

Solving discrete support vector machines with tabu search

Stefan Lessmann¹, Stefan Voß²

Abstract— We consider the case of classification with discrete support vector machines. While standard support vector machines use a continuous approximation to measure classification errors discrete support vector machines incorporate a step-function minimizing errors directly. We argue that this modification facilitates more accurate class predictions. In particular, applications that involve cost sensitive learning with asymmetric costs per error type should benefit from a discrete error measure. However, while support vector machines are trained by minimizing a convex quadratic program discrete support vector machines require solving a more complex mixed integer program. Therefore, we develop a tabu search heuristic to train the respective classifier. Considering the structure of the underlying optimization problem its optimal solution has to be contained in the set of extreme points of a relaxed problem that can be solved by fast linear programming methods. Our tabu search exploits this observation utilizing the respective extreme points as the neighbourhood between candidate solutions. Using this extreme point tabu search we compare discrete support vector machines with standard support vector machines on well known benchmark data sets. Experiments include standard as well as cost sensitive classification settings. The discrete support vector machine is found to deliver superior results when cost-distributions are uneven while it performs competitive in standard settings.

I. INTRODUCTION

The support vector machine (SVM) is a supervised learning algorithm capable of solving linear and non-linear binary classification problems. A classifier D is a machine that performs a mapping from a multi-dimensional input space $X \subseteq \mathbb{R}^n$ to a discrete output space Y . Elements from Y are interpreted as class labels and we use the term classification to refer to the process of assigning a label $y_i \in Y$ to $\mathbf{x}_i \in X$. The vector \mathbf{x}_i is called an example or observation and consists of n individual attributes. Given a data set S of m pre-classified patterns $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$ a classifier is constructed by induction such that the risk of misclassifying novel examples with yet unknown class membership is minimized [1, 2]. This minimization is also referred to as training the classifier. Subsequently, we consider the special case of binary classification where $y_i \in \{-1, +1\}$.

Applications of classification can be found in several domains like managerial decision support (e.g. classifying loan applicants in good or bad credit risks [3]), medical diagnosis (e.g. classifying cancerous versus healthy cells [4]), text mining as well as speech or image recognition.

SVM training consists of solving a convex, quadratic program (QP) using a continuous approximation of classification error. Recently, Orsenigo and Vercellis [5] proposed a discrete SVM (DSVM) formulation that avoids this approximation and minimizes classification errors directly. Therewith, DSVM also facilitates explicit cost-minimization making it a promising candidate for scenarios where the costs of misclassification are uneven. However, constructing a DSVM with binary error variables involves solving a non-convex combinatorial program. Since standard SVM is based on a quadratic convex problem classifier training is substantially more expensive for DSVM.

We propose an extreme point tabu search heuristic for DSVM training exploiting the fact that the optimal solution of the resulting mixed integer program (MIP) is contained in the set of extreme points of the continuous relaxation [6]. Consequently, standard simplex pivot operations provide an effective way to define the neighborhood for any local search based algorithm. To evaluate our training procedure we conduct an empirical study comparing SVM and DSVM on well known benchmark data sets. These experiments include symmetric as well as asymmetric cost settings to investigate not only the accuracy of DSVM but also its effectiveness for cost-sensitive learning.

The remainder of the paper is organized as follows. A brief introduction to SVMs is given in the following section before discussing the DSVM formulation and its extensions to non-linear classification. Subsequently, Section III describes the tabu search heuristic to solve the DSVM training problem. Results of our empirical comparisons with the standard SVM can be found in Section IV. Conclusions are given in Section V.

II. SUPPORT VECTOR MACHINES

A. Standard support vector machines

The standard SVM is a linear classifier so that class predictions $y(\mathbf{x})$ are based on evaluating a linear equation:

$$y(\mathbf{x}) = \text{sign}(\mathbf{w}^* \cdot \mathbf{x} + b^*). \quad (1)$$

The parameters \mathbf{w}^* and b^* in (1) are obtained by solving a convex QP with linear constraints; see (3) below. This program is derived from the idea to separate the training data set S with a maximal margin hyperplane [7]. That is, the

¹ Stefan Lessmann, Institute of Information System, University of Hamburg, Von-Melle-Park 5, D-20146 Hamburg, Germany (phone: 0049-40-42838-5500; fax: 0049-40-42838-5535; e-mail: lessmann@econ.uni-hamburg.de).

