than the quantization radius δ, discard x_v from Q^t and set the label ID_v associated to x_v as ID_u. The vector ID then reflects that the point x_v has been discarded and is represented by x_u.

Quasi-Newton implementation

The implementation of the Quasi-Newton algorithm is inspired from the implementation proposed in Numerical Recipes [8]. The latter is based on the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method [9]. Note that the details provided below is a partial excerpt from [8].

The BFGS formula for updating the approximation to the inverse Hessian matrix of function f_i for intermediate solution x_i, is described as:

$$\begin{aligned} H_{i+1} = H_i + & \frac{(x_{i+1}-x_i)\otimes(x_{i+1}-x_i)}{(x_{i+1}-x_i)\cdot(\nabla f_{i+1}-\nabla f_i)} \\ & - \frac{\left[H_i\cdot(\nabla f_{i+1}-\nabla f_i)\right]\otimes\left[H_i\cdot(\nabla f_{i+1}-\nabla f_i)\right]}{(\nabla f_{i+1}-\nabla f_i)\cdot H_i\cdot(\nabla f_{i+1}-\nabla f_i)} \\ & + \left[(\nabla f_{i+1}-\nabla f_i)\cdot H_i\cdot(\nabla f_{i+1}-\nabla f_i)\right]u\otimes u \end{aligned} \tag{13}$$

where $\otimes$ denotes the "outer" or "direct" product of two vectors, a matrix where the ij component of $u\otimes v$ is $u_i v_i$, and where u is a vector defined as

$$\begin{aligned} u \equiv & \frac{(x_{i+1}-x_i)}{(x_{i+1}-x_i)\cdot(\nabla f_{i+1}-\nabla f_i)} \\ & - \frac{H_i\cdot(\nabla f_{i+1}-\nabla f_i)}{(\nabla f_{i+1}-\nabla f_i)\cdot H_i\cdot(\nabla f_{i+1}-\nabla f_i)} \end{aligned} \tag{14}$$

The C++ implementation used in the proposed algorithm is an adaptation of the C code provided in [8].

4.2 Optimal interconnection test

The proposed interconnection test analyzes interconnection paths between critical points of $d(x)$ in relation to the cluster contours to split the set of contours into distinct contours and associate each pattern to its nearest contour. We now present concepts that introduce our new optimal interconnection test used by our labeling method.

Consider all possible continuous paths $\Gamma \in P$ connecting two local minima x_s and x_t such as $d(m_s) \leq 0$ and $d(m_t) \leq 0$. Let's define as $\Gamma^* = \arg\min_{\Gamma \in P}\left(\max_{y_u \in \Gamma} d(y_u)\right)$ the path connecting two local minima (x_s, x_t) exhibiting the maximum probability of being included within the contours. Of all possible paths $\Gamma \in P$, Γ^* has the smallest maximal value of $d(x)$ along its trajectory and if no other local minima lie along Γ^*, the latter will pass through one saddle point $s_u = \arg\max_{y_u \in \Gamma^*} d(y_u)$. If $d(s_u) \leq 0$, it confirms that the entire path Γ^* is located with a single cluster contour and that the two minima x_s and x_t and all point converging toward these two minima belong to a same cluster.

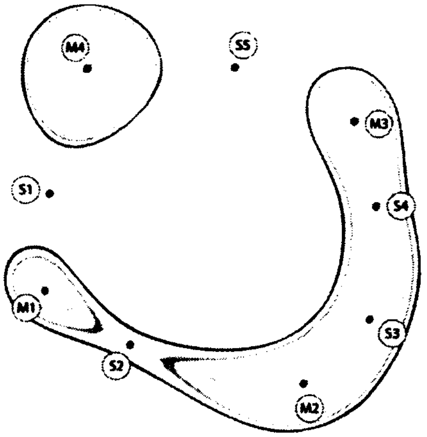


Figure 6 - Interconnection paths between local minima (blue points) and saddle points (red points).

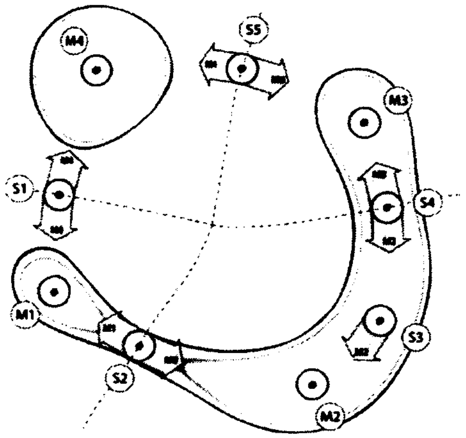


Figure 7 - Convergence paths (arrows) toward local minima M_i (blue points), in the neighborhood of saddle points S_i (red points).

Figure 6 illustrates this concept, the maximum probability path Γ^* represented by gray curves connect adjacent pairs of local minima (m_s, m_t) separated by a saddle point s_u at the highest value of $d(x)$ along Γ^* whose relative position to the cluster contour determines if the two minima are internally interconnected. This principle plays a key role in the design of the L-CRITICAL labeling method presented in the next section.

4.3 L-CRITICAL

We describe in this section L-CRITICAL, a new SVC labeling methods designed to achieve a high cluster labeling accuracy within competitive training time. L-CRITICAL relies on no specific input parameter, beside a trained SVDD solution and a dataset to label.

L-CRITICAL starts by assigning each point of the dataset to its nearest attractive critical point (minimum or saddle point) using the algorithm presented in Section 4.1.2. The second step consists in performing the optimal interconnection tests introduced previously in Section 4.2 whose implementation is centered around the principle explained next and illustrated in Figure 7.

Let's consider two sets of points $X_{\min}$ and X_{saddle} uniformly distributed along the surface of two spheres of radius $r = \frac{1}{2}\lambda_{crit}$, centered respectively on a local minimum $x_{\min}$ and a nearby saddle point x_{saddle}. Performing gradient descents over $d(x)$ starting from each $x \in X_{\min}$ will result in all points converging toward the same local minimum $x_{\min}$, allowing the classification of the critical point $X_{\min}$ as a local minimum. Adversely, performing the same process over each $x \in X_{saddle}$ will result in all points converging toward one of the adjacent local minima (m_u, m_v), allowing classifying x_{saddle} as a saddle point connecting each adjacent minimum (m_u, m_v) along the path with maximum probability of inclusion described in Section 4.2. If $d(x_{saddle}) \leq 0$, then both minima can be connected internally by a path Γ such that $d(x) \leq 0$ for $x \in \Gamma$ indicating that both minima belongs to the same cluster.

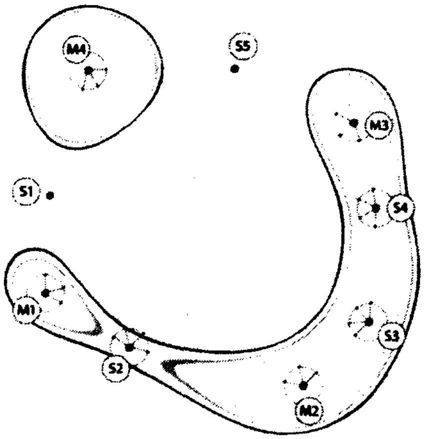


Figure 8 - Generation of test points (orange points) distributed around each critical point.

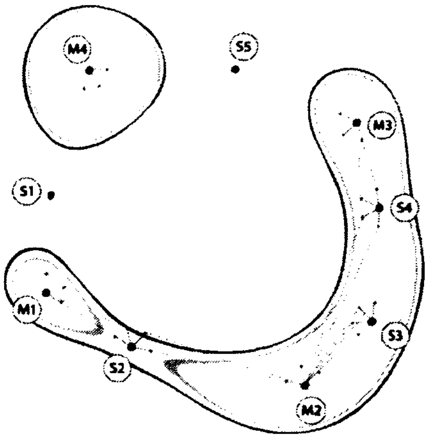


Figure 9 - Connections (orange lines) of all test points to their nearest attractive local minimum.

This principle is implemented in our method as follows. A set of points are generated on the surface of spheres of radius $r = \frac{1}{2}\lambda_{crit}$ centered around each critical point x_u^* detected such as $d(x_u^*) \leq 0$. The sets of test points are generated as the intersections between a) the virtual lines connecting each pair of critical points and b) each generated sphere as illustrated in Figure 8.

Each generated test points are then fed to the QN method described in Section 4.1.2 to detect their nearest attractive local minima (which is a subset of the critical points detected previously as illustrated in Figure 9). If a test point generated around a critical point converges toward the same critical point, it classifies the critical point as a local minimum. Adversely, if a test point converges toward a different critical point, it indicates that the original critical point is a saddle point linked to the attractive minimum. Repeating this process over all test points centered around all the critical points allows simultaneously classifying each critical point as a minimum or a saddle point. It also allows identifying saddle points on the paths with maximum probability of inclusion connecting pairs of adjacent minima. Evaluating the inclusion of each saddle point then allows deciding whether a pair of adjacent minima is connected within a same cluster contour and if they belong to the same cluster. By extension, this allows grouping data points converging toward these minima into their respective clusters.Note that in order to restrict the number of test points generated around each critical point, a quantization pass with radius r_t (15) is applied on the set of test points with $t_{\max} = 16$ representing the maximal number of test points distributed around the surface of the hypersphere.

$$r_t = \sqrt{2\lambda_{crit} \sin\left(\frac{\pi}{t_{\max}}\right)} \tag{15}$$

DERIVATION OF THE QUANTIZATION RADIUS r_t

Let's consider a circle of radius λ_{crit} divided into $t_{\max}$ arcs of equal lengths. As illustrated in figure 10, the angle between each arc is $\mathrm{A} = 2\pi / t_{\max}$, $b = c = \lambda_{crit}$ and $r_t = a$.

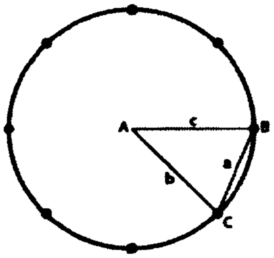


Figure 10 - Derivation of the quantization radius r_t for $t_{\max} = 8$ points equally distributed.

We solve the value $r_t = a$ using the law of cosine for an arbitrary triangle state where R is the radius of the circumscribed circle of the triangle.

$$\begin{aligned} &\frac{a}{\sin A} = 2R \\ &a = 2R\sin A \rightarrow r_t = 2\lambda_{rit} \sin\left(\tfrac{2\pi}{t_{max}}\right) \end{aligned} \tag{16}$$

Each of the critical point $x^* \in X^*$ is associated to a unique cluster label l_i initialized such as $L = \{l_i, i = 1..|X^*|\}$. When a test point generated around a critical point x_u^* such as $d(x_u^*) \leq 0$ converge toward a different critical point x_v^* when minimizing $d(x)$, x_u^* is classified as saddle point connecting internally the minima x_v^* to another minima, and the label value of x_u^* is set to the label value of x_v^* to reflect their membership to a same cluster. No test point is generated around a critical points x_u^* such as $d(x_u^*) > 0$ as these critical points cannot be connected internally to any other critical point. The cluster label of each data point of the dataset is finally computed based on the cluster label of the critical point to which it is connected.

Algorithm 2 - L-CRITICAL.

Input parameters:

Dataset X
SVDD solution with parameters σ and ρ

Main process:

1. **Estimate QN input parameters**
   Calculate the minimal distances ($\lambda_{\min}$ and λ_{crit}) between critical points and QN stopping criteria ($\eta_{\min}$ and η_{crit}) (Section 4.1.1).

2. **Discover all critical points (Section 4.1.2)**
   Apply QN routine to minimize $N(x)$ to discover all critical points and associate each data points to its converging critical point. QN routine is applied using quantization value $\delta_{crit}=\frac{1}{2}\lambda_{qn}^{crit}$, QN step length λ_{qn}^{crit} and stopping criterion η_{qn}^{crit}.

3. **Generate candidates around each critical point**
   Generate test points distributed around each critical points. Test points are distributed at a distance $r=\frac{1}{2}\lambda_{crit}$ from each critical point and then test points are merged together with a quantization radius r_t to reduce the number of test points.

4. **Associate candiates to their nearest minima**
   Link test points to minima by making test points converge toward local minima (Section 4.1.2), with quantization parameter $\delta_{\min}=\lambda_{qn}^{crit}$, QN step length $\lambda_{qn}^{\min}$ and stopping criterion $\eta_{qn}^{\min}$.

5. **Partition data points into clusters**
   Partition data points into disjoint clusters based on the cluster membership of each critical points stored in vector L and deduce each data point cluster label.

4.3.1 Complexity analysis of L-CRITICAL

The main time consuming phase of L-CRITICAL lies in the the numerical integration of each critical point (phase 2 of Algorithm 2), starting from each point of the training set. As the minimization process is performed on the reduced training set, where the n training points are merged into n_r candidates, the complexity of the search for critical points is $O(n_r d)$ where d is the number of dimensions of the data space. Note that as the dataset is reduced using the merging radius λ_{crit}, the size n_r of the reduced training set remains constant regardless of the training set size. Consequently, the time complexity of the proposed method is bounded on $O(n_r d)$ and is sub-linear in regard to the size of input.

4.4 Comparison between L-CRITICAL and DYNAMIC

To simplify the comparison between L-CRITICAL and DYNAMIC, the main steps of the two algorithms are summarized in Table 3.

Table 3 - Comparison between L-CRITICAL and DYNAMIC.

STEP	L-CRITICAL	DYNAMIC
1. Detection of critical points	All critical points are detected using QN method (Section 4.1.2) with optimal parameters (Section 4.1.1)	All local minima are detected using gradient descent starting from each data point. Test points are sampled and used as starting points for a root detection algorithm to detect critical points.
2. Classification of critical points	Test points are created around each critical points and fed to QN (same method used in Step 1) to connect them to their attractive minima and classify them as minima or saddle-point.	Eigenvalues and eigenvectors are calculated at each critical point detected. Each critical point is classified as minima or saddle-point based on its eigenvalue sign.
3. Analysis of interconnection paths between critical points	The analysis of interconnection paths between critical points is solved implicitely in Steps 1 and 2.	Test points are generated along eigenvectors of saddle points. Gradient descent starting from each test point is performed to detect adjacent minima and to connect saddle-points to minima.
4. Cluster labeling	The cluster labeling is solved implicitely in Steps 1 and 2	Cluster partitioning based on the adjacency matrix generated.

From a computational efficiency perspective, DYNAMIC exhibits the same high complexity of LEE as it performs exhaustively gradient descents starting from each point of the dataset to associate each point to its local minimum. L-CRITICAL solves this step in a more efficient way using the algorithm described in Section 4.1.2.

L-CRITICAL performs the detection, classification and interconnection analysis of critical points in two steps that rely on the same QN algorithm (Section 4.1.2), in contrast to DYNAMIC, which involves first searching for local minima and then performing a grid sampling on the input space to use sampled points as starting locations for a root detection algorithm to detect saddle points. As noted by the author in [4], an improper sampling could result in the failure to detect essential saddle points connecting adjacent local minima and in detecting a single connected cluster as multiple disjoint clusters. This limitation is circumvented in our approach.

The critical point classification process in L-CRITICAL is also more efficient as it is performed in Steps 2 and 3, simultaneously detecting, classifying and evaluating interconnection paths between critical points. DYNAMIC detection and classification of critical points require using a root detection algorithm and solving eigenvectors and eigenvalues, processes which are time consuming with high-dimensional datasets.

Finally, the interconnection analysis step in L-CRITICAL is more robust as it produces more candidates uniformly around each critical point, mitigating the event of candidates failing to converge toward an adjacent minimum potentially resulting in labeling errors. L-CRITICAL also generates candidates at a distance $r = \frac{1}{2}\lambda_{crit}$ from each critical point which is adapted to the RBF kernel bandwidth, as opposed to DYNAMIC which uses a constant distance which may yield labeling errors for small kernel bandwidths.

For all of the reasons previously discussed, it is reasonable to assume that L-CRITICAL is fundamentally more robust and efficient than DYNAMIC, although the two algorithms share conceptual similarities. Note that although we provided a comparison between DYNAMIC and L-CRITICAL, DYNAMIC was not tested in the experiments presented in the next section, as the author's paper provides only a general description of the algorithm, without supplying implementation details.

5 Experiments and results

The new SVC labeling algorithms (L-CRITICAL) proposed in this paper is first tested against the two competitive methods (BENHUR and LEE) on synthetic datasets sampled from 15 uniform density functions represented by the white regions of bitmap images illustrated in Figure 10. For each sampling size, 15 datasets are generated. Each of the three algorithms is executed on these datasets and their respective processing time and labeling accuracy are measured.

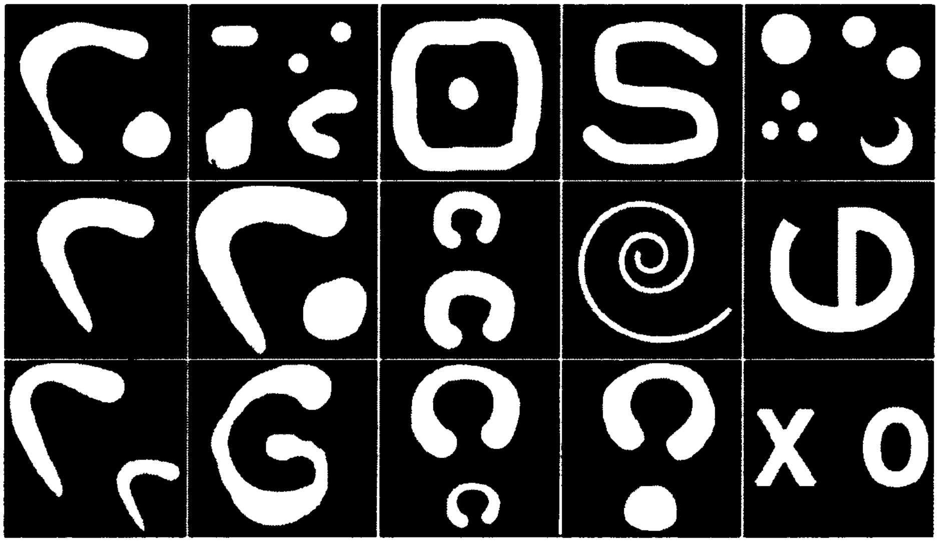
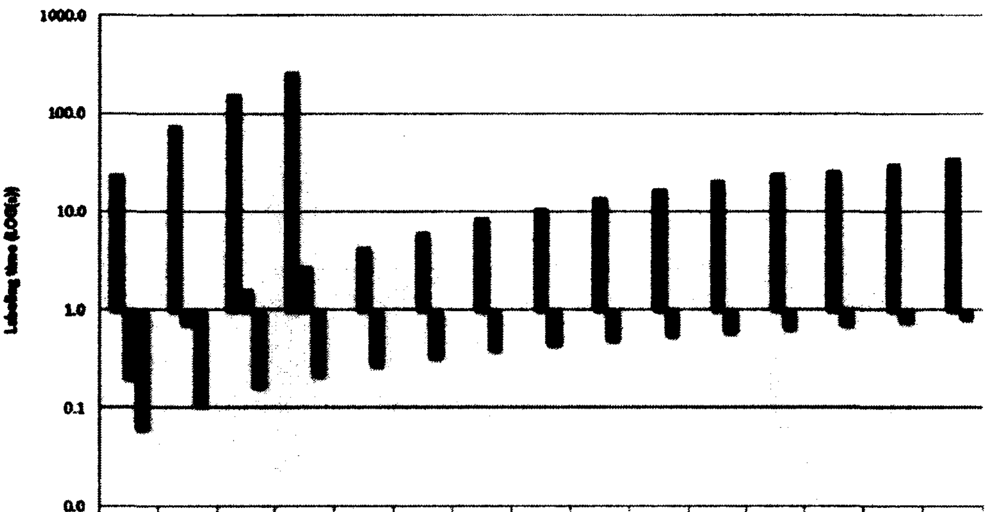


Figure 11 - 15 density maps for generating synthetic datasets tested in the experiments.

The density maps shown in Figure 11 are created in such a way to exhibit complex contours features on which existing SVC labeling algorithms typically fail to produce accurate partitioning. These density maps are used to generate datasets for testing labeling accuracy and robustness of the algorithms in presence of complex clusters features. These problematic contours features include curved contours, narrow contours, concavity (holes) within a cluster, concentric clusters (one within another) and distinct clusters located at proximity one from the others. Note that as most UCI benchmark clustering datasets exhibits clusters that are typically spherically or elliptically shaped. These datasets are consequently too simple to allow a proper evaluation of the labeling robustness of the labeling algorithms.

Figure 12 illustrates respective the processing time of each algorithm averaged over the 15 datasets of same however increasing sample size. All datasets used in this experiment are accessible on the authors' website[1] . All experiments were performed on an Intel Q6600 CPU.



	1000	2000	3000	4000	5000	6000	7000	8000	9000	10000	11000	12000	13000	14000	15000
■ BENHUR	23.99	75.87	157.38	265.76											
■ LEE	0.20	0.73	1.58	2.74	4.26	6.09	8.41	10.83	13.74	16.93	20.71	24.18	26.03	30.16	34.63
■ L-CRITICAL	0.06	0.11	0.16	0.22	0.27	0.33	0.39	0.44	0.49	0.54	0.60	0.65	0.71	0.77	0.80

Figure 12 - Labeling time (vertical axis) log(time) of each algorithm averaged over all 15 synthetic datasets, with sample sizes (horizontal axis) ranging from 1,000 to 15,000 data points.

As illustrated in Figure 12, L-CRITICAL dramatically outperforms LEE and BENHUR in terms of processing times by several orders of magnitude. Note that BENHUR's labeling times are not reported for training sets above 4,000 data points as we restricted the maximal processing times of each experiment to 360 seconds. The high efficiency of L-CRITICAL results from the highly efficient critical points detection algorithm presented in Section 4.1.2, which discards redundant descent trajectories and reduces dramatically the processing time of the detection of critical points, as opposed to LEE which performs exhaustively gradient descents starting from every point of the dataset. BENHUR's processing time becomes prohibitive on large scale datasets as it performs its linear interconnection test exhaustively between all possible pairs of points, yielding a number of tests which becomes intractable with increasing numbers of data

[1] www.usherbrooke.ca/prospectus/vdorangeville

points.

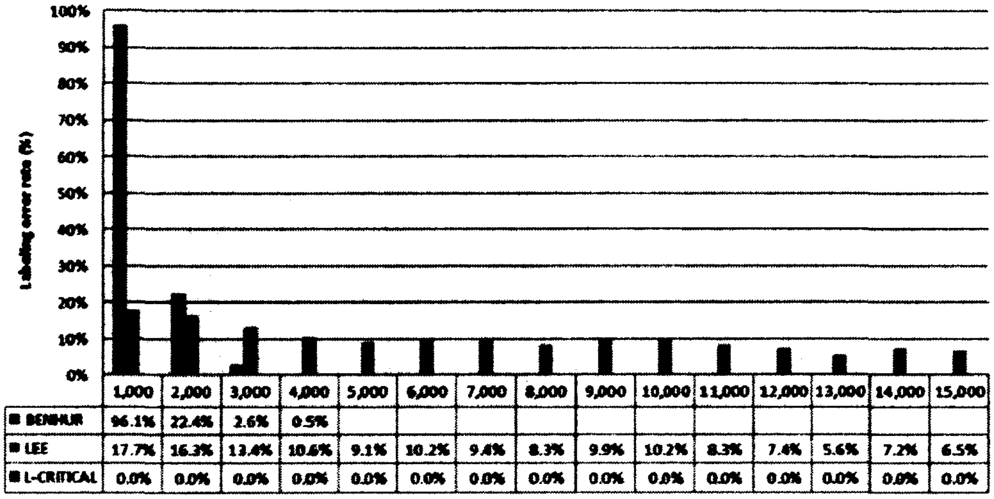


Figure 13 - Labeling error rates (vertical axis) of each algorithm averaged over all 15 synthetic datasets, with sample sizes (horizontal axis) ranging from 1,000 to 15,000 data points.

Figure 13 report the average proportions of labeling errors of each algorithm for all the tests performed. The L-CRITICAL method yields perfect labeling accuracy in every simulation, independently of dataset size. This high accuracy supports the excellent flexibility and robustness of its interconnection test, which deals efficiently with clusters of complex shapes. LEE exhibits a near constant labeling error rate on all datasets independently of the sample size due to its use of the linear interconnection test which fails to connect pair of local minima in presence of curved or narrow shaped clusters.

Finally, BENHUR exhibits a labeling error rate that decreases with sample size, providing an almost perfect accuracy on datasets of 4,000 points. It supports our assumption that on small sized dataset, BENHUR is affected by improper covering of the clusters inner volumes, preventing some crucial internal connections between data points and resulting in high labeling error rates.

A second set of experimentations is performed on a set of benchmark clustering datasets referred to as "Fundamental Clustering Problem Suite (FCPS)"[2]. FCPS offers a variety of clustering problems with known a priori classifications, intentionnally create to represent diverse type of data configuration on which standard clustering methods (single-linkage, ward and k-means) fails. The configurations exhibits clusters of different variances, different inter cluster distances, almost touching clusters, linearly not separable clusters and presence of outliers. This ensemble of datasets allows us to demonstrate the efficiency and accuracy of L-CRITICAL where conventional SVC labeling algorithms typically fail.

Table 4 illustrates the results of our experiments on 8 datasets, and compares the accuracy measured by adjusted rand index [10], the labeling time and the number of clusters detected. JUNG algorithm is presented using different values of merging radii in order to illustrates the dependency of its labeling time and accuracy to the merging radius.

Table 4 - Comparison of labeling algorithms on 8 datasets (FCPS) in term of adjusted rand index (ADJ_RI), labeling time (T(s)) and number of clusters detected (CLUSTERS). Jung's algorithm is applied using varying merging radii (0.01, 0.1 and 0.5).

	DATA		SVC			CRIT				LEE				BEN			
[illegible]	N	DIM	VAR	%OUT	[illegible]	ADJ_RI	T(s)	[illegible]	[illegible]	[illegible]	T(s)	CLUSTERS	[illegible]	ADJ_RI	T(s)	CLUSTERS	[illegible]
atom	800	3	0.1	10%	2	1.00	1.58	2	0	1.00	0.61	2	0	1.00	5.53	2	0
chainlink	1000	3	0.05	10%	2	1.00	0.30	2	0	0.85	0.85	3	1	0.99	5.99	6	4
Hepta	212	3	0.05	10%	7	1.00	0.03	7	0	1.00	0.03	7	0	1.00	0.45	7	0
Lsun	400	2	0.05	10%	3	1.00	0.14	3	0	1.00	0.13	3	0	0.53	1.71	3	0
Target	770	2	0.05	10%	6	1.00	0.13	6	0	0.86	0.39	7	1	1.00	4.21	14	8
Tetra	400	3	0.025	30%	4	0.71	1.43	4	0	0.71	0.18	3	1	0.71	1.59	3	1
TwoDiamonds	800	2	0.025	40%	2	0.99	0.60	2	0	0.02	0.64	3	1	0.00	6.11	78	76
WingNut	1016	2	0.01	0.10%	2	0.64	8.94	7	5	0.09	2.14	18	16	0.00	10.93	144	142

	DATA		SVC			JUNG(0.01)				JUNG(0.1)				JUNG(0.5)			
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
atom	800	3	0.1	10%	2	1.00	0.55	2	0	1.00	0.21	2	0	0.83	0.04	3	1
chainlink	1000	3	0.05	10%	2	0.85	0.58	3	1	0.69	0.14	4	2	0.49	0.03	5	3
Hepta	212	3	0.05	10%	7	1.00	0.01	7	0	1.00	0.01	7	0	1.00	0.00	7	0
Lsun	400	2	0.05	10%	3	1.00	0.12	3	0	1.00	0.04	3	0	0.95	0.00	3	0
Target	770	2	0.05	10%	6	0.78	0.20	8	2	0.78	0.05	8	2	0.69	0.01	9	3
Tetra	400	3	0.025	30%	4	0.71	0.25	3	1	0.71	0.21	3	1	0.64	0.02	11	7
TwoDiamonds	800	2	0.025	40%	2	0.02	0.54	3	1	0.01	0.16	3	1	0.00	0.01	1	1
WingNut	1016	2	0.01	0.10%	2	0.09	3.11	17	15	0.09	1.08	17	15	0.15	0.05	7	5

Figures 14, 15 and 16 illustrate the average labeling accuracy, number of errors in number of clusters detected and labeling times for the 8 datasets (FCPS).

[2] Available at http://www.uni-marburg.de/fb12/datenbionik/data?language_sync=1

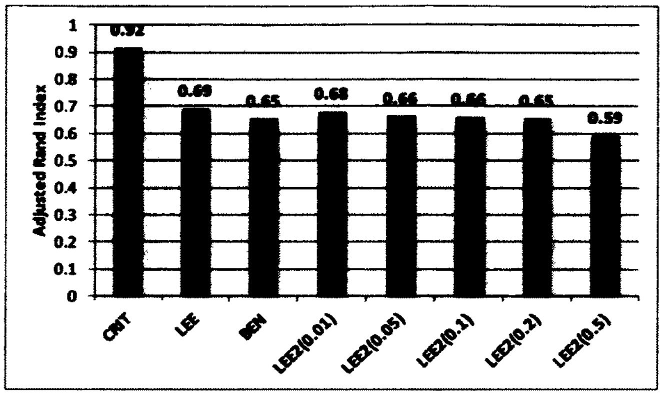


Figure 14 - Average labeling accuracy (adjusted rand index) for all 8 datasets (FCPS) presented in Table 4.

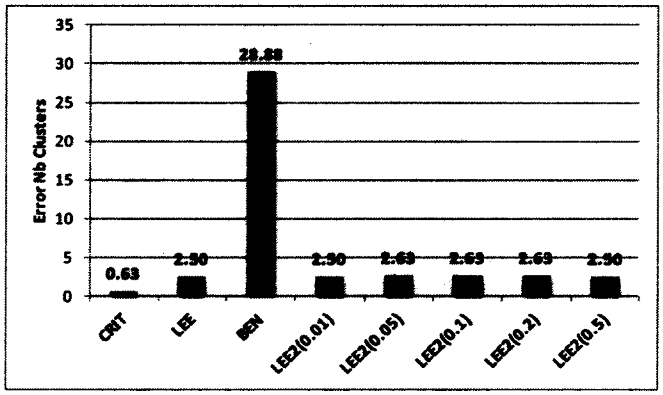


Figure 15 - Average errors in number of clusters detected for all 8 datasets (FCPS) presented in Table 4.

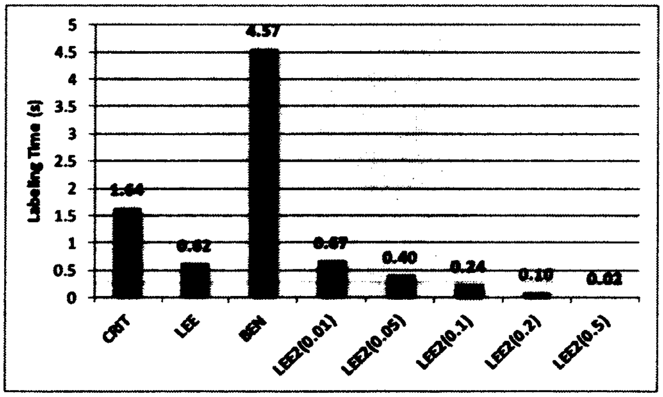


Figure 16 - Average labeling time for all 8 datasets (FCPS) presented in Table 4.

The analysis of the results of these experiments highlights two important properties of the labeling algorithms described in this article:

- BENHUR is computationally very intensive on small scale datasets, due to its computation of interconnection test between each pair of data points, although providing low labeling error rates on most datasets. The labeling accuracy is highly affected by the proportion of points excluded from the clusters contours as each of them is considered as a separate cluster.
- LEE provides an improvement in processing time over BENHUR although suffering from labeling error rates in presence of clusters with complex shapes. The exhaustive search for minima starting from each data points restricts its application to small sized dataset.
- JUNG provides a significant improvement in term of processing time over LEE and exhibits the same accuracy than LEE. However, a drawback of JUNG remains in its absence of strategy for selecting the merging radius during the numerical integration step. As illustrated in our experiments, the choice of a too large radius impacts negatively its labeling accuracy, while a very small radius reduce its labeling time to the one of LEE.
- L-CRITICAL exhibits very competitive labeling processing times while achieving perfect labeling accuracy in all experiments performed. Although slower than JUNG, L-CRITICAL adapts automatically its merging radius and is significantly more accurate than the other labeling methods tested.

The clear winner for SVC labeling is the L-CRITICAL algorithm, which outperforms existing state-of-the-art SVC labeling algorithms BENHUR and LEE by several orders of magnitude in terms of processing time, while yielding improved labeling accuracy on all the tests performed. Although slower than JUNG, its significant higher accuracy makes it the most compelling SVC labeling algorithm.

6 Conclusion

We have presented L-CRITICAL, a new SVC cluster labeling algorithm which efficiently and accurately solves the labeling phase of the Support Vector Clustering (SVC) method within competitive processing time. The proposed algorithm is based on a new efficient and accurate interconnection test between critical points of the function describing the SVC cluster contours, and allows distinguishing accurately distinct clusters in situations where most competitive labeling algorithms fail. Experiments indicate that the proposed algorithm provides a very satisfactory solution both in terms of labeling accuracy and processing time over BENHUR, LEE and JUNG in the presence of clusters of complex shape.

From our point of view, L-CRITICAL is the first method that can be realistically implemented for large real-world datasets while guaranteeing a state-of-the-art processing time and accuracy. The development of L-CRITICAL is essential to exploit SVC's ability to distinguish clusters of the high shape complexity typically encountered in handwritten character recognition and image clustering. This method allows us to benefit from SVC's high adaptability to the inherent characteristics specific to the data analyzed in a real-world data-mining context.

References

[1] A. Ben-Hur, D. Horn, H.T. Siegelmann, V. Vapnik, "A Support Vector Clustering Method", *Proceedings 15th International Conference on Pattern Recognition*, vol. 2, pp. 724-727, 2000.

[2] D.M.J. Tax, R.P.W. Duin, "Support Vector Domain Description", *Pattern Recognition Letters*, vol. 20, pp. 1191-1199, 1999.

[3] J. Lee, D. Lee, "An Improved Cluster Labeling Method for Support Vector Clustering", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, pp. 461-464, 2005.

[4] J. Lee, D. Lee, "Dynamic Characterization of Cluster Structures for Robust and Inductive Support Vector Clustering", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 1869-1874, 2006.

[5] W. Karush, "Minima of Functions of Several Variables with Inequalities as Side Constraints", M.Sc. Dissertation, Dept. of Mathematics, Univ. of Chicago, 1939.

[6] H. W. Kuhn, A. W. Tucker, "Nonlinear Programming", *Proceedings of 2nd Berkeley Symposium*, pp. 481-492, 1951.

[7] K. H. Jung, D. Lee, J. Lee, "Fast support-based clustering method for large-scale problems", *Pattern Recognition*, v.43, n.5, pp. 1975-1983, 2010.

[8] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, "Numerical Recipes. The Art of Scientific Computing", 3rd Edition, Cambridge University Press, 2007.

[9] A. Mordecai, "Nonlinear Programming: Analysis and Methods", Dover Publishing, 2003.

[10] N. X. Vinh, J. Epps and J. Bailey, "Information Theoretic Measures for Clustering Comparison: Is a Correction for Chance Necessary?" *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*. ACM., pp. 1073–1080, 2009.

Chapitre 3

Sélection des hyperparamètres pour SVDD

Cet article présente une stratégie robuste de sélection des hyperparamètres pour « *Support Vector Data Description* », permettant d'estimer le domaine de la fonction de distribution d'une classe cible à des fins de détection d'anomalies. La méthode proposée procède à une analyse des effets d'une variation des hyperparamètres (σ,ρ) sur la transformation des contours générés par un modèle SVDD, et identifie un ensemble d'hyperparamètres résultant en une estimation précise et compacte du domaine de la classe cible. La méthode intègre par ailleurs un mécanisme prévention du phénomène de surgénéralisation. Elle bénéficie d'un avantage crucial par rapport aux méthodes existantes supervisées [17] [13] puisque son processus d'induction dépend exclusivement d'observations de la classe cible sans requérir un ensemble de données classifiées comme atypiques. La performance de généralisation des modèles SVDD entrainés avec cette méthode a été évaluée sur des ensembles de données synthétiques et réels. Nous procédons à l'évaluation de la précision et de la robustesse de notre méthode à produire des représentations SVDD compactes permettant une discrimination efficace entre des observations normales et atypiques sur des ensembles de données artificiels, sujets à des proportions croissantes de bruit additif gaussien. Nous proposons ensuite des résultats expérimentaux comparant la performance de généralisation de notre méthode à l'algorithme « *abnomaly detection* » implémenté dans le logiciel SPSS Clementine 12.0, et démontrons la supériorité de notre approche.

La contribution de l'auteur (V. D'Orangeville) à cet article représente 90% de la charge de travail globale liée au développement des stratégies et de la rédaction de l'article.

Hyperparameters Selection for Support Vector Domain Description

V. D'Orangeville, A. Mayers, E. Monga and S. Wang

Abstract — This paper presents a new parameters selection strategy for the Support Vector Domain Description (SVDD) for automatic novelty/outlier detection. The method proceeds by detecting characteristic transformations of SVDD contour induced by parameters variations, which we use to develop an accurate estimate of the distribution support of the target class by the contours. The proposed method offers three major advantages over related strategies. Its entire inductive process relies exclusively on target class observations, i.e. positive instances, and does not require negative instances. The method is efficient on datasets of varying dimensions and sizes, and implements a mechanism preventing overfitting. Experiments on various synthetic and real-world datasets suggest that the proposed method allows identifying parameters yielding SVDD models that distinguish accurately normal patterns from outliers with an accuracy close to the optimal achievable separation for any set of parameters, significantly outperforming the SPSS Clementine 12.0 proprietary abnormality detection algorithm.

1 Introduction

THE *Support Vector Domain Description* (SVDD) is an algorithm introduced by Tax and Duin in 1999 [1] and inspired from the *Support Vector Machine* (SVM) algorithm proposed by Vapnik in 1995 [2]. The goal of the SVDD is to estimate the unknown distribution support of an arbitrary target class, to allow classifying unseen patterns as normal or abnormal.

The SVDD method acts by projecting a set of input patterns into a high-dimensional nonlinear feature space and by generating a hypersphere of minimal radius which encloses a controlled proportion of projected patterns. The hypersurface defines in input space a set of boundaries that provides an estimate of a level set of the target class data distribution function, enclosing the most representative input patterns and excluding the least representative ones. The projection into feature space is achieved implicitly by the use of Gaussian kernels computed on the patterns coordinates of input space. The Gaussian kernel is parameterized by a bandwidth σ that controls the complexity of the generated contours, and by a factor ρ that constrains the proportion of outliers rejected by the contours.

The SVDD has two important qualities: the simplicity of the interpretation of its parameters and its strong theoretical foundation. The complexity of its contours and sensitivity to outliers can be controlled respectively with only two parameters σ and ρ. Furthermore, the SVDD is founded on structural risk minimization principles that yield models theoretically less sensitive to noise compared to models based on empirical risk minimization principles. The main limitation of SVDD is the lack of criterion for selecting model parameters that generate a compact representation of the target data, prevent overfitting and distinguish accurately typical patterns from outliers.

Various approaches [3][4] have been proposed for optimizing parameters for SVDD, in a context where both positive (normal) and negative (abnormal) instances of a target class are available. These methods act by exploring combinations of hyperparameters and identifying parameters which result in a SVDD model that minimizes the classification error rate on positive and negative instances of the target class. For instance, the method proposed by Zhuang [3] makes use of grid-search strategies over the hyperparameters space and selects a SVDD model to achieve an accurate detection of abnormal patterns. Tax [4] proposed a similar strategy focusing on the selection of the Gaussian kernel bandwidth σ. It trains a series of SVDD models of increasing complexity and calculates a measure of the discriminative power (convex combination of type-I and type-II detection errors) of each model on positive and negative instances of the class. It then selects the minimal complexity model (largest σ) having a discrimination power index below a threshold value. This strategy suffers from important limitations as it provides no criterion for selecting the index threshold nor for optimizing the rejection rate ρ and has been validated only on a single dataset.

The most serious limitation of Zhuang's and Tax's selection strategies is that they rely on the availability of negative instances of the target class. However, most novelty detection problems involve situations where negative instances of a target class are unavailable or associated to high acquisition costs (insurance claim fraud detection, defect identification in supply chains, rare disease diagnosis, etc.). In fact, if these negative instances were readily available, a classification approach using, for example, the Support Vector Machine, would be more appropriate. In practical applications of novelty/outlier/abnomality detection, it is not reasonable to assume the availability of representative instances of novelty/outliers/abnomality.

Tax [5] proposed a different selection strategy addressing the situation where only positive instances of the target class are available. It acts by generating a cloud of data points covering uniformly the inner volume of a sphere enclosing all target class patterns. The set of SVDD parameters are selected as the one minimizing the classification error rate between positive patterns and artificially generated outliers. A fundamental limitation of this strategy lies in the fact that the number of artificially generated outliers is proportional to the volume of the enclosing sphere, which in turn is quadratically related to the dimension of the input dataset. This has for consequence of restricting the use of this method to only low-dimensional datasets, as the number of points required to provide a uniform coverage becomes intractable even for moderate dimensions.

In this paper, we present an alternative parameter selection strategy for the SVDD that addresses all the limitations exhibited by existing methods. Our method allows identifying SVDD parameters to produce contours that estimate accurately the distribution support of the target class. This is made possible by identifying some distinguishing features of parameter variations on the SVDD contours. Our method achieves state-of-the-art detection rates of abnormal patterns on synthetic and real-world datasets. Its entire inductive process relies exclusively on positive patterns, does not make use of negative instances and is applicable to high-dimensional datasets. Moreover, the method is designed specifically to achieve a high accuracy in presence of high proportion of outliers in the training set. The method also implements a novel overfitting index effectively preventing the selection of inadequate parameters.

Section 2 introduces the formulation of the SVDD optimization problem. Section 3.1 discusses the typical effects of variations of parameters σ and ρ on the SVDD decision boundaries, some distinguishing features of the contours transformations serve as a basis for our parameters selection method. Section 3.2 describes typical symptoms of overfitting in an SVDD model and proposes a simple index to prevent overfitting in the parameters selection. Section 5 details our new strategy for selecting parameters that yields a precise domain representation of any input target class of any size and dimension and an accurate identification of abnormal observations. It also describes our approach for minimizing the impact of outlier patterns in the training set on the detection accuracy of our method. Section 6 reports experimental results on synthetic and real-world datasets and evaluates the impact of the dataset size, dimension and noise on the capacity of our method to differentiate between normal and abnormal patterns.

2 Support Vector Domain Description

The SVDD is designed to characterize the support of the unknown distribution function of an input target class by computing a set of contours that rejects a controlled proportion ρ of patterns. These contours provide an estimate of a specific level set associated with the probability $1-\rho$ of the distribution function and allow unseen patterns to be classified as normal or abnormal.

2.1 Optimization problem

Given a set of N patterns $x \subseteq X$, where $X \subset \mathbb{R}^d$, and a nonlinear mapping ϕ from X to some high-dimensional nonlinear feature space Φ, we seek a hypersphere of center a and minimal radius R that encloses most data points and reject a proportion ρ of the less representative patterns:

$$\begin{aligned} &\min_{R^2,\varepsilon_i,a} R^2 + C\sum_i \varepsilon_i \\ &\text{s.t.} \begin{cases} \|\phi_i - a\|^2 \le R^2 + \varepsilon_i \\ \varepsilon_i \ge 0 \end{cases} \end{aligned} \tag{1}$$

Slack variables ε_i are added to the constraints to allow soft boundaries, and ϕ_i denotes the coordinate $\phi(x_i)$ of x_i in the feature space. Points associated with $\varepsilon_i > 0$ are excluded from the contours and penalized by a regularization constant C which controls the proportion ρ of points lying outside the hypersphere.

$$C = \frac{1}{\rho \cdot N} \tag{2}$$

The optimization problem can be converted into its primal Lagrangian form:

$$\begin{aligned} L_p\left(R^2, \varepsilon_i, a\right) &= R^2 + C\sum_i \varepsilon_i - \sum_i \alpha_i\left(R^2 + \varepsilon_i - \|\phi_i - a\|^2\right) - \sum_i \beta_i \varepsilon_i \\ s.t.\quad & \alpha_i \geq 0,\quad \beta_i \geq 0 \end{aligned} \tag{3}$$

where $\alpha_i \geq 0$ and $\beta_i \geq 0$ are Lagrange multipliers enforcing both constraints. The Karush-Kuhn-Tucker (KKT) optimality conditions [6] are obtained by setting to zero the partial derivatives of (3) with respect to R and ε_i, and are expressed as follows:

$$\begin{aligned} \alpha_i - C + \beta_i &= 0 \\ \beta_i \varepsilon_i &= 0 \\ \textstyle\sum_i \alpha_i &= 1 \\ \alpha_i\left(R^2 + \varepsilon_i - \|\phi(x_i) - a\|^2\right) &= 0 \end{aligned} \tag{4}$$

It follows from this last equation that the image of a point x_i with $\varepsilon_i > 0$ and $\alpha_i > 0$ lies outside the feature-space sphere. By substituting (4) into the primal Lagrangian [3] , we derive the Wolfe dual form:

$$\begin{aligned} L_d &= \sum_i \alpha_i \phi_i \cdot \phi_i - \sum_i \sum_j \alpha_i \alpha_j \phi_i \cdot \phi_j \\ s.t.\quad & \begin{cases} \sum_i \alpha_i = 1 \\ 0 \leq \alpha_i \leq C \cdot \end{cases} \end{aligned} \tag{5}$$

The dot product $\phi_i \cdot \phi_j$ is replaced by an appropriate Mercer [7] kernel $k(x_i, x_j)$, referred to as $k_{i,j}$ for notation simplicity, overcoming the explicit reference to ϕ_i of possible infinite dimension. The Gaussian kernel is used in this context; specificity of the cluster contours with a single parameter σ controlling the Gaussian kernel bandwidth.

$$k_{i,j} = e^{-\frac{1}{2\sigma^2}\left\|x_i - x_j\right\|^2} \tag{6}$$

The Lagrangian dual (5) is simplified by replacing the dot products $\phi_i \cdot \phi_j$ by the Gaussian kernel $k_{i,j}$:

$$\begin{aligned} L_d &= \sum_i \alpha_i k_{i,i} - \sum_i \sum_j \alpha_i \alpha_j k_{i,j} \\ s.t. &\begin{cases} \sum_i \alpha_i = 1 \\ 0 \le \alpha_i \le C \end{cases} \end{aligned} \tag{7}$$

2.2 Decision Function

The center a of the hypersphere is described as a linear combination of the feature-space vectors ϕ_i:

$$a = \sum_i \alpha_i \phi_i \tag{8}$$

The square Euclidian distance from an image ϕ_t of x_t to the sphere center a is defined as:

$$\begin{aligned} r^2(x_t) &= \left\|\phi_t - a\right\|^2 \\ &= k_{t,t} - 2\sum_i \alpha_i k_{i,t} + \sum_i \sum_j \alpha_i \alpha_j k_{i,j} \end{aligned} \tag{9}$$

The decision surface is defined as the implicit surface $\{x : d(x) = 0\}$ of the function $d(x)$ described below, and evaluates the relative position from the image $\phi(x)$ to the surface of the hypersphere. The function $d(x)$ classifies a point x inside the contours if $d(x) < 0$, on its surface if $d(x) = 0$ and outside otherwise.

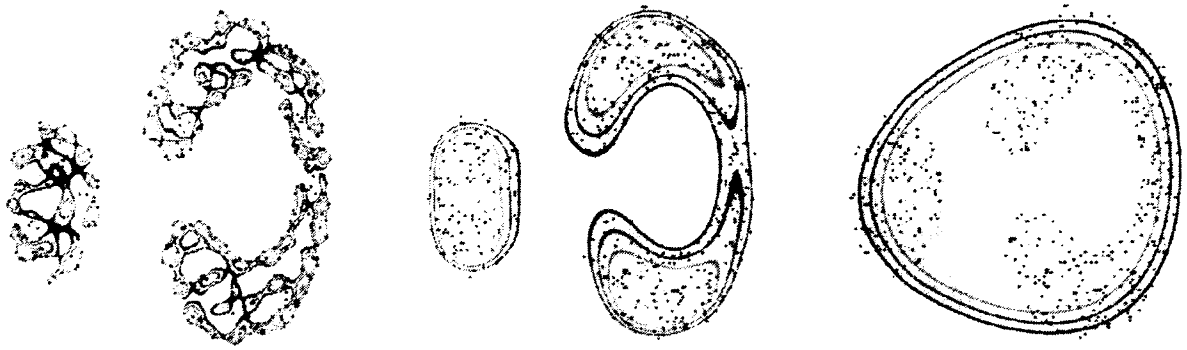
$$d(x_i) = O_S - O_i \quad \text{where} \quad O_i = \sum_j \alpha_j k_{i,j}$$

$$O_S = \tfrac{1}{2}\left(O_i + O_j\right) \quad \text{where} \quad \begin{cases} i \leftarrow \arg\min_k O_k & s.t. \quad \alpha_k < C \\ j \leftarrow \arg\max_k O_k & s.t. \quad \alpha_k > 0 \end{cases} \qquad (10)$$

3 Effect of hyperparameters and prevention of overfitting

3.1 Gaussian kernel bandwidth σ and rejection rate ρ

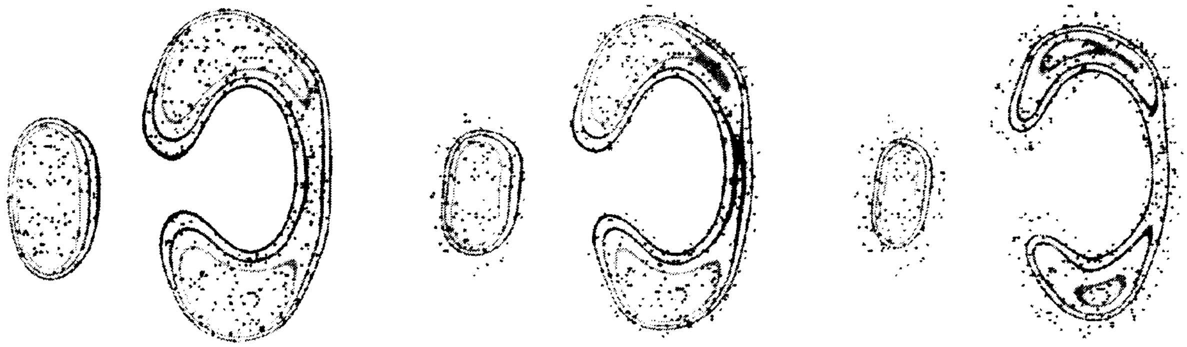
The Gaussian kernel bandwidth $\sigma \in \mathbb{R}^+$ controls the complexity of the decision surface generated by a trained SVDD model Θ ; low values of σ produces high contours complexity and high values result in low complexity.



Figures 1a, 1b and 1c (from left to right) - Decision surfaces resulting from the choice of different Gaussien kernel bandwidths and a fixed rejection rate $\rho = 10\%$ (Fig. 1a) $\sigma = 0.005$, (Fig. 1b) $\sigma = 0.1$, (Fig. 1c) $\sigma = 1.0$.

It is clear that the choice of an infinitesimal value of σ results in the contours fragmenting into N individual disjoint contours, each one enclosing an individual pattern of the dataset. Conversely, choosing an arbitrary large bandwidth produces a single connected contour of round shape enclosing training patterns. Intuitively, an appropriate choice of σ produces a compact representation as observed in Figure 1 (b) of the target class support which excludes any superfluous space, while avoiding overfitting (Figure 1 (a)) and over simplistic representation (Figure 1 (c)). Section 4 presents an overfitting index allowing the identification of parameters producing SVDD models exhibiting symptoms of overfitting.

Now let us look at the rejection rate $\rho \in \mathbb{R}^+$ which controls the level of tolerance of a SVDD model to outliers as it constrains the percentage of points rejected by the decision surface, and bounds the Lagrange multipliers α as $0 \leq \alpha \leq C$ with $C = 1/\rho N$. Figure 2 (a), (b) and (c) illustrate the effect of the rejection rate ρ, respectively set to 0.2%, 25% and 50% for a constant kernel bandwidth $\sigma = 0.1$.



Figures 2a, 2b and 2c (from left to right) - Decision surfaces resulting from the choice of different rejection rates and a fixed kernel bandwidth $\sigma = 0.1$ (Fig. 2a) $\rho = 0.2\%$, (Fig. 2b) $\rho = 25\%$, (Fig. 2c) $\rho = 50\%$.

To the extreme, the choice of the minimal rejection rate $\rho = \frac{1}{N}$ would result in contours enclosing $N-2$ training points and only two points lying on their surface acting as active support vectors. The decision surface then would provide an estimate of the distribution support defined by only two patterns and assumes the complete absence of noise in the training set. On the other hand, the choice of a maximal rejection rate $\rho = \frac{N-1}{N}$ would result in the exclusion of $N-1$ data points, where the SVDD distribution support estimate would converge toward a Parzen window estimator, approximating the underlying density distribution of the training set as a weighted sum of $N-1$ Gaussian kernel functions of bandwidth σ each centered on each individual point $x \in X$.

3.2 Prevention of overfitting

Overfitting is a crucial concept in machine learning,. It characterizes a learning method tendency to adapt itself to random features of a dataset instead of its underlying structure and is a symptom of an excessive model complexity in respect to the inner specificity of the dataset. While overfitting typically occurs for smaller values of σ in the SVDD context, the risk of overfitting is also augmented by the choice of a too low rejection rate ρ, increasing the sensibility of the model to noise.

As supported by Figures 1 (a), (b) and (c), overfitting is observed by training a sequence of SVDD models on a same dataset while increasing the complexity (decreasing σ) for a constant ρ. At some value of σ, the model yields a significant and sudden rise in the number of support vectors lying precisely on the contour surface. This effect is symptomatic of an overly complex representation, where the contours are forced to pass through too many data points of the dataset.

Given that the number of support vectors is directly related to the model complexity, we define and make use of the following two concepts, $\rho_{\exp}(\Theta)$ and $\rho_{obs}(\Theta)$, respectively as the expected and observed proportions of support vectors of a trained model Θ as

$$\begin{aligned}&\rho_{\exp}(\Theta)=\rho\\&\rho_{obs}(\Theta)=\tfrac{1}{N}\sum_{i=1\cdots N}\mathrm{I}(\alpha_i>0)\end{aligned}\tag{11}$$

The Sequential Minimal Optimization (SMO) algorithm [8] that is used for training the SVDD model initializes Lagrange multipliers such that the expected proportion $\rho_{\exp}$ of support vectors equals $\rho'=\min\left(\tfrac{N-1}{N},\max\left(\tfrac{1}{N},\rho\right)\right)$, with the proportion of support vectors ρ_{obs} remaining approximately constant ($\rho_{obs}\approx\rho_{\exp}$) throughout the optimization process.

A spontaneous increase in the number of SV lying exactly on the decision surface results in an increase in the gap between $\rho_{obs}(\Theta)$ and $\rho_{\exp}(\Theta)$, and provides a means of detecting symptoms of overfitting in a SVDD model. This indicator can be formalized by defining $\Lambda(\Theta)$ as the difference between the expected and observed proportions of support vectors $\rho_{\exp}(\Theta)$ and $\rho_{obs}(\Theta)$. A high value of $\Lambda(\Theta)$ will then allow to detect symptoms of overfitting.

$$\begin{aligned}\Lambda(\Theta) &= \rho_{obs}(\Theta) - \rho_{\exp}(\Theta)\\ &= \tfrac{1}{N}\sum_{i=1\cdots N} \mathrm{I}(\alpha_i > 0) - \rho \\ &\begin{cases} \Lambda(\Theta) \geq \tau_{ofit} \Rightarrow \Theta \text{ overfits} \\ \Lambda(\Theta) < \tau_{ofit} \Rightarrow \Theta \text{ is admissible} \end{cases}\end{aligned} \qquad (12)$$

Equation (12) provides a simple and practical criterion which allows detecting and preventing the selection of parameters leading to overfitting in a SVDD model Θ. In fact, a model Θ can be considered at higher risk of overfitting if $\Lambda(\Theta) \geq \tau_{ofit}$ for a typically small value of $\tau_{ofit} = 1\%$. The SVDD models displayed in Figures 1 (a) to (c) exhibit values of $\Lambda(\Theta)$ of 27%, 0.5% and 0.4%, for kernel bandwidth values of 0.005, 0.1 and 1.0. This suggests that only the model in (a) with a small bandwidth suffers from overfitting.

Table 1 states the asymptotic relation between $\Lambda(\Theta)$ and σ; an infinitesimally small value of σ leads to a maximal value of $\Lambda(\Theta)$ interpreted as a higher risk of overfitting, a large value of σ leads to a low risk of overfitting.

Table 1 - Asymptotic limits of $\Lambda(\Theta)$ in relation to σ.

Value of	**Limit value of Λ**
$\sigma \to 0$	$\lim_{\sigma\to 0} \Lambda(\Theta) = 1 - \rho$
$\sigma \to \infty$	$\lim_{\sigma\to\infty} \Lambda(\Theta) = \tfrac{1}{N} - \rho$

This criterion is used to constrain the admissible hyperparameters search space and thereby minimize the computational cost of the parameter selection process presented in the next sections. Figure 3 illustrates the typical relation between $\Lambda(\Theta)$ and σ for a fixed rejection rate ρ and is calculated on a synthetic dataset using a tolerance factor $\tau_{ofit} = 1\%$. It illustrates the typical trend of $\Lambda(\Theta)$ over σ, showing an exponential decay in the overfitting domain $\sigma_{overfitting} \in \left[0, \sigma_{ofit}\right]$ and stabilizing to a small value in the admissible domain $\sigma_{admissible} \in \left(\sigma_{ofit}, \infty\right]$

.

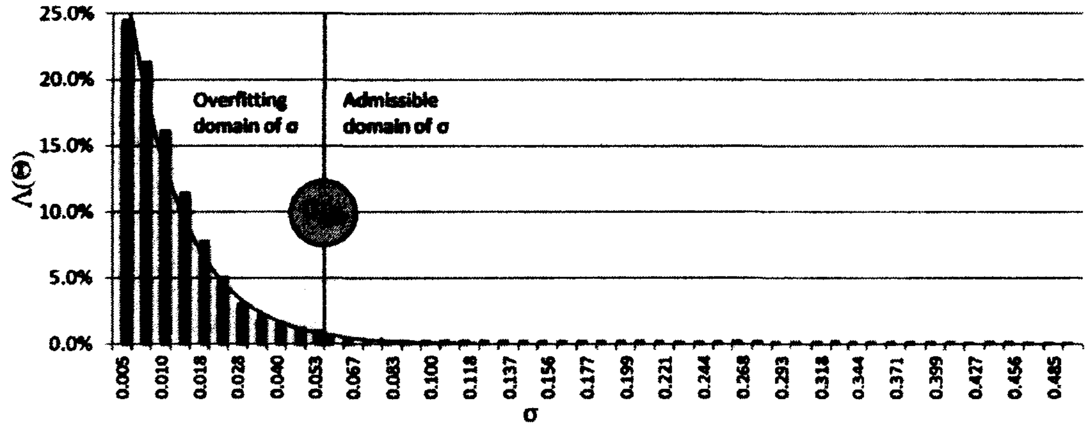


Figure 3 - Relation between $\Lambda(\Theta)$ and σ.

Figure 4 extends this demonstration to both parameters (σ, ρ) and displays the contour map associated with $\Lambda(\Theta)$ in relation to (σ, ρ), calculated on the same dataset used to generate Figure 3. It reaffirms that the risk of overfitting is higher for small values of σ and ρ. The *admissible* hyperparameters space is represented by the blue area of the contour map, while the *overfitting* parameter space is identified as the remaining portion of the map.

As well as being simple from an implementation stand-point, the proposed overfitting criterion makes it possible to reduce dramatically the hyperparameters search space and does not require any prior information about the datasets such as its size, dimension and inner complexity, and is independent from the kernel function used.

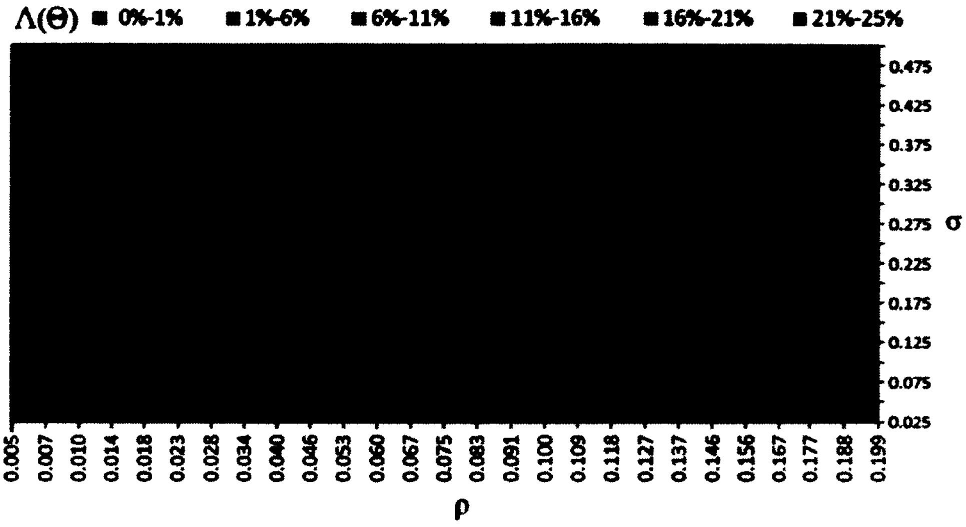


Figure 4 - Relation between $\Lambda(\Theta)$ function and (σ, ρ).

Finally, we adopted a data driven approach to automatically determine the threshold τ_{ofit}. As detailed in Section 4.3, the threshold τ_{ofit} is calculated as the upper bound of a 97.5% confidence interval of the values of $\Lambda(\Theta)$ evaluated while performing a grid-search on the parameter space (defined in Section 4.3). Using the estimated average μ_Λ and standard deviation σ_Λ of $\Lambda(\Theta)$, we have

$$\tau_{ofit} = \mu_\Lambda + z \cdot \sigma_\Lambda \quad \text{for } z = 1.96 \tag{13}$$

Eq. (13) provides a simple way to reject 2.5% of hyperparameters leading to the highest risk of overfitting. This approach in turn prevents rare cases where all combinations of hyperparameters are rejected due to an overly restrictive choice of τ_{ofit}.

The following section describes our strategy for selecting SVDD hyperparameters within the admissible search space bounded by $\Lambda(\Theta) < \tau_{ofit}$.

4 Characterization of SVDD contours

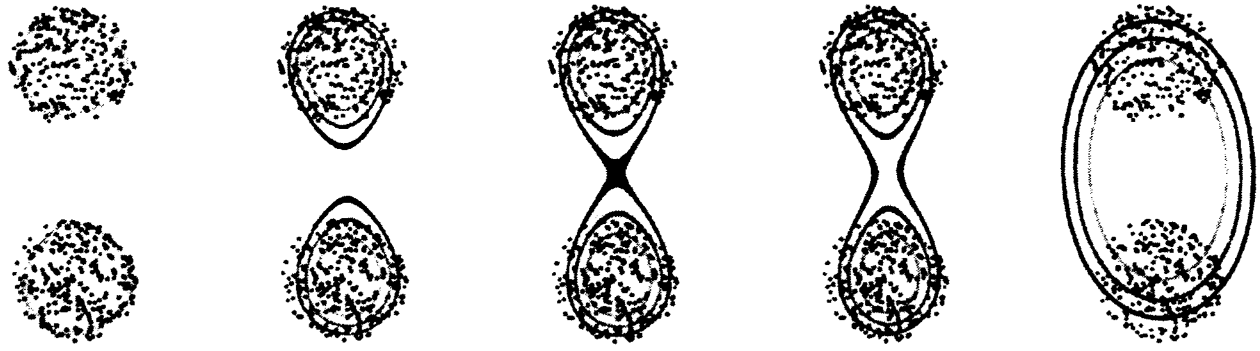
We characterize the effects of parameter variations on SVDD contours that will set ground to our strategy for identifying a SVDD contour configuration that provides accurate estimates of the distribution supports of any target classes for novelty detection.

4.1 Classification of steady and transient states

A model Θ trained on any dataset with an infinitesimal σ will produce a set of distinct contours each one enclosing a distinct data point. Increasing in small steps the kernel bandwidth σ to any arbitrary large value will result in a sequence of fusions of adjacent contours merging together, which in turn will combine to ultimately form a single connected contour grouping all of data points into a single cluster.

This sequence of contours fusions reduces the overall number and complexity of resulting contours and is analogous to small water drops merging together until forming a single larger drop. This phenomenon can be interpreted as a series of transformations between *steady states* (single drops) separated by *transient states* (drops merging together). We call *steady state* any contour configurations which exhibits a local stability in their number and shape in regard to any small variation of both parameters (σ, ρ). We use the term *transient state* to characterize contour configuration that exhibits local distortions (fusion) for a small variation of parameters. Transition between adjacent steady states occurs when two or more distinct contours merge together, resulting in a less compact grouping.

Figures 5 (a) to (e) illustrate the typical transformation of a SVDD decision surface trained with increasing values of σ on a dataset composed of two distinct round-shaped clusters of points. Decreasing the complexity by increasing σ from $\sigma_{S_1} = 0.2$ to $\sigma_{S_2} = 0.7$ produces the fusion of the two groupings S_1 into the new, simpler and less compact steady state S_2, separated by three transient states that we refer to as *attraction* A_1, *junction* J_1 and *consolidation* C_1 states.



S_1 - STEADY $\sigma_{S_1} = 0.2$

A_1 - ATTRACTION $\sigma_{A_1} = 0.25$

J_1 - JUNCTION $\sigma_{J_1} = 0.3$

C_1 - CONSOLIDATION $\sigma_{C_1} = 0.35$

S_2 - STEADY $\sigma_{S_2} = 0.7$

Figures 5 (a to e) (from left to right) - Steady states S_1, S_2 and transient states A_1, J_1, C_1 resulting from increasing values of σ.

We observe an *attraction state* A_1 when increasing the kernel bandwidth from $\sigma_{S_1} = 0.2$ to $\sigma_{A_1} = 0.25$, producing a local attraction between the two adjacent contours of the two distinct contours seen at S_1, eventually leading to the contact of the two disconnected contours at $\sigma_{J_1} = 0.3$ identified as a *junction state* J_1. This state is then followed by a *consolidation state* where the newly connected contours consolidate into a new *steady state* S_2 at $\sigma_{S_2} = 0.7$, displaying a single elliptical contour of reduced complexity.

We now define the function $\Omega(\Theta)$ that allows identifying each of distinguishing feature state. This function measures the average distance from points to the decision surface.

$$\begin{gathered}\Omega(\Theta) = \tfrac{1}{N}\sum_i d(x_i) = O_s(\Theta) - \bar{O}(\Theta) \\ \text{with } \bar{O}(\Theta) = \tfrac{1}{N}\sum_i O_i\end{gathered} \tag{14}$$

As observed in Figures 5 (a) and (b), transiting from a steady state S_1 to a junction state J_1 produces a local inflation of the contours attracting to each other at A_1, resulting in the decrease of the values of $d(x)$ of all points near the contour. Conversely, a consolidation state C_1 following any contact of distinct contours at J_1, causes a local increase in $d(x)$ in the locality of the inflating contour until a new steady state S_2 of reduced complexity is reached. In a new steady state, the values of $d(x)$ stabilize temporarily until the system transits into another sequence of transient states. These transformations produce some variations in the monotonicity of the function $\Omega(\Theta)$, which can be measured to identify each specific state resulting from the choice of any parameters (σ, ρ). As summarized in Table 2, the classification of any state can be achieved based on the signs of the first and second derivatives of $\Omega(\Theta)$.

Table 2 - Effects of stready and transient states on $\Omega(\Theta)$ and first and second derivatives.

	(S)STEADY	(F)FUSION	(J)JUNCTION	(C)CONSOLIDATION
$\Omega(\Theta)$	max	decreasing	min	increasing
$\partial/\partial\sigma\Omega(\Theta)$	0	<0	0	>0
$\partial^2/\partial\sigma^2\Omega(\Theta)$	<0	n/a	>0	n/a

Figure 6 illustrate the typical behavior of the function $\Omega(\Theta)$ calculated on the dataset illustrated in Figures 5 (a) to (e). The function $\Omega(\Theta)$ shows three local maxima each identifying a distinct steady state, identifying in turn different SVDD representations of the same dataset with different levels of complexity and numbers of clusters.

The steady states S_0, S_1, S_2 and junction states J_0, J_1 are respectively identified as local maxima and minima of $\Omega(\Theta)$; the fusion and consolidation states F_0, F_1 and C_0, C_1 are correspondingly identified as functionally decreasing and increasing sections of $\Omega(\Theta)$. The left-hand red portion of function $\Omega(\Theta)$ identifies values $\sigma < \sigma_{\min}$ associated with a higher risk of overfitting according to $\Lambda(\Theta)$. The right-hand gray section identifies the over-generalizing domain $\sigma > \sigma_{\max}$, lower bounded by the value $\sigma_{\max}$ associated to the least complex steady state. Any model trained with $\sigma > \sigma_{\max}$ will produce a contour configuration similar to the one achieved at $\sigma > \sigma_{\max}$ with identical patterns groupings and numbers of clusters. The admissible domain $\sigma \in (\sigma_{\min}, \sigma_{\max}]$ is represented as the middle blue section bounded between the overfitting and over-generalizing domains of σ.

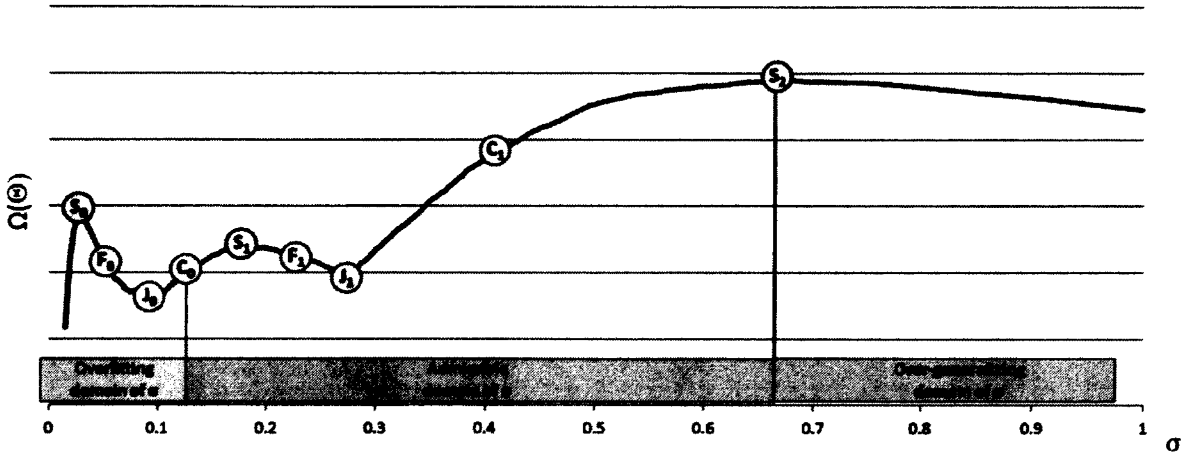


Figure 6 - Characterization of each state of the function $\Omega(\Theta)$ in relation to σ with identification of the overfitting, admissible and over-generalizing values of σ.

We restrict the set of admissible contours configurations to the subset of steady states identifiable within the admissible domain. We identify the best configuration as the one with highest complexity (lowest σ), as it will provide the most compact representation of the data support excluding any superfluous space, and will distinguish accurately all natural grouping without suffering from overfitting.

Until now, we centered our demonstration around the identification of the best value of σ using a fixed rejection rate ρ to ease the definition of the key concepts of this method. The parameter selection method we propose is a generalization of these concepts to both parameters (σ, ρ), transforming the 2D representation of $\Omega(\Theta)$ into the following 3D representation of the same function shown below by varying (σ, ρ) simultaneously.

The set of parameters (σ, ρ) are finally selected as the one associated to the local maxima of $\Omega(\Theta)$ located within the admissible domain of (σ, ρ) and with minimal value of σ.

This method allows identifying adequate hyperparameters independently of the dataset size, dimension and inner structure, does not require any expert knowledge relative to the dataset, and prevents actively overfitting in the selected solution. The prevalence given to the steady state with minimal σ derives from the fact that it will guarantee producing compact SVDD contours with minimal inner surface, which in turn ensures the minimal misclassification rate on negative instances of the target class in comparison to all other admissible steady states.

4.2 Improving the robustness on noisy patterns

The goal in this section is to improve the robustness of the measure $\Omega(\Theta)$ in presence of noise in the dataset, which have the effect of flattening the function $\Omega(\Theta)$ and make more difficult the detection of local maxima distinguishing steady states.

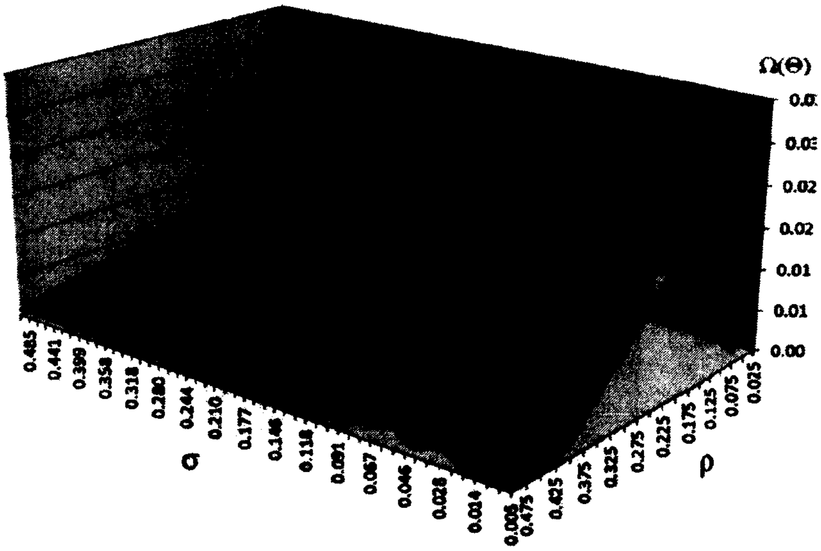


Figure 7 - 3D representation of $\Omega(\Theta)$ in relation to hyperparameters (σ, ρ).

The variations in monotonicity of $\Omega(\Theta)$ result primarily from local changes in values of $d(x)$ for points x located in the zone where the contours shape change during transient states. We observed that a high proportion of noise in the training set has the effect of flattening the ridges and valleys of the function $\Omega(\Theta)$ and may prevent in some cases from detecting some steady states. The sensitivity of $\Omega(\Theta)$ to noise results essentially from the fact that it computes the arithmetic mean of $d(x)$ over all the observations of the dataset and attributes equal weights to normal and noisy patterns which alter the monotonicity of $\Omega(\Theta)$. We developed a strategy to reduce the impact of these noisy patterns on (14), by weighting the contribution of each point according to their relative distance to the surface contour, reducing the weights of noisy patterns located far away from the natural grouping and their enclosing contours.

As variations in the monotonicity of $\Omega(\Theta)$ are a direct consequence of local deviations of $d(x)$ in the neighborhood of the contours, we reduce the sensitivity of $\Omega(\Theta)$ to outliers by assigning a greater importance to variations of $d(x)$ within a virtual margin centered around the contours and lesser weights to distant observations, transforming the function $\Omega(\Theta)$ into $\Pi(\Theta)$. The intuition underlying the design of the weighted average relies on the hypothesis that points sufficiently distant from the SVDD contours contain a negligible amount of information that is relevant for assessing transition of states and thus should have an insignificant effect on the monotonicity of $\Pi(\Theta)$.

$$\Pi(\Theta) = \frac{\sum_i \omega(x_i)\cdot d(x_i)}{\sum_i \omega(x_i)} \tag{15}$$

The contours are defined as the isosurface of $d(x)$ in primal space describing the surface of the enclosing hypersphere of radius O_s centered on a described in feature space. We thus define the virtual margin as two hyperspheres each centered on a, with radii set to $d_+ = b$ and $d_- = -b$, enclosing any data point x if $|d(x)| \leq b$.

The choice of b is critical, as it controls the margin thickness $2b$ that is responsible for the robustness of our method to noise, as it will be used to assign positive or null weights $\omega(x)$ to each point x. Our weighting scheme requires that $\omega(x) > 0$ for points enclosed within the margin ($|d(x)| < b$) and $w(x) = 0$ for points excluded from the margin ($|d(x)| \geq b$).

Before providing further details on the margin thickness calculation, we will characterize domain characteristics of O_s, $O(x)$ and then $d(x)$. We will then use this information to apply some transformations to the margin definition to make its thickness invariant to variations of parameters values (σ, ρ). In other words, we want the margin thickness to remain of constant size for changing hyperparameters values.

Based on the domain characteristics of the Lagrange multipliers α and Gaussian kernel k, the function $O(x_i) = \sum_j \alpha_j k(x_i, x_j)$ can be bounded according to (16).

$$0 \leq O(x_i) \leq \frac{1}{\rho} \quad \text{with} \quad \begin{cases} 0 \leq \alpha_i \leq C = \frac{1}{\rho N} \\ 0 \leq k(x_i, x_j) \leq 1 \end{cases} \tag{16}$$

The domain of $O(x)$ is thus independent of the value of σ but is dependant of ρ. The hypersphere radius O_s detailed in (10) is the distance from a virtual point lying on the isosurface of $d(x)$ to the center a, allowing to define the domain of O_s as follows:

$$0 \leq O_s \leq \frac{1}{\rho} \quad \text{with} \quad \begin{cases} O_s = O(x_s) \\ x_s \text{ s.t. } d(x_s) = 0 \text{ and } 0 < \alpha_s < C \end{cases} \tag{17}$$

Based on (16) and (17), the domain of $d(x)$ is consequently bounded as:

$$-\frac{1}{\rho} \leq d(x) \leq \frac{1}{\rho} \quad \text{with} \quad d(x) = O_s - O(x) \tag{18}$$

The decision function values $d(x)$ varies within the interval $\left[-\frac{1}{\rho},\frac{1}{\rho}\right]$. We now define $d'(x)$ as the transformation of $d(x)$ scaled by a factor ρ, now domain invariant ($d'(x)\in[-1,1]$) in relation to the parameters values. We will use this new definition of relative distance to the hypersphere surface to define our parameter invariant margin. Note that the sign of $d'(x)$ allows classifying a point as within, outside or on the decision surface similarly to $d(x)$.

$$d'(x)=\rho(O_s-O(x)) \quad \text{with} \quad -1\leq d'(x)\leq 1 \tag{19}$$

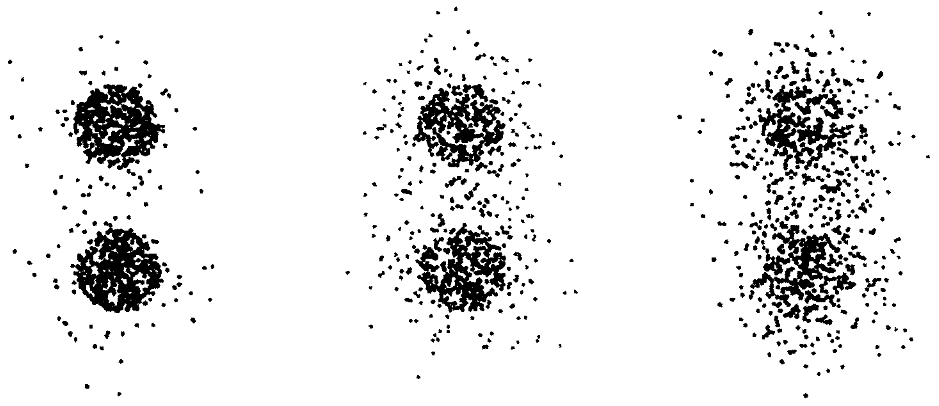
We define the thickness parameters b using the new domain invariant function $d'(x)$ as the average value of $d'(x)$, for any point enclosed within the contour $d'(x)=0$. We will ultimately use this definition to calculate the new invariant margin.

$$\begin{aligned} &b=-\tfrac{1}{N_{in}}\sum_{x}d'(x) \\ &\text{with} \quad x\in X_{in},\ N_{in}=|X_{in}| \ \text{ s.t. } \ d'(x)\leq 0 \end{aligned} \tag{20}$$

We choose b as the average distance to contours $d'(x)$ from points located within the contours $d'(x)\leq 0$, as it defines a dynamic reference distance that adjusts to any particular problem. The negative sign forces b to be positive.

Finally, we provide a formal definition of $\omega(x)$ based on the newly defined invariant margin, which assigns positive weights to points x located within the margin with weight value equal to one on the decision surface ($d'(x)=0$) , and decreasing toward zero the closer it gets to the margin boundaries ($|d'(x)|=b$). All points excluded from the margin are assigned zero weights.

$$\omega(x)=\max\left(0,1-\frac{|d'(x)|}{b}\right) \tag{21}$$



$\rho = 30\%$ $p_{noise} = 25\%$ $\rho = 55\%$ $p_{noise} = 50\%$ $\rho = 80\%$ $p_{noise} = 75\%$

Figures 8a, 8b and 8c (from left to right) - Magnitude of weights in relation to increasing proportion of noise p_{noise}. Red points identify weights such that $\omega(x) = 0$ and blue points $\omega(x) > 0$. All models are trained with $\sigma = 0.1$.

Figures 8 (a) to (c) illustrate this weighting strategy by displaying the magnitude of individual weights $\omega(x)$ assigned to each point x of a dataset. Blue data points are associated with positive weights $\omega(x) > 0$ and red data points with zero weights. We generated the synthetic dataset and then applied an additive Gaussian noise of normal distribution $\mathrm{N}(\mu = x, \sigma = 0.4)$ centered on each point to a varying proportion p_{noise} of random observations.

As displayed in Figure 8, the weighting function (21) successfully assigns zero weights to noisy patterns and positive weights to normal observations located within the two clusters, even for high proportion of noise. The weighting scheme dramatically mitigates the influence of outliers in the dataset and allows the manifestation of local maxima in function $\Pi(\Theta)$ that is crucial for detecting steady states.

Note that on artificially generated datasets (not including artificial noise), the 3D representations of functions $\Omega(\Theta)$ and $\Pi(\Theta)$ (in respect to both parameters) exhibit the same local maxima at the same parameters coordinates. When increasing the level of artificial noise, some key local maxima from function $\Omega(\Theta)$ disappear while remaining clearly detectable in function $\Pi(\Theta)$, confirming its improved robustness to noise.

4.3 Selection of parameters

This section describes our actual strategy for selecting SVDD hyperparameters, based on the steady state concepts and function $\Pi(\Theta)$ presented earlier.

The proposed algorithm starts with a normalization step of the input dataset, centering X on its median $\mu_{\frac{1}{2}}(X)$ and scaling it in such a way that the high proportion $\tau_d \approx 95\%$ of points $x \in X$ are bounded within a constant interval $[-1,1]$. This transformation (described in [22]) transforms X into X' and makes it invariant to translation and affine scaling, and most importantly reduces the upper bound search space of parameter σ as a result of the constrained scaling of X'.

$$X' \leftarrow \frac{(X-c)}{d} \quad \text{with} \quad \begin{cases} c = \mu_{\frac{1}{2}}(X) \\ d \ \text{ s.t. } \Pr\left[|X-c| \leq d\right] \approx \tau_d \end{cases} \tag{22}$$

The second step evaluates the values of $\Pi(\Theta)$ for different combinations of parameters within the intervals $\sigma \in [0.005, 0.5]$ and $\rho \in [2.5\%, 50\%]$ and such as $\Lambda(\Theta) < \tau_{ofit}$. Note that we choose as lower and upper bound of σ values leading to extremely highly complex ($\sigma = 0.005$) and simple ($\sigma = 0.5$) contours that allows adapting to the inner complexity of most datasets encountered.

The grid of parameters explored has n_ρ constant step over ρ, and n_σ dynamic steps over σ. Each combination of parameters (σ_i, ρ_j) on the grid are calculated as

$$\begin{aligned}
&\sigma_i = \sigma_{\min} + \frac{\sum_{j=1\cdots i}\sqrt{i-1}}{\sum_{k=1\cdots n_\sigma}\sqrt{k-1}}\left(\sigma_{\max} - \sigma_{\min}\right) \\
&\rho_j = \rho_{\min} + j\tfrac{1}{n_\rho}\left(\rho_{\max} - \rho_{\min}\right) \\
&\textit{with} \quad \Delta\rho_i = \tfrac{1}{n_\rho}\left(\rho_{\max} - \rho_{\min}\right) \quad \textit{and} \quad \Delta\sigma_i = \frac{\sum_{j=1\cdots i}\sqrt{i-1}}{\sum_{k=1\cdots n_\sigma}\sqrt{k-1}}\left(\sigma_{\max} - \sigma_{\min}\right)
\end{aligned} \tag{23}$$

The use of dynamic increments on σ_i increases the grid resolution for smaller values of σ and produces increasing steps for larger values of σ, thereby allowing exploring more variations of complex configurations and improving the ability to identify complex cluster configurations. The values of $\Pi(\Theta)$ are stored in a matrix M_Π of size $n_\sigma \times n_\rho$. Figure 7 is a 3D representation of a typical matrix M_Π.

The third step identifies the maximal value of every row of $M_\Pi(i,\cdot)$ and stores the results into a table R_Π of size n_σ. The table allows a simpler interpretation that $M_\Pi(i,\cdot)$ as it allow displaying in 2D all local maxima allowing in turn to identify steady states of varying complexity. This process produces a curve analogous to a maximal energy path over $\Pi(\Theta)$ passing through each of the local maxima.

Also note that each combination of parameter is tested for overfitting using $\Lambda(\Theta)$ and all steady states associated to inadmissible parameters are discarded for the selection process (represented in red in Figure 9). All admissible sets of parameters associated with their respective steady states (identified with green dots in Figure 9) are then sorted in increasing order according to their individual σ. The set of parameters with minimum σ is then retained as the winning one producing the most compact representation of the input dataset.

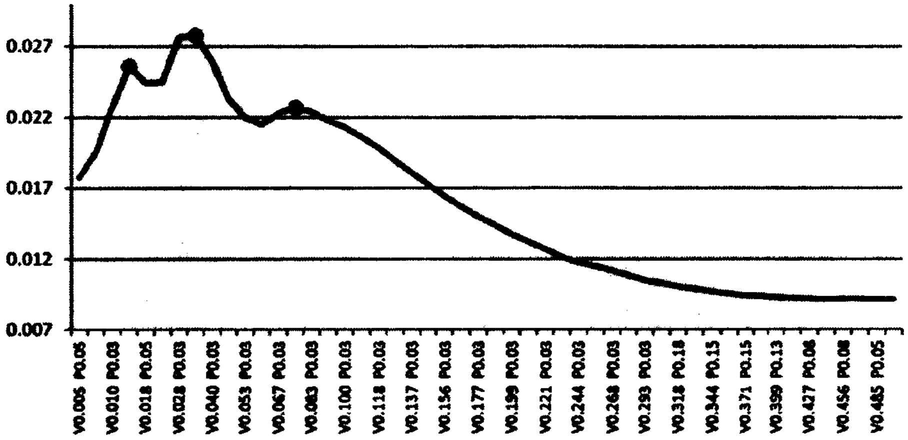


Figure 9 - Visualisation of R_Π as a simplified representation of M_Π in relation to hyperparameters (σ, ρ). Note that V and P stand for σ and ρ.

Analysis of the Figure 9 is very straightforward as it allows us to visualize the simultaneous effects of both hyperparameters (σ, ρ) on function $\Pi(\Theta)$ and to identify all admissible representations of the target class, independently of the nature of the input target class. The admissible hyperparameters space is greatly reduced in the case presented above, to three combinations of parameters producing representations of different complexity. This yields to a completely automatic selection strategy which is entirely data-driven and does not rely on any input parameters aside from the input dataset.

5 Experiments and results

We evaluate in this section the capacity of the proposed method to select parameters that produce SVDD models that distinguishes accurately normal patterns from abnormal ones. We also measure the impact of dataset sizes, dimensions, varying complexities and degrees of noise on the accuracy of the proposed method.

Section 5.1 looks at the impact of outliers in the dataset on the accuracy of our method. We developed a method to allow quantifying the impact of varying proportions of noise on the accuracy of models trained with our method to discriminate normal from abnormal patterns. This method seeks to determining the critical proportion of noise at which our method fails to estimate the support of the dataset, allowing in turn to quantify its robustness. Section 5.2 evaluates the accuracy of the proposed method on real-world datasets of varying sizes and dimensions.

5.1 Experiments on noisy synthetic datasets

To allow generating 2D datasets composed of two classes of patterns, normal and abnormal observations, we used bitmap images to describe the domains of these two mutually exclusive classes. This representation provides an exact description of the distribution support of the target class and allows creating and controlling the proportion of outliers to be excluded from the SVDD contours. This results in the creation of positive and negative instances of the target class, and allows evaluating with precision the ability generalization properties of our method at producing compact contours enclosing the theoretical domain of the normal patterns while excluding accurately the abnormal ones.

5.1.1 Generation of synthetic datasets

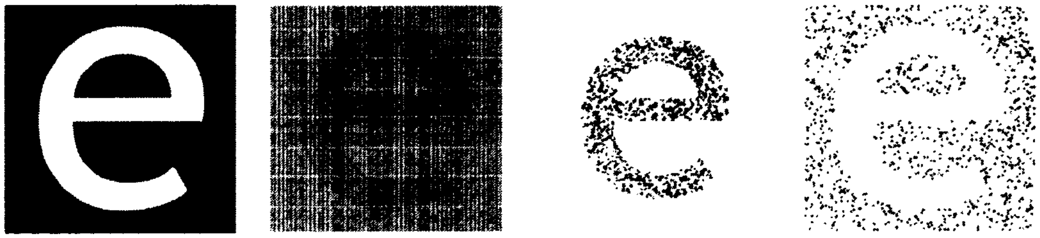
This allows generating 2D datasets of arbitrary size and shape complexity based on the discrete uniform probability distribution functions described by the white and black pixels in a square monochrome bitmap image file. This probability distribution representation has the advantage of providing an intuitive and accurate representation of both normal and the outlier domains relative to each dataset analyzed.

Each white and black pixel coordinate are converted into a data point and respectively added to X_{in} and X_{out}. The 2D coordinates of each points are extracted as the pixel coordinates in the bitmap matrix. All patterns in X_{in} and X_{out} are then centered on the median $\mu_{\frac{1}{2}}(X_{in})$ and each coordinate scaled in such a way that a proportion $\tau_d \approx 95\%$ of points $x \in X_{in}$ are bounded within an interval $[-1,1]$. The scaling is thus distinct for each coordinate and is such as s_1 for $\Pr\left[\left|X_{in}(1) - \mu_{\frac{1}{2}}(X_{in}(1))\right| \leq s_1\right] \approx \tau_d$ for the coordinate 1 and the same principle is applied for the second coordinate. This normalization process transforms X_{in} and X_{out} into X'_{in} and X'_{out}.

$$X'_{in} \leftarrow \left(X_{in} - \mu_{\frac{1}{2}}(X_{in})\right) \cdot \begin{bmatrix} \frac{1}{s_1} & 0 \\ 0 & \frac{1}{s_2} \end{bmatrix}$$

$$X'_{out} \leftarrow \left(X_{out} - \mu_{\frac{1}{2}}(X_{in})\right) \cdot \begin{bmatrix} \frac{1}{s_1} & 0 \\ 0 & \frac{1}{s_2} \end{bmatrix} \qquad (24)$$

Each of the two normal and abnormal sets X'_{in} and X'_{out} are then partitioned into a training and a validation set $X_{in}^{train}, X_{in}^{test} \subset X'_{in}$ and $X_{out}^{train}, X_{out}^{test} \subset X'_{out}$.

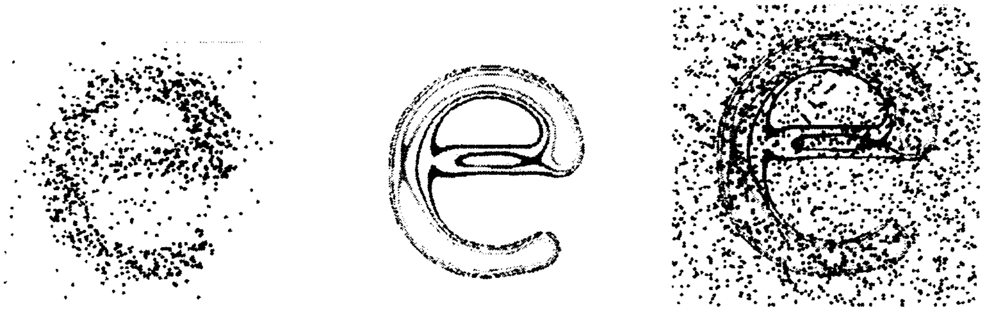


Figures 10 (a to d) (from left to right) - (Fig. 10a) Bitmap file representing the distributions associated with the normal and abnormal distribution supports. (Fig. 10b) Datasets X'_{in} and X'_{out} extracted based on the bitmap file. (Fig. 10c and 10d) Resulting normal and abnormal samples X_{in}^{train} and X_{out}^{train}.

5.1.2 Measuring the impact of noise

Using the process presented in Section 5.1.1, two datasets X_{in}^{train} and X_{in}^{test} of normal patterns and a dataset X_{out}^{test} of abnormal observations are sampled from X'_{in} and X'_{out}. The additive noise is then applied to a proportion p_{noise} of randomly sampled points $x \in X_{in}^{train}$, such as $x' = x + x_{noise}$ with $x_{noise} \sim \mathrm{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_{noise} & 0 \\ 0 & \sigma_{noise} \end{bmatrix}\right)$ and with σ_{noise} controlling the amplitude of the perturbations.

The artificial perturbations illustrated in Figure 11 (a), serves to assess the impact of the presence of noise or outliers in the training set X_{in}^{train} on the ability of our method to select parameters $\left(\sigma_{auto}, \rho_{auto}\right)$ that produce a SVDD model Θ_{auto} that estimates accurately the support of X'_{in}, and distinguish accurately normal patterns X_{in}^{test} from outliers X_{out}^{test}.



Figures 11 (a to c) (from left to right): (Fig. 11a) Dataset X_{in}^{train} with a proportion of noise $p_{noise} = 15\%$ with $\sigma_{noise} = 0.4$. (Fig. 11b) SVDD contours resulting from parameters selected by our method. (Fig. 11c) Validation of the SVDD contours on datasets X_{in}^{test} and X_{out}^{test} (blue and red data points).

The type-I and type-II error rates E_{in} and E_{out} on validation sets X_{in}^{test} and X_{out}^{test} of sizes N_{in}^{test} and N_{out}^{test} are calculated as

$$\begin{aligned} E_{in} &= \frac{1}{N_{in}^{test}} \sum_{x \in X_{in}^{test}} \mathrm{I}\left(d(x) \geq 0\right) \\ E_{out} &= \frac{1}{N_{out}^{test}} \sum_{x \in X_{out}^{test}} \mathrm{I}\left(d(x) < 0\right) \end{aligned} \tag{25}$$

We define E as the average of E_{in} and E_{out}.

$$E = \frac{E_{in} + E_{out}}{2} \tag{26}$$

We consider a set of parameters $\left(\sigma_{auto}, \rho_{auto}\right)$ as adequate if it leads to a compact set of SVDD contours enclosing most data points $x \in X_{in}^{test}$ while excluding most abnormal values $x \in X_{out}^{test}$, equivalent to producing contours that converge toward the theoretical distribution support of X'_{in}

. In that regard, we view as optimal the set of parameters minimizing simultaneously the average of E_{in} and E_{out}. The quality of every set of parameters $(\sigma_{auto}, \rho_{auto})$ generated by our selection method is measured by comparing its resulting E to the global minimum value of E at $(\sigma_{opt}, \rho_{opt})$ for all possible hyperparameters.

5.1.3 Experimental results

The probability distributions of the 17 synthetic datasets analyzed in this section are represented by the bitmap files illustrated in Figure 12. Each datasets were designed to reproduce diverse shape of different level of complexity, distinct groupings of points of varying numbers, relative sizes and proximity, and complex features such as concavity (hole) and occlusions.

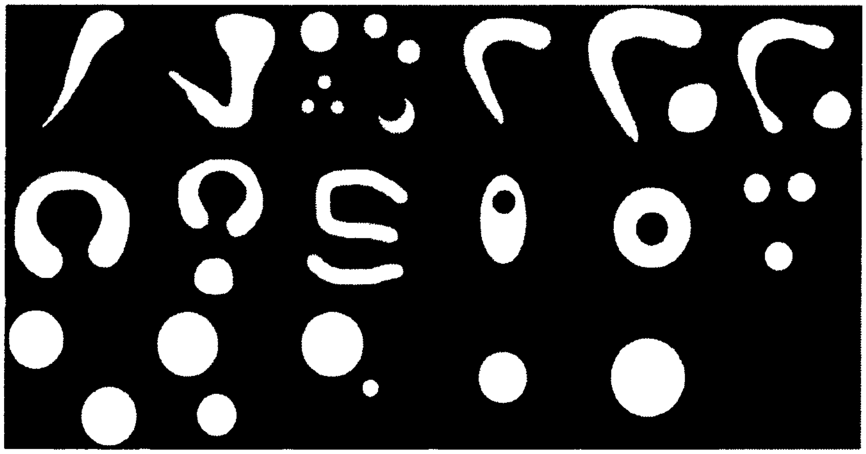


Figure 15 - Bitmap representations of the distributions of the synthetic datasets 1 to 17.

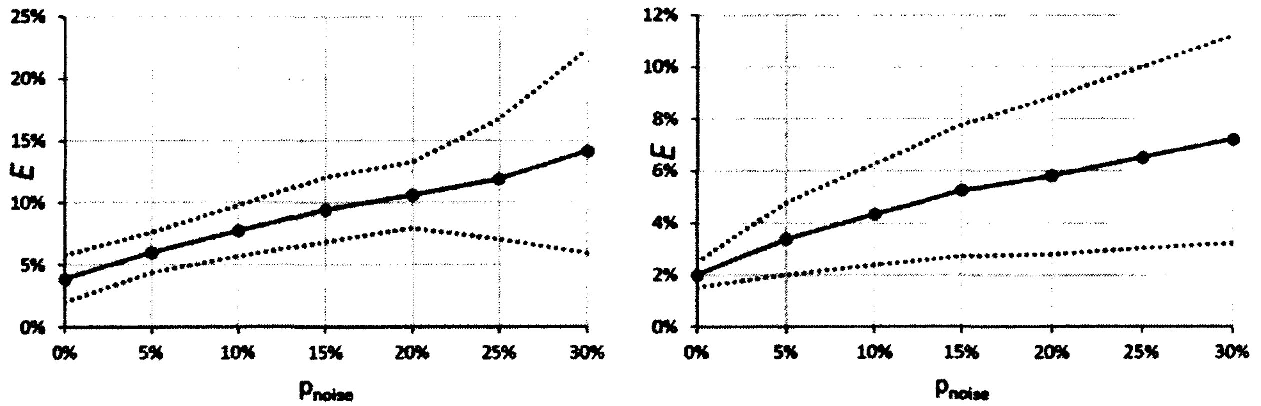
The results presented in this section compare values of E of SVDD models Θ_{auto} trained with $(\sigma_{auto}, \rho_{auto})$ selected according to our parameter selection method, to the minimal achievable values of E at $(\sigma_{opt}, \rho_{opt})$. Table 3 allows quantifying the sensibility of our method to noise, by selecting parameters on datasets X_{in}^{train} subjected to varying level of noise p_{noise} ranging from 0% to 30%, with fixed training set sizes $N_{in}^{train} = 1{,}000$, amplitude of noise $\sigma_{noise} = 0.4$ and $N_{in}^{test} = N_{out}^{test} = 10{,}000$. Note that every results presented in this section are calculated by performing a 40-fold cross-validation on experimental results.

Table 3 - Comparison of error rates E resulting from parameters $(\sigma_{auto},\rho_{auto})$ and $(\sigma_{opt},\rho_{opt})$ for varying proportions of noise p_{noise}.

[illegible]	[illegible]	1.8%	7.1%	9.2%	10.7%	10.9%	10.7%	8.6%
	[illegible]	1.8%	2.6%	3.3%	3.7%	3.8%	4.0%	4.6%
[illegible]	[illegible]	2.1%	3.9%	7.3%	8.7%	11.7%	11.5%	11.8%
	[illegible]	2.1%	4.0%	5.4%	6.4%	7.3%	7.9%	8.8%
[illegible]	[illegible]	5.4%	7.2%	11.0%	13.0%	13.0%	13.4%	20.3%
	[illegible]	3.2%	5.7%	6.8%	8.4%	9.2%	11.2%	12.4%
[illegible]	[illegible]	6.7%	6.9%	8.0%	9.5%	10.7%	9.1%	15.6%
	[illegible]	1.8%	3.1%	3.4%	4.3%	5.0%	5.2%	6.1%
[illegible]	[illegible]	5.9%	7.0%	8.6%	10.4%	12.2%	24.2%	21.9%
	[illegible]	2.3%	5.2%	7.0%	9.1%	10.1%	11.4%	13.2%
[illegible]	[illegible]	4.3%	6.7%	9.3%	10.3%	11.7%	11.5%	15.9%
	[illegible]	1.8%	3.6%	5.0%	6.2%	6.9%	8.0%	8.8%
[illegible]	[illegible]	3.2%	5.8%	8.7%	8.9%	14.4%	15.8%	16.8%
	[illegible]	3.2%	4.3%	6.0%	7.3%	8.9%	10.1%	11.5%
[illegible]	[illegible]	3.4%	7.7%	8.4%	14.6%	15.4%	15.7%	16.3%
	[illegible]	2.2%	4.5%	6.5%	7.9%	8.2%	10.3%	10.8%
[illegible]	[illegible]	4.9%	8.8%	11.3%	17.8%	17.5%	20.9%	21.6%
	[illegible]	2.7%	5.8%	7.8%	9.6%	11.2%	12.2%	14.0%

[illegible]	[illegible]	4.8%	4.3%	6.2%	7.2%	7.8%	8.1%	7.9%
	[illegible]	2.5%	2.3%	2.5%	3.0%	2.7%	2.7%	3.0%
[illegible]	[illegible]	8.3%	4.4%	4.9%	5.9%	6.8%	7.2%	6.8%
	[illegible]	1.6%	2.5%	2.9%	3.0%	2.9%	3.1%	3.3%
[illegible]	[illegible]	4.3%	6.5%	9.2%	11.5%	11.7%	11.4%	11.5%
	[illegible]	1.6%	2.8%	3.3%	4.3%	4.9%	5.3%	5.8%
[illegible]	[illegible]	3.2%	6.8%	7.7%	10.4%	11.1%	10.7%	10.9%
	[illegible]	1.6%	2.8%	3.3%	4.1%	4.0%	5.1%	6.0%
[illegible]	[illegible]	3.0%	4.4%	6.1%	7.8%	9.4%	9.8%	10.1%
	[illegible]	1.6%	3.1%	3.9%	4.8%	5.2%	5.7%	6.0%
[illegible]	[illegible]	2.4%	5.5%	6.7%	7.7%	8.3%	9.0%	10.1%
	[illegible]	2.2%	3.0%	3.5%	3.9%	4.0%	4.4%	4.7%
[illegible]	[illegible]	1.4%	3.3%	4.0%	6.1%	7.0%	6.2%	6.9%
	[illegible]	1.4%	1.6%	1.5%	1.3%	1.7%	1.8%	1.6%
[illegible]	[illegible]	1.6%	3.3%	5.2%	6.0%	6.8%	7.4%	7.0%
	[illegible]	1.3%	1.3%	1.6%	1.9%	1.7%	2.2%	2.1%
Average	[illegible]	3.8%	6.0%	7.8%	9.8%	10.8%	11.8%	14.1%
	[illegible]	2.0%	3.4%	4.3%	5.2%	5.8%	6.8%	7.2%

Figures 13 (a) and (b) compare the average error rates E over all 17 datasets (represented in Table 3) with parameters $(\sigma_{auto}, \rho_{auto})$ and $(\sigma_{opt}, \rho_{opt})$, for increasing proportions of noise p_{noise}.

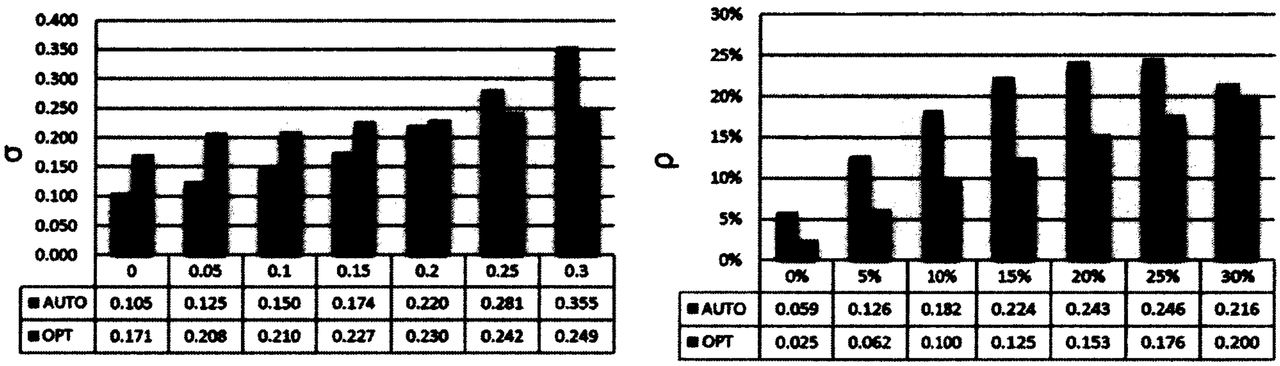


Figures 13a and 13b - Average error rates E for all 17 synthetic datasets for Θ_{auto} (left) and Θ_{opt} (right). Dotted lines represent standard deviations.

As illustrated in Figure 13 (b), the averaged minimal error achieved by Θ_{opt} grows linearly ($R^2 = 0.9798$) with the proportion of noise added to the training sets X_{in}^{train}. In Figure 13 (a), the SVDD models Θ_{auto} exhibits a moderate increase in errors increasing linearly ($R^2 = 0.9925$) with the proportion of noise. Standard deviations in Figure 13 (a) remain small and constant for $p_{noise} \leq 20\%$, and increase significantly past this point, suggesting that our method's robustness tends to deteriorate for $p_{noise} > 20\%$.

Figure 13 (b) also reveals that for a clean training set X_{in}^{train} where $p_{noise} = 0\%$, Θ_{opt} still exhibits an error rate of 2.0%, indicating that Θ_{opt} is unable to achieve an ideal classification rate for any combination of hyperparameters. This property is a consequence from the use of a single complexity parameter σ the whole dataset which acts as a trade-off over grouping of different complexity (as observed in [3]).

Figures 14 (a) and (b) summarize the average values of parameters $(\sigma_{auto}, \rho_{auto})$ selected according to our strategy and compared to $(\sigma_{opt}, \rho_{opt})$. As observed in Figure 14 (a), for $p_{noise} \leq 20\%$, our selection strategy produces slightly overly complex contours, resulting from underestimated values of σ and overestimated rejection rates ρ. The overestimated σ and underestimated rejection rates ρ for $p_{noise} > 20\%$ reflects the critical proportion of noise where our selection method become less efficient at estimating accurately the support of the noisy training set.



	0	0.05	0.1	0.15	0.2	0.25	0.3
■ AUTO	0.105	0.125	0.150	0.174	0.220	0.281	0.355
■ OPT	0.171	0.208	0.210	0.227	0.230	0.242	0.249

	0%	5%	10%	15%	20%	25%	30%
■ AUTO	0.059	0.126	0.182	0.224	0.243	0.246	0.216
■ OPT	0.025	0.062	0.100	0.125	0.153	0.176	0.200

Figures 14a and 14b - Average SVDD hyperparameters $\left(\sigma_{auto}, \rho_{auto}\right)$(left) and $\left(\sigma_{opt}, \rho_{opt}\right)$(right).

5.2 Real-world datasets

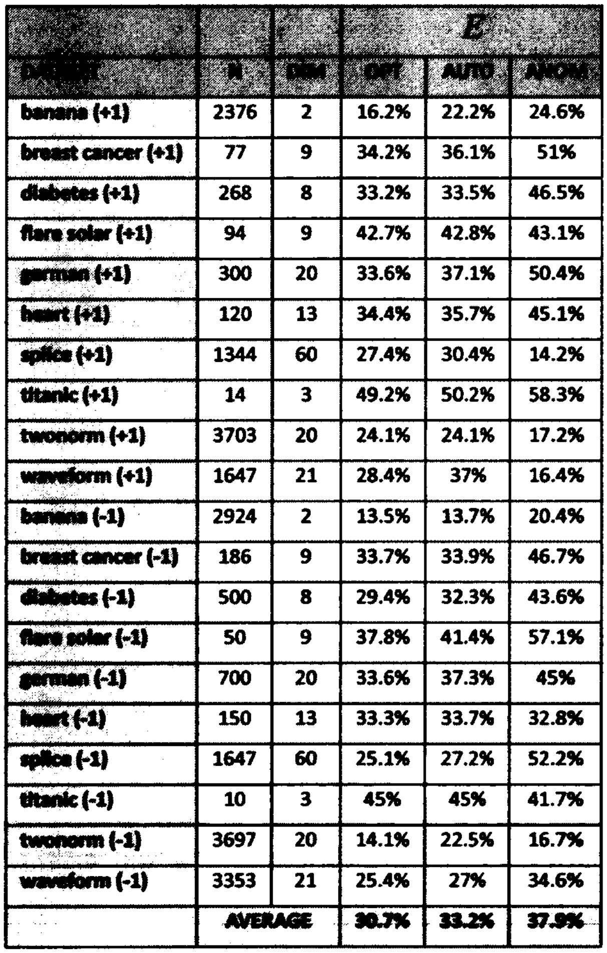
This section presents results on real-world benchmark datasets of varying sizes and dimensions. We compare the accuracy of SVDD models Θ_{auto} and Θ_{opt} trained with $(\sigma_{auto}, \rho_{auto})$ and $(\sigma_{opt}, \rho_{opt})$ to the SPSS Clementine 12.0[1] proprietary anomaly detection algorithm, referred to as ANOM. The SPSS anomaly detection algorithm is based on the SPSS proprietary TwoStep Cluster algorithm, which first performs a clustering process on the input dataset with the TwoStep method, then classifies all data points as normal or abnormal based on their respective cluster distance. Our choice of this algorithm for comparison results from the fact that, from our knowledge, it is the only other anomaly detection designed to perform anomaly detection without

[1] SPSS Clementine 12.0 is widely established software package, developed by SPSS Inc., which implements state-of-the-art algorithms.

relying on input parameters.

Experiments were performed on 10 classification datasets, each dataset being composed of two classes of patterns referred as +1 and -1. We used positive instances of each datasets as normal observations to train each algorithm, and evaluate the accuracy of each algorithm at identifying positive and negative instances of the class. We repeated this process on each class of each datasets, yielding to the evaluation of 20 datasets of dimensions varying from 2D to 60D, and sizes ranging from 10 to 3,703 observations.

Table 4 - Comparison of error rates E on 20 benchmark datasets.



			E		
DATASET	N	DIM	OPT	AUTO	ANOM
banana (+1)	2376	2	16.2%	22.2%	24.6%
breast cancer (+1)	77	9	34.2%	36.1%	51%
diabetes (+1)	268	8	33.2%	33.5%	46.5%
flare solar (+1)	94	9	42.7%	42.8%	43.1%
german (+1)	300	20	33.6%	37.1%	50.4%
heart (+1)	120	13	34.4%	35.7%	45.1%
splice (+1)	1344	60	27.4%	30.4%	14.2%
titanic (+1)	14	3	49.2%	50.2%	58.3%
twonorm (+1)	3703	20	24.1%	24.1%	17.2%
waveform (+1)	1647	21	28.4%	37%	16.4%
banana (-1)	2924	2	13.5%	13.7%	20.4%
breast cancer (-1)	186	9	33.7%	33.9%	46.7%
diabetes (-1)	500	8	29.4%	32.3%	43.6%
flare solar (-1)	50	9	37.8%	41.4%	57.1%
german (-1)	700	20	33.6%	37.3%	45%
heart (-1)	150	13	33.3%	33.7%	32.8%
splice (-1)	1647	60	25.1%	27.2%	52.2%
titanic (-1)	10	3	45%	45%	41.7%
twonorm (-1)	3697	20	14.1%	22.5%	16.7%
waveform (-1)	3353	21	25.4%	27%	34.6%
	AVERAGE		30.7%	33.2%	37.9%

5.2.1 Experimental results

For each experiment on the positive classes (+1) of patterns of a dataset, datasets X_{in} and X_{out} are respectively set to the classes (+1) and (-1) of pattern. Conversely, experiments on negative instances (-1) set X_{in} and X_{out} to negative (-1) and positive (+1) instances of the dataset. The sizes of X_{in}^{train} and X_{in}^{test} are both set to half the size of X_{in} , and X_{out}^{test} as the size of X_{out}. Experimental results are summarized in Table 4.

Average error rates E stated in Table 4 show that the SVDD models Θ_{auto} trained with parameters selected with our selection method exhibit an average mislabelling errors rate of 33.2% compared to the SVDD models Θ_{opt} exhibiting a classification accuracy of 30.7%, followed by the SPSS anomaly detection algorithm with an error rate of 37.9%.

From an anomaly detection perspective, SVDD models trained with the proposed parameter selection method significantly outperform the SPSS anomaly detection algorithm in terms of ability to distinguish accurately normal than abnormal patterns, on real-world datasets of varying sizes and dimensions.

6 Conclusion

In this paper, we have presented a robust and unsupervised hyperparameter selection method for SVDD which allows an accurate novelty detection on input datasets of arbitrary size, complexity and dimension. The proposed method yields a highly efficient generalization of the distribution support on even noisy datasets and does not require negative instances of the target class to select adequate hyperparameters. It relies on no input parameters, automatically adapts the SVDD complexity and rejection rate based solely on the input dataset characteristics, and yields high generalization performance on all synthetic and multidimensional real-world datasets evaluated. As revealed through our experimentation, the proposed method outperforms the SPSS anomaly detection algorithm on the majority of real-world datasets analyzed. The proposed strategy offers automatic anomaly detection with high accuracy without necessitating expert knowledge relative to a specific field. Furthermore, the proposed method implements an active overfitting prevention mechanism, minimizing the risk of overgeneralization commonly encountered in most machine learning algorithms trained on noisy, high-dimensional or small-sized datasets. Future research will focus on dynamic adjustment of the specificity for different regions of the domain of the dataset, and computational optimization of the hyperparameter selection process.

References

[1] Tax D M J, Duin R P W (1999) Support vector domain description. Pattern Recognition Letters, 20:1191-1199.

[2] Vapnik V. The nature of statistical learning theory. Springer, 1995.

[3] Zhuang L, Dai H (2006) Parameter Optimization of Kernel-based One-class Classifier on Imbalance Text Learning. Lecture Notes in Computers Sciences, 4099:434-443.

[4] Tax D M J, MüllerK (2004) A Constistency-Based Model Selection for One-class Classification. Proceedings on the 17th International Conference on Pattern Recognition, 3:363-366.

[5] Tax D M J , Duin R P W (2001) Uniform Object Generation for Optimizing One-class Classifiers. Journal of Machine Learning Research, 2:155-173.

[6] Karush W (1939) Minima of Functions of Several Variables with Inequalities as Side Constraints. M.Sc. Dissertation, Dept. of Mathematics, Univ. of Chicago.

[7] Kuhn H W, Tucker A W (1951) Nonlinear programming. Proceedings of 2nd Berkeley Symposium, University of California Press, pp 481-492.

[8] Keerthi S, Shevade S K, Bhattacharyya C, Murthy K R K (2001) Improvements to Platt's SMO Algorithm for SVM Classifier. Neural Computation, 13(3):637-649.

Conclusion

Contributions

Cette thèse doctorale présente une série de solutions algorithmiques et fonctionnelles visant à simplifier l'usage des méthodes « *Support Vector Data Description* » et « *Support Vector Clustering* » dans un contexte d'exploration non supervisée de données.

Cette recherche présente des solutions efficaces à trois limitations importantes inhérentes à ces deux méthodes, notamment 1) l'absence d'algorithmes d'optimisation efficaces et de stratégie d'apprentissage actif, permettant de résoudre la phase d'entrainement d'un SVC ou SVDD sur des données volumineuses dans un délai acceptable, 2) le manque de robustesse des méthodes existantes de partitionnement des données en sous-groupes distincts pour SVC, ainsi que 3) l'absence de stratégie guidant la sélection d'hyperparamètres contrôlant la complexité et la tolérance au bruit du modèle SVDD généré.

Un algorithme d'optimisation, F-SMO, a été mis au point afin de résoudre efficacement la phase d'entrainement d'un SVDD. F-SMO se distingue par sa capacité à compléter la phase d'entrainement au cours d'une lecture séquentielle des données, avec un temps d'entrainement réduit de 85% par rapport à l'algorithme usuel SMO. La stratégie d'apprentissage actif proposée, F-SMO-AL, constitue la première application d'apprentissage actif appliquée au SVDD. Cette stratégie intègre un mécanisme de sélection dynamique des candidats les plus informatifs lors du processus d'entrainement, et permet d'entrainer un modèle SVDD sur des données massives en un temps évoluant quasi linéairement en fonction du nombre de données. Le temps de calcul de F-SMO-AL offre une réduction du temps d'entrainement de 92.5% par rapport à SMO pour un nombre de supports vectoriels réduit de

75%. Notons que cette réduction significative du nombre de supports vectoriels permet une classification ultérieure largement plus rapide d'observations.

Le second objectif a mené au développement de L-CRITICAL, un algorithme robuste et efficace de segmentation des données en groupes homogènes pour SVC. Cet algorithme est basé sur un principe selon lequel l'ensemble de contours générés par un SVDD peut être divisé en contours distincts en analysant les interconnections entre chacun de leurs points critiques, permettant ensuite d'assigner chaque point à son contour disjoint le plus proche. Un algorithme efficace et précis de recherche de points critiques a été mise au point, basé sur l'algorithme Quasi-Newton jumelé à un processus de fusion des trajectoires de descente similaires. Les expérimentations effectuées sur des ensembles de données artificiels et réels complexes ont confirmé la robustesse et l'excellente vitesse d'exécution de L-CRITICAL, significativement plus précis et rapide que les algorithmes concurrents.

Le troisième objectif a été atteint par la création d'une stratégie de sélection des hyperparamètres pour SVDD. Un critère a été développé, permettant la détection et le rejet de paramètres induisant le phénomène de surgénéralisation (« *overfitting* »), ainsi qu'une stratégie permettant d'identifier les combinaisons de paramètres résultant en une représentation juste et compacte du domaine d'un ensemble de données en milieux bruités. La stratégie développée se distingue des méthodes concurrentes par sa capacité à optimiser les paramètres à partir uniquement d'instances positives de la classe, sans requérir à un ensemble d'instances négatives. Tel qu'illustré lors d'expérimentations sur des données artificielles et réelles, la stratégie proposée permet une excellente capacité de discrimination entre les éléments normaux et atypiques, comparables aux méthodes supervisées de sélection d'hyperparamètres sans dépendre d'un ensemble de validation composé d'instances négatives de la classe cible.

Critique du travail

L'algorithme F-SMO-AL intégrant le mécanisme d'apprentissage actif, bien que très efficace dans les expérimentations décrites dans l'article 1, est adapté aux ensembles de données de volume élevé, et peut faillir à sélectionner certains supports vectoriels essentiels sur de petits ensembles de données. Chaque étape de sélection consistant à choisir parmi une vingtaine d'observations le candidat le plus informatif, certains candidats cruciaux peuvent conséquemment être manqués lors d'une étape de sélection. Notons cependant que la stratégie d'apprentissage actif est réservée au traitement des ensembles de données volumineux, les ensembles de formats restreints pouvant être efficacement traités par l'algorithme F-SMO.

Par ailleurs, bien que largement plus efficace que les algorithmes concurrents, l'algorithme de partitionnement des données L-CRITICAL proposé pour SVC dans l'article 2, affiche un coût computationnel dépendant du nombre de supports vectoriels de la solution SVDD analysée. Par conséquent, la vitesse d'exécution de L-CRITICAL demeure proportionnelle au nombre de supports vectoriels de la solution et peut par conséquent s'alourdir en présence d'ensembles d'observations volumineux.

Finalement, tel que discuté dans l'article 3, la méthode d'optimisation des hyperparamètres pour SVDD produit des modèles offrant une excellente performance de généralisation en présence d'ensembles de données bruitées. Cependant, en présence de données hautement bruitées, la méthode peut faillir à généraliser convenablement la structure intrinsèque aux données. Par ailleurs, la méthode proposée permet de sélectioner le paramètre d'un noyau gaussien limité au traitement de données continues (réelles). L'extension de cette méthode à l'optimisation de paramètres de différents noyaux permettant le traitement de données de type mixte (continue, ordinales, nominales) demeure un sujet ouvert de recherche.

Travaux futurs de recherche

Les extensions potentielles à cette recherche sont multiples. En premier lieu, le processus d'optimisation des hyperparamètres présenté dans l'article 3 pourrait être appliqué récursivement sur chaque groupement homogène distinct, détecté par l'algorithme de partitionnement L-CRITICAL présenté dans l'article 2. Une telle stratégie permettrait de raffiner la représentation globale des données en présence de groupements de complexités variables, en attribuant à chaque groupe son propre ensemble optimal d'hyperparamètres.

La seconde extension concerne l'utilisation des méthodes proposées dans un contexte de classification automatique sur des données multi-classes. Un modèle SVDD serait optimisé séparément sur chaque classe individuelle, la classification d'une observation inconnue consistant à évaluer un point pour chaque modèle, résultant en une appartenance simultanée à plusieurs classes (analogue aux algorithmes de logique floue). Une telle approche offrirait deux avantages par rapport aux classificateurs SVM multi-classe actuels. Elle permettrait un traitement s'adaptant automatiquement aux classes d'observations sous représentées, chaque classe étant associée à un jeu spécifique d'hyperparamètres. De plus, le traitement indépendant des classes résulterait en un allègement considérable du temps de calcul global par rapport aux SVM multi-classes actuels sur un grand nombre de classes.

Perspective

L'exploration de données a été un domaine sujet à une croissance phénoménale au cours de la dernière décennie. Ce dernier a permis de résoudre des problèmes complexes tels que le dépistage de maladies génétiques, l'analyse de profils comportementaux chez les consommateurs, la détection de fraudes bancaires, et offrent un avantage stratégique considérable à toute entreprise possédant des bases de données, permettant d'optimiser l'efficacité de leurs opérations, de mieux cibler leur clientèle, et maximiser leur profit.

Les algorithmes et stratégies présentés dans cette étude offrent des solutions pratiques à plusieurs limitations fondamentales inhérentes aux SVM non supervisés, et permettent une utilisation simplifiée et plus efficace des algorithmes SVDD et SVC sur des ensembles de données réels. Nous souhaitons que les avancées présentées dans cette thèse puissent faciliter et favoriser l'utilisation des SVM non supervisés dans des contextes concrets d'analyse de données.

Bibliographie

[1] Ben-Hur, A., Horn, D., Siegelmann, H.T., Vapnik, V.: A Support Vector Clustering Method, *Proceedings 15th International Conference on Pattern Recognition*, 2, 2000, pp 724-727.

[2] Chicco, G., Ilie, I. S.: Support vector clustering of electrical load pattern data, *IEEE Transactions on Power Systems*, 24(3), 2009, pp 1619-1628.

[3] Dong, X., Zhaohui, W., Wanfeng, Z.: Support vector domain description for speaker recognition, *Proceeding of the 2001 IEEE signal processing society workshop*, 2001, pp 481-488.

[4] Hansen, M. S., Olafsdottir, H., Sjostrand, K., Erbou, S. G., Stegmann, M. B., Larsson, H. B. W., Larsen, R.: Ischemic segment detection using the support vector domain description, *Proceeding of SPIE*, 6512(1), 2007, pp 65120F-65120F-8.

[5] Hittiwachana, S., Ferreira, D. L. S., Lloyd, G. R., Fido, L. A., Thompson, D. R., Escott, R. E. A., Brereton, R. G.: One class classifiers for process monitoring illustrated by the application to online HPLC of a continuous process, *Journal of chemometrics*, 24, 2010, pp 96-110.

[6] Hu, Z. P., Tan, Y.: Novel region energy evolution image segmentation based on fuzzy object confidence description, *Acta Automatica Sinica,* 34(9), 2008, pp 1047-1052.

[7] Huang, J. J., Tzeng, G. H., Ong, C. S.: Marketing segmentation using support vector clustering, *Expert Systems with Applications*, 32, 2007, pp 313-317.

[8] Lee, J., Lee, D.: An Improved Cluster Labeling Method for Support Vector Clustering, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27, 2005, pp 461-464.

[9] Platt, J. C.: Fast training of support vector machines using sequential minimal optimization, *Advance in Kernel Methods*, 1999.

[10] Seo, J., Ko, H.: Face dectection using support vector domain description in color images, *IEEE International Conference on Acoustics, Speech and Signal Processing*, 5, 2004, pp 729-732.

[11] Siala, K., Chakchouk, M., Nesnes, O., Chaieb, F.: Moving shadow detection with support vector domain description in the color ratios space, *Proceedings of the 17th international conference on pattern recognition*, 1, 2004, pp 384-387.

[12] Tang, R., Han, J., Zhang, X.: Efficient iris segmentation method with support vector domain description, *Optica Applicata,* 39(2), 2009, pp 365-374.

[13] Tax, D. M. J., Müller, K.: A Constistency-Based Model Selection for One-class Classification, *Proceedings on the 17th International Conference on Pattern Recognition*, 3, 2004, pp 363-366.

[14] Tung, J. W., Hsu, C. T.: Lerning hidden semantic cues using support vector clustering, *IEEE International conference on image processing*, 2, 2005, pp 1189-1192.

[15] Vapnik, V.: *The Nature of Statistical Learning Theory* (ed. 2), New York: Springer-Verlag, 1995.

[16] Wang, C. H.: Robust segmentation for the service industry using kernel induced fuzzy clustering techniques, *IEEE international conference on industrial engineering and engineering management*, 2009, pp 2197-2201.

[17] Zhuang, L., Dai, H.: Parameter Optimization of Kernel-based One-class Classifier on Imbalance Text Learning, *Lecture Notes in Computers Sciences*, 4099, 2006, pp 434-443.