Improvement of Nearest-Neighbor Classifiers via Support Vector Machines

Marc Sebban and **Richard Nock**
TRIVIA-Department of Mathematics and Computer Science
Université des Antilles-Guyane 97159 Pointe-A-Pitre, France
msebban@univ-ag.fr, rnock@martinique.univ-ag.fr

Abstract

Theoretically well-founded, Support Vector Machines (SVM) are well-known to be suited for efficiently solving classification problems. Although improved generalization is the main goal of this new type of learning machine, recent works have tried to use them differently. For instance, *feature selection* has been recently viewed as an indirect consequence of the SVM approach. In this paper, we also exploit SVMs differently from what they are originally intended. We investigate them as a data reduction technique, useful for improving case-based learning algorithms, sensitive to noise and computationally expensive. Adopting the margin maximization principle for reducing the Structural Risk, our strategy allows not only to eliminate irrelevant instances but also to improve the performances of the standard k-Nearest-Neighbor classifier. A wide comparative study is presented on several benchmarks of UCI repository, showing the utility of our approach.

Introduction

Support Vector Machine (SVM) algorithms have been successfully applied in recent years to solve classification and regression problems. Their goal consists in constructing nonlinear decision functions by training classifiers to perform a linear separation on the original instances mapped by a function ϕ to a very high-dimensional feature space. To avoid computationally expensive calculations in this new space, one usually uses suitable *kernel* functions K. The choice of K determines if the SVM is a polynomial, a neural-network or a Radial Basis Function classifier.

The SVM strategy adopts the Structural Risk Minimization principle (Vapnik 1979). Theoretically, it amounts to minimizing the *VC (Vapnik–Chervonenkis)-dimension*, which is one of the two parameters (with the *empirical risk*) controllable for converging on the theoretical risk of the problem. The VC-dimension h for a set of functions $\{f(\alpha)\}$ is defined as the maximum number of points that can be *shattered* by $\{f(\alpha)\}$ (ex: $h = 3$ for the set of oriented lines in R^2). SVMs are in fact a practical implementation of this Structural Risk Minimization, and consists of a quadratic programming problem, and for which Burges and Crisp (2000) have presented conditions for the uniqueness of the solution. Geometrically, this approach consists in practice in maximizing margins between typical instances of classes, called *support vectors*.

Even if conceptually SVM algorithms are intended for solving classification problems, recent researches have tried to exploit the interesting theoretical properties of these machines for solving other learning or data analysis problems. For instance, Bradley and Mangasarian (1998) have shown that feature selection is an indirect consequence of the support vector machine approach. Moreover, Schölkopf et al. (1998) analyze SVM as an efficient way for extracting polynomial features for nonlinear Principal Component Analysis. In this paper, we also exploit SVMs differently from what they are originally intended. We show that *prototype selection* (PS) is a direct consequence of the SVM approach, and we investigate them as a data reduction technique in favor of case-based learning algorithms, such as the k-nearest neighbor classifiers (kNN) (Cover and Hart 1967). Actually, kNN classifiers are well known to be good candidates for solving machine learning problems. Nevertheless, this effectiveness is counterbalanced by large computational and storage requirements. In such a context, many data reduction methods has been proposed last decades for reducing storage requirements, while not compromising the generalization accuracy of the model (Hart 1968), (Gates 1972), (Aha, Kibler et Albert 1991), (Aha 1992), (Brodley and Friedl 1996), (Brighton and Mellish 1999), (Wilson and Martinez 2000), (Sebban and Nock 2000). Wilson and Martinez (2000) have presented a framework for clustering PS algorithms according to different criteria: *instance representation, type of selected instances (center or border points)*, and *direction of search (incremental, decremental);* we can also add to this list the *type of optimized criterion (empirical risk, information measure, statistical criterion, etc.)*. In this paper, we propose a new manner to select prototypes which uses the SVM principle for improving the performances of a kNN classifier, not only in terms of complexity but also of generalization. This way to proceed can be justified by the

following arguments:

- While most of the prototype selection procedures are based on heuristics minimizing suited criteria (*empirical risk, information measures*, etc.), several results have already shown that SVMs are theoretically well-founded (reduction of the Structural Risk (Vapnik 1979), uniqueness of the solution (Burges and Crisp 2000), etc.). This rigorous framework, added to the fact that the Structural Risk has not been yet exploited in the field of prototype selection, is a convincing argument for a wider use of SVMs.
- SVMs adopt the margin maximization principle which consists in pushing apart two parallel planes (for a 2 class problem), while keeping decision boundaries. This planes are *supported* by *support vectors*, which can be viewed as typical instances. SVMs use these *support vectors* in their decision function for labelling a new unknown example. We think that this way to proceed can be extended to the prototype selection field which also searches for relevant prototypes while not compromising the generalization accuracy.
- Schölkopf et al. (1995) have shown that most deduced *support vectors* are shared by the different *kernels* K, *i.e.* most of the centers of an SVM with Gaussian kernels coincide with the weights of the polynomial and neural network SVM classifiers. This interesting property seems to show that SVMs construct efficient and stable decision boundaries whatever the kernel functions used. Then, we think that we can exploit these support vectors for extracting prototypes also useful for a kNN classifier. That is the aim of this paper.

After a brief review of the SVM principles in Section 2, we present the framework of our adaptation of SVMs to prototype selection in Section 3, as well as our algorithm, called PS^{2}VM. A large comparative study with the state-of-the-art prototype selection algorithms is provided in Section 4, according to three performance measures: the *storage reduction*, the *generalization accuracy* and the *sensitivity to noise*.

The Support Vector Machine

Many theoretical results have shown that the minimization of the empirical risk R_{emp} is not sufficient to guaranty the minimization of the actual risk R, when R_{emp} is assessed from a limited number l of learning examples. The Support Vector Machines are based on the following inequality, with probability $1-\eta$:

$$R \leq R_{emp} + \Phi(\frac{h}{l}, \frac{\log(\eta)}{l})$$
$$\text{where } \Phi(\frac{h}{l}, \frac{\log(\eta)}{l}) = \sqrt{\frac{h(\log\frac{2l}{h}+1)-\log(\frac{\eta}{4})}{l}}$$

The main parameter of the second term is h, which corresponds to VC-dimension of a set of functions. This Vapnik Chervonenkis-dimension describes the capacity of a set of functions implementable by the learning machine. Then, one can control R by controlling two quantities: R_{emp} which depends on the function chosen by the learning machine, and h the VC dimension. The Support Vector Machines propose an efficient practical implementation of this Structural Risk Minimization, for which we propose to present below a brief review.

Consider a problem with two classes and l learning instances (x_i, y_i), where x_i is an n-dimensional real vector and y_i is the class (-1 or +1). If the points are *linearly separable*, there exists a decision rule $f(x_i) = w.x_i + b$. which attributes the label +1 to a given x_i instance, if $w.\ x_i + b \geq 1$, and the label -1 if $w.\ x_i + b \leq -1$. The problem consists in determining the n-vector w and the scalar b. In fact, in many real world problems, a model without classification errors does not exist. That is the reason why Cortes and Vapnik (1995) introduce slack variables $\xi_i \geq 0, i = 1, ..., l$, to get $y_i(w.\ x_i + b) \geq 1 - \xi_i$.

Minimizing the guaranteed risk R with a SVM algorithm amounts in fact to minimizing the following quantity $\Phi(w, \xi) = w.w + \gamma \sum \xi_i$, where γ is a positive constant. Minimizing the first term amounts to minimizing the VC dimension h, and minimizing the second (which corresponds to an upper bound of the number of misclassified instances) amounts to minimizing the empirical risk R_{emp}.

Using Lagrange multipliers α_i and the Kuhn-Tucker theorem, the solution of this optimization problem is

$$w = \sum_{i=1}^{l} y_i \alpha_i x_i$$

Then, the decision rule becomes,

$$f(x) = sgn(\sum_{i=1}^{l} y_i \alpha_i (x.x_i) + b)$$

Instance x_i having a corresponding $\alpha_i \neq 0$ are called *support vectors* of the problem. Geometrically, minimizing $w.w$ corresponds in fact to maximize margins between two separating planes, one supporting the class -1 and the other supporting the class +1 (see Figure 1 for a 2-D example with its corresponding support vectors).

So far, we have briefly described linearly separable problems. In order to allow decision surfaces which are *not linear*, the SVM algorithm projects, by a function ϕ, input vectors in a high-dimensional feature space allowing then a linear separation. In order to avoid computationally expensive calculations in this high-dimensional space (*i.e.* $\phi(x).\phi(x_i)$), one usually use suitable functions K such as,

$$\phi(x).\phi(x_i) = K(x, x_i)$$

Then, the decision rule becomes,

$$f(x) = sgn(\sum_{i=1}^{l} y_i \alpha_i K(x.x_i) + b)$$
where K is called the *kernel*.

Different kernel functions as been already used, which determine whether the resulting SVM is a polynomial

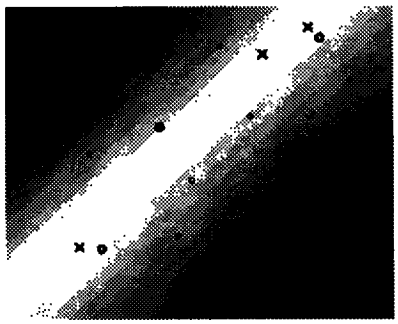

Figure 1: A 2-class simulated problem with its support vectors (indicated by an extra circle) and its errors (indicated by a cross).

classifier, a neural network, a radial basis function machine, or some other learning machine. For example, we can use the following kernels:

- $K(x, x_i) = (x.x_i + 1)^d$ *i.e.* $\phi(x)$ is a polynomial function,
- $K(x, x_i) = \tanh(ax.x_i)$ for building a neural network,
- $K(x, x_i) = e^{-a\|x.x_i\|^2}$ is a Gaussian kernel.

Surprisingly, Schölkopf et al. (1995) have shown that most of the deduced support vectors are shared by the different kernels. Starting from this interesting property, our aim consists in exploiting the SVM decision rule for eliminating irrelevant learning instances and improving the performances of the kNN classifiers.

SVM and Prototype Selection

Prototype selection can be viewed as an implicit consequence of the Support Vector Machines. Actually, even if the decision rule $f(x) = sgn(\sum_{i=1}^{l} y_i \alpha_i K(x.x_i) + b)$ uses theoretically all the l learning instances, only those which have a coefficient $\alpha_i \neq 0$ are in fact taken into account. All the remaining examples x_i of the training set which have a coefficient $\alpha_i = 0$ can be viewed as being irrelevant. Then, not only SVMs provide a rigorous theoretical framework by minimizing the VC-dimension, but also it provides a favorable framework for reducing storage requirements, in accordance with the prototype selection constraints. Actually, PS algorithms search for small and efficient subsets of instances which keep as well as possible decision boundaries, after removing different categories of irrelevant instances:

- the first belong to regions where local densities are evenly distributed,
- the second concerns mislabeled instances,
- the last concerns useless instances at the center of clusters.

By analyzing the SVM strategy, we can note that all these objectives can be satisfied. In the linear case

PS^{2}VM(LS)
  Determine Support Vectors
  Remove from LS instances
    misclassified by SV
  Remove useless instances such as
    $\forall x_j \in LS$: **if**

$$|\sum_{i=1}^{n_{sv}} \alpha_i K(x_i.x_j))| > 1$$

  then remove x_j **from LS**
  Return LS

Figure 2: Pseudocode for PS^{2}VM.

(which can be easily extended to nonlinear decision surfaces), three categories of instances can be drawn from the constraint $y_i(w.\ x_i + b) \geq 1 - \xi_i$.

- the support vectors for which the equality $y_i(w.\ x_i + b) - 1 = 0$ (with $\xi_i = 0$) holds. They maximize margins and do not affect decision boundaries. An efficient PS algorithm based on SVM must keep these relevant instances.
- the irrelevant instances which satisfy automatically the constraint $y_i(w.\ x_i + b) > 1 - \xi_i$ with $\xi_i = 0$. These instances are irrelevant because they are automatically correctly classified by support vectors. They do not bring useful information for the model and so they have to be remove by our PS algorithm.
- the last concerns misclassified and mislabeled instances. For an error to occur, ξ_i must exceed unity (that is the reason why $\sum \xi_i$ is an upper bound on the number of test errors). Then, we have to remove instances for which $\xi_i > 1$.

From these different properties of SVMs, we can easily derive our new PS^{2}VM (for Prototype Selection via Support Vector Machine) algorithm, for which the pseudocode is presented in Figure 2..

Experimental Results

Presentation

The goal of this section consists in assessing PS^{2}VM according to three performance criteria, usually used in prototype selection: the *storage reduction*, the *generalization accuracy* and the *sensitivity* of the algorithm *in the presence of noise*. Moreover, our aim is to compare PS^{2}VM with the state-of-the-art PS algorithms. Even if several PS methods have been proposed last decades, we decided to compare PS^{2}VM with three recent approaches which cover a large spectrum of PS strategies:

- The Consensus Filter CF (Brodley and Friedl 1996) was originally proposed for eliminating mislabeled instances, by a consensus vote of different base-level detectors, in agreement to remove an example. The CF has already shown very interesting results not only for improving kNN performances (Sebban and Nock

Datasets	kNN	CF		RT3		PSRCG		SVM (linear)		SVM (polyn.)		SVM (RBF)	
	Acc.	Acc.	prot.	Acc.	prot.	Acc.	prot.	Acc.	prot.	Acc.	prot.	Acc.	prot.
Audiology	73.7	74.4	85.7	71.1	11.3	73.9	68.0	70.6	40.2	72.1	52.1	75.6	66.4
Australian	78.7	77.5	84.4	75.3	11.8	77.5	58.4	79.1	85.1	79.4	50.4	77.7	48.8
Big Pole	59.9	60.8	69.8	57.6	14.1	59.2	86.9	64.4	49.4	61.1	50.8	62.1	63.4
Breast	96.2	96.6	97.3	63.6	1.1	96.9	9.7	95.3	5.1	96.9	35.8	96.0	44.2
Echocardio.	55.7	68.4	67.0	55.9	6.2	62.0	68.1	67.0	42.6	66.9	58.9	64.9	53.0
German	71.5	72.6	77.4	69.1	9.4	72.2	72.2	71.5	35.4	72.7	48.1	72.4	39.3
Glass	73.2	71.0	79.2	55.5	9.6	74.0	66.0	74.6	72.6	75.2	78.6	74.6	71.9
Hard	47.5	47.4	65.5	44.7	14.7	45.4	88.1	50.8	56.3	41.2	75.9	48.0	57.9
Heart	76.7	79.2	80.8	72.1	4.2	77.4	45.6	80.7	35.2	80.3	61.8	80.3	38.7
Hepatitis	82.4	81.0	87.7	76.3	9.0	80.1	45.7	82.2	30.9	78.0	51.3	76.3	40.9
Horse Col.	72.9	72.5	80.0	71.1	9.2	72.7	74.7	62.1	28.8	62.9	60.9	74.8	79.1
Ionosph.	81.0	80.4	87.3	75.6	9.4	80.4	62.1	75.6	30.5	75.0	70.9	78.3	56.8
LED+17	76.1	76.6	81.5	72.8	21.8	76.1	83.8	63.7	31.3	76.1	98.7	76.1	64.7
LED2	85.1	89.0	91.0	62.5	2.3	84.3	45.6	79.4	44.1	89.2	57.7	88.2	57.0
Pima	67.6	67.4	75.9	61.9	6.3	67.3	65.6	71.3	46.4	70.1	46.8	69.1	43.6
Vehicle	71.8	71.7	77.5	60.8	8.9	70.3	65.1	72.2	71.2	71.5	56.2	73.5	58.0
White House	91.9	91.2	94.5	84.5	4.2	91.7	29.7	65.7	10.8	91.5	46.2	91.4	28.6
Xd6	78.8	79.3	85.5	69.0	14.9	78.8	71.8	78.3	41.7	81.3	51.9	78.0	48.4
Average	74.5	75.4	81.5	66.6	9.4	74.5	61.5	72.5	42.1	74.5	58.5	75.5	53.4

Table 1: Generalization accuracy (Acc.) and storage requirements (prot in % of the original size) for 3 PS algorithms and 3 kernel functions used in PS^2VM on 18 datasets. The standard kNN is added for comparisons.

2000) but also for reducing decision tree complexity (Brodley and Friedl 1996). Here, we used three kNN classifiers as base-level detectors, k=1,..,3.

- In (Sebban and Nock 2000) authors proposed with PSRCG a new way to proceed by tackling the PS problem as an information preserving problem, and the minimization of local and global quadratic entropies in a neighborhood graph.
- Finally, in their algorithm RT3 (Wilson and Martinez 2000), an instance ω is removed if its removal does not hurt the classification of the instances remaining in the sample set, notably points that have ω in their neighborhood (called *associates*).

We compared these PS algorithms on 18 datasets, mainly coming from the UCI database repository[1]. For each dataset we achieved a 10 fold-cross-validation procedure, applying the prototype selection algorithms on 9 folders and testing the model on the remaining one. The same procedure is repeated for the 10 permutations. In order to estimate the sensitivity of PS^{2}VM to the choice of the *kernel function*, we applied with the same experimental setup three standard functions K: *linear, polynomial and Radial Basis Function.*

Storage Reduction and Generalization Accuracy

In this section, we intend to evaluate the storage reduction and the generalization ability of the classifiers obtained after elimination of irrelevant instances *via* our PS^{2}VM algorithm. The performance on the tolerance to the presence of noise in data is discussed in the next subsection.

From results of Table 1, we note that it is very difficult both to reduce the storage requirements and control the generalization accuracy. The Consensus Filters provides actually the best generalization accuracy (75.4%, *i.e.* +0.9 in comparison with the standard kNN), but also the worst storage reduction (only 18.5%). On the other hand, RT3 achieves the highest storage reduction (80.6%), while not controlling the accuracy of the model (-7.9). According to the two first performance criteria, PSRCG achieves the highest performances among the state-of-the-art PS algorithms (this note was already done in (Sebban and Nock 2000)), reducing the storage requirements of about 40%, while achieving the same accuracy.

Table 1 also presents the accuracy generalization and the storage requirement of the models built after removing irrelevant instances by PS^{2}VM. Except RT3 (which controls only one performance criterion), PS^{2}VM removes more instances than the standard PS algorithms, while controlling accuracies. The best one is indisputably PS^{2}VM with the RBF kernel. Not only this procedure allows to reduce more efficiently the storage requirements than PSRCG (53.4%, resulting in a large computation reduction during the classification phase of new unkown instances) but also it improves the generalization accuracy of the standard kNN on the whole learning set (75.5% vs 74.5%). Using a Student paired t-test, we can note that this difference is highly significant with a critical risk near 5%. Then, PS^{2}VM with the RBF kernel provides the best balance between the storage reduction and the generalization accuracy.

[1] http://www.ics.uci.edu/~mlearn/MLRepository.html

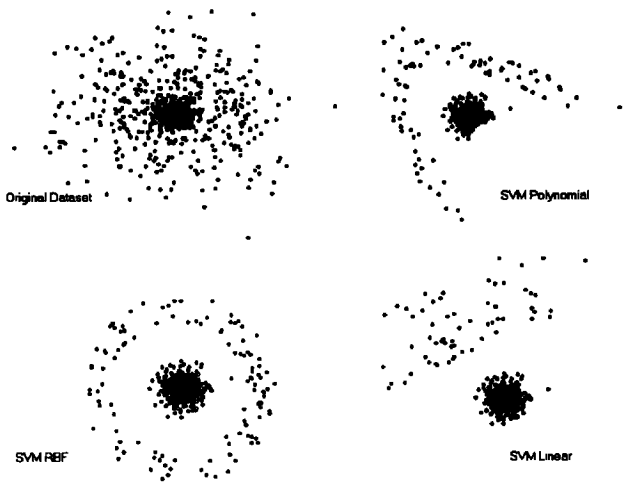

Figure 3: 2D-example and prototypes selected by PS^{2}VM according to three kernel functions.

Figure 3 shows empirically on a 2-class problem that PS^{2}VM with the RBF kernel can actually efficiently solve problems for which the other kernel functions can encounter difficulties.

Sensitivity to Noise

After analyzing the storage reduction and the generalization accuracy, we assess here the tolerance of PS^{2}VM in the presence of noise. A standard approach consists in adding artificially some noise in the data, by randomly changing the output class of 5% of the learning instances to an incorrect value. Table 2 presents results on these noisy datasets, which confirm those presented on noise-free databases, and the order established between the different PS algorithms.

PS Algorithm	Noise Free		Noisy	
	Acc.	Prot.	Acc.	Prot.
kNN	74.5±6.1	100	72.4±7.1	100
CF	75.4±6.7	81.5	73.4±6.8	79.4
RT3	66.6±9.9	9.4	64.4±9.7	8.9
PSRCG	74.5±6.3	61.5	72.3±7.2	64.3
SVM Linear	72.5±7.1	42.1	70.5±8.2	42.9
SVM Polyn.	74.5±6.6	58.5	72.0±7.9	52.2
SVM RBF	75.5±6.4	53.4	73.4±6.9	54.6

Table 2: Generalization accuracy and storage requirements when noise is inserted

Conclusion

We have presented in this paper an adaptation of Support Vector Machines to the prototype selection problem for kNN classifications tasks. Results on several data bases confirm the utility of our approach, and show that such a strategy seems to be more efficient than the standard prototype selection algorithms, with higher accuracy and storage reduction. So far, we decided to remove an instance if this one is misclassified (*i.e.* $\xi_i > 1$). Nevertheless, we think that it might be interesting in future works to statistically study the deletion of an instance according to a minimal risk of error, that would allow higher storage reduction.

References

D.W. AHA. Tolerating noisy, irrelevant and novel attributes in instance-based learning algorithms. *International Journal of Man-Machine Studies*, 36(1):267–287, 1992.

D.W. AHA, D. KIBLER, and M.K. ALBERT. Instance-based learning algorithms. *Machine Learning*, 6:37–66, 1991.

P.S. BRADLEY and O.L. MANGASARIAN. Feature selection via concave minimization and support vector machines. In *International Conference on Machine Learning*, 1998.

H. BRIGHTON and C. MELLISH. On the consistency of information filters for lazy learning algorithms. In *Third European Conference on Principles and Practices of Knowledge Discovery in Databases, PKDD'99*, pages 283–288, 1999.

C.E. BRODLEY and M.A. FRIEDL. Identifying and eliminating mislabeled training instances. In *Thirteen National Conference on Artificial Intelligence*, 1996.

C. BURGES and D.J. CRISP. Uniqueness of the svm solution. In *To appear in NIPS 12.*

C. CORTES and V. VAPNIK. Support vector networks. *Machine Learning*, 20:273–297, 1995.

T.M. COVER and P.E. HART. Nearest neighbor pattern classification. *IEEE. Trans. Info. Theory*, IT13:21–27, 1967.

G.W. GATES. The reduced nearest neighbor rule. *IEEE Trans. Inform. Theory*, pages 431–433, 1972.

P.E. HART. The condensed nearest neighbor rule. *IEEE Trans. Inform. Theory*, pages 515–516, 1968.

B. SCHOLKOPF, C. BURGES, and V. VAPNIK. Extracting support data for a given task. In *First International Conference on Knowledge Discovery and Data Mining*, 1995.

B. SCHOLKOPF, A. SMOLA, K.R. MULLER, C. BURGES, and V. VAPNIK. Support vector methods in learning and feature extraction. *Australian Journal of Intelligent Information Processing Systems*, 1:3–9, 1998.

M. SEBBAN and R. NOCK. Instance pruning as an information preserving problem. In *International Conference on Machine Learning*, 2000.

V. VAPNIK. *Estimation of Dependences Based on Empirical Data.* Springer Verlag, 1979.

D.R. WILSON and T.R. MARTINEZ. Reduction techniques for exemplar-based learning algorithms. *Machine Learning*, 38(3):257–286, 2000.