² Stefan Voß, Institute of Information System, University of Hamburg, Von-Melle-Park 5, D-20146 Hamburg, Germany (phone: 0049-40-42838-3064; fax: 0049-40-42838-5535; e-mail: stefan.voss@uni-hamburg.de).

algorithm strives to maximize the distance between examples of contrary classes which are closest to the separating plane; see Fig. 1. It has been shown that maximizing the margin of separation improves the generalization ability of the resulting classifier, i.e., it minimizes the risk of misclassifying novel examples [8]. To construct such a maximal margin classifier one has to minimize the norm of the weight vector \mathbf{w} under the constraint that the training patterns of each class reside on opposite sides of the separating surface; Fig. 1. Since $y_i \in \{-1, +1\}$ we can formulate this constraint as:

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

Examples which satisfy (2) with equality are called support vectors since they define the orientation of the resulting hyperplane.

To account for misclassifications, e.g. examples where constraint (2) is violated, the so called soft margin formulation introduces continuous slack variables $\xi_i \in \mathfrak{R}$ [7]. To construct a SVM classifier one has to solve a convex QP:

$$\begin{aligned} \min_{\mathbf{w}, b, \xi} \quad & \frac{\beta}{2} \|\mathbf{w}\| + (1 - \beta) \left(C^+ \sum_{\{i|y_i=+1\}} \xi_i + C^- \sum_{\{i|y_i=-1\}} \xi_i \right) \\ \text{s.t.} \quad & y_i((\mathbf{w} \cdot \mathbf{x}_i) + b) \geq 1 - \xi_i, \quad i = 1, \dots, m \\ & \xi_i \geq 0, \quad i = 1, \dots, m. \end{aligned} \quad (3)$$

In (3), β is a tuning parameter which allows the user to control the trade off between maximizing the margin and minimizing misclassifications within the training set. Note that (3) is an slightly extended SVM providing a mechanism to account for asymmetric misclassification costs by weighting errors with class dependant weights C^+ and C^- [9, 10].

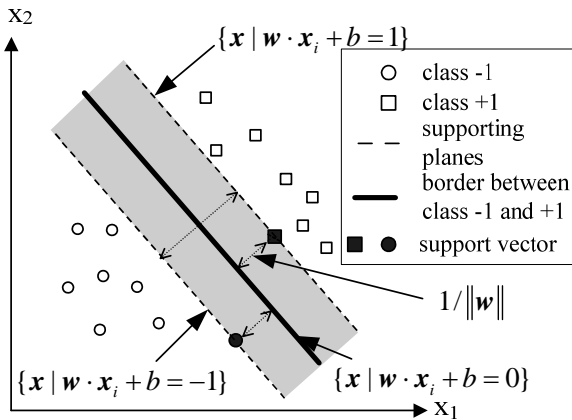


Fig. 1: Linear separation of two classes +1 and -1 in two-dimensional space with SVM classifier [8].

To construct non-linear decision surfaces SVMs implement the idea to map the input vectors into a high-dimensional feature space via an a priori chosen non-linear mapping function Φ . Constructing a separating hyperplane in this feature space leads to a non-linear decision boundary in the input space. Expensive calculations in high-dimensional spaces can be avoided by introducing a kernel

function. A kernel K calculates the dot product of two vectors in a transformed space directly in the input space:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j). \quad (4)$$

Instead of solving (3) directly, standard methods for SVM training [11-13] consider the dual problem. Since the input data appears only in form of dot products in the dual integration of kernel functions this straightforward and does not affect the overall training algorithm [7]. Prominent candidate kernel functions are linear, radial or polynomial kernels [8].

B. Discrete support vector machines

The original SVM utilizes the distance of a misclassified point to the separating hyperplane as error measure. That is, the discrete classification error is replaced by the continuous proxy ξ_i for computational convenience; see (3). DSVM [5] reverses this simplification and replaces ξ_i by $\theta_i \in [0, 1]$ to obtain a QP with integer constraints:

$$\begin{aligned} \min_{\mathbf{w}, b, \xi} \quad & \frac{\beta}{2} \|\mathbf{w}\| + (1 - \beta) \left(C^+ \sum_{\{i|y_i=+1\}} \theta_i + C^- \sum_{\{i|y_i=-1\}} \theta_i \right) \\ \text{s.t.} \quad & y_i((\mathbf{w} \cdot \mathbf{x}_i) + b) \geq 1 - \theta_i, \quad i = 1, \dots, m \\ & \theta_i \in \{0, 1\}, \quad i = 1, \dots, m. \end{aligned} \quad (5)$$

This program is non-convex and non-differentiable due to the integer constraint so that standard procedures for SVM training are no longer applicable. One idea to solve (5) is to replace the integer constraint with a linear relaxation, e.g.:

$$\begin{aligned} \min_{\mathbf{w}, b, \xi} \quad & \frac{\beta}{2} \|\mathbf{w}\| + (1 - \beta) \left(C^+ \sum_{\{i|y_i=+1\}} \theta_i + C^- \sum_{\{i|y_i=-1\}} \theta_i \right) \\ \text{s.t.} \quad & y_i((\mathbf{w} \cdot \mathbf{x}_i) + b) \geq 1 - \theta_i, \quad i = 1, \dots, m \\ & 0 \leq \theta_i \leq 1, \quad i = 1, \dots, m. \end{aligned} \quad (6)$$

Since (6) can be solved by standard QP algorithms one could follow a branch and bound strategy [14] to find an optimal solution. However, considering the performance of standard QP solvers this approach would be feasible only for small data sets [15]. Therefore, we consider a modified DSVM formulation: We substitute the L2-norm in SVMs' objective by the L1-norm to obtain a linear program (LP). We justify this simplification with two observations. First, the modification does not affect the core principle of SVM to balance the two competitive goals of constructing a classifier with large margin and low training error. Therefore, the modified SVM is still compatible with the principle of structural risk minimization [8]. Second, it has been shown that the L1-norm forces more elements of the weight vector to zero and therewith increases the interpretability of the classifier [16, 17]. Interpretability is beneficial in any application giving further support for using the L1-norm.

The resulting formulation is given in (7) where u_j denotes a primal decision variable that controls the value of the weight vector \mathbf{w} and Q is a sufficiently large number.

$$\begin{aligned}
\min_{w,b,\theta,u} & \frac{\beta}{2} \sum_{j=1}^n u_j + (1-\beta) \left(C^+ \sum_{\{i|y_i=+1\}} \theta_i + C^- \sum_{\{i|y_i=-1\}} \theta_i \right) \\
s.t.: & y_i((\mathbf{w} \cdot \mathbf{x}_i) + b) \geq 1 - Q\theta_i, \quad i = 1, \dots, m \\
& -u_j \leq w_j \leq u_j, \quad j = 1, \dots, n \\
& \theta_j \in [0, 1], \quad j = 1, \dots, n \\
& u_j \geq 0, \quad j = 1, \dots, n.
\end{aligned} \tag{7}$$

Note that (7) allows a different interpretation of C^+ and C^- . These values serve as an abstract measure of error importance in standard SVM (3) used to weight a proxy for classification error. As a rule of thumb the reciprocal of the classes' prior is a prominent choice for these parameters [17-19]. On the contrary, in DSVM we can interpret C^+ and C^- as true cost values in an economical sense.

Consider, e.g., the case of credit scoring. The two error types of predicting a defaulting customer as credit worthy and denying credit to a good risk customer are associated with different costs. While one could incorporate these values or respective estimates directly into (7) their usage in (3) is questionable due to the multiplication with a continuous distance. Therefore, we argue that DSVM might be a better choice for cost-sensitive learning.

C. Extension to non-linear classification

DSVM (7) is a linear classifier. For SVMs, the standard procedure to overcome this limitation is the usage of a kernel function to non-linearly transform data prior to separation. This is accomplished by considering the dual of program (3) where the input data appears only as dot products. Unfortunately, this procedure is not feasible for DSVM: Consider, e.g., problem (6). This differs from the original SVM (3) only by imposing an upper bound on the slack variable. This primal constraint causes a non-linear constraint $\omega_i(-\theta_i + 1) = 0$ in the dual problem (omitted for brevity), with ω_i being the Lagrangian multiplier of the primal constraint. Consequently, the dual cannot be solved efficiently and we will develop a primal algorithm to solve (7) directly.

Following the suggestion of [5] we construct a hierarchy of nested DSVM classifiers, i.e., a decision tree using DSVM as splitting criterion. Hence, the input space is recursively partitioned into disjoint regions by linear hyperplanes and each region is labeled as belonging to class +1 or -1 respectively. This procedure is continued until the tree depth, e.g. the number of nodes, exceeds a user-specified value. In addition, we prohibit splitting a node if either the ratio between minority and majority class examples or the overall number of examples would fall below a user-specified threshold. Our experimental results confirm that these rules are sufficient to prevent over-fitting and that no additional pruning is required.

III. A TABU SEARCH HEURISTIC FOR DSVM

Problem (7) is a LP with continuous (w, b, u) and discrete (θ) decision variables. Obviously, it is computationally much more expensive than (3) due to the integer constraint. Since standard SVM optimization techniques are no longer applicable we develop an alternative approach. As a starting point, we consider the aforementioned relaxation of DSVM replacing the integer variable θ with an upper bounded continuous variable.

$$\begin{aligned}
\min_{w,b,\theta,u} & \frac{\beta}{2} \sum_{j=1}^n u_j + (1-\beta) \left(C^+ \sum_{\{i|y_i=+1\}} \theta_i + C^- \sum_{\{i|y_i=-1\}} \theta_i \right) \\
s.t.: & y_i((\mathbf{w} \cdot \mathbf{x}_i) + b) \geq 1 - Q\theta_i, \quad i = 1, \dots, m \\
& -u_j \leq w_j \leq u_j, \quad j = 1, \dots, n \\
& 0 \leq \theta_j \leq 1, \quad j = 1, \dots, n \\
& u_j \geq 0, \quad j = 1, \dots, n
\end{aligned} \tag{8}$$

Program (8) provides an upper bound for the optimal solution of DSVM and can be solved by fast LP-algorithms [20].

In order to solve (7) we develop a tabu search (TS) algorithm. TS is a meta-heuristic to solve combinatorial optimization problems. The idea is to find a feasible solution and search its neighborhood for better candidates using local hill-climbing strategies. Here, better means higher / lower objective values for maximization / minimization problems. The name TS originates from the fact that the algorithm incorporates some heuristics which prohibit certain moves (tabu moves) to avoid cycling and stops at suboptimal points [21]. For example, consider an optimization problem incorporating only zero-one variables. Let the values of the binary variables be arranged in a bit string of length n . Then, we could define the neighborhood of a given solution to be the set of all adjacent solutions that have a Hamming distance of one from the current solution and consider all moves leading to solutions previously visited in the search as tabu moves; see [22-24] for details.

Our TS implementation is based on the observations that feasible solutions, and consequently the optimal one, for the zero-one problem (7) are contained in the set of extreme points of the relaxation (8); see [6]. Therefore, the extreme points of the polyhedral constraint region defined by (8) form a natural neighbourhood for TS. The general structure of this extreme point tabu search (EPTS) [6, 25] is as follows: 1) Use the simplex method to solve (8) and use the LP-optimal solution e as a starting point for TS. 2) Examine adjacent solutions in the neighbourhood of e . These are all solutions that can be obtained by ordinary simplex pivot operations, e.g. exchanging a current basis variable for a non-basis variable. 3) Select the move that results in the largest improvement of the objective value and is not contained in the tabu list. 4) Execute the selected move and update the tabu list using information on the time a variable is pivoted (recency information) and its overall numbers of pivots (frequency information).

Important design decisions are: the strategy for screening the candidate list in step 2), the move evaluation function and the rules and memory structures for the tabu list.

In general, the optimal solution for the relaxation (8) will not be MIP-feasible so that some of the relaxed θ_i will be between zero and one and. We define the sum of these variables to measure the integer infeasibility (ii) of the given solution (9). Put another way, ii is the amount a given solution fails to fulfil the integer constraint and is zero for all MIP-feasible solutions.

$$ii = \sum_{\{j|0 < j < 1\}} \theta_j. \quad (9)$$

Considering screening the candidate list, each TS move can increase/decrease the current objective value z and increase/decrease the current amount of integer infeasibility ii . Therefore, every pivot operation belongs to one of four elementary types, e.g. increase z and decrease ii (best moves) or decrease z and increase ii (worst moves) [26]. Our TS implementation evaluates each candidate move according to its move type. To resolve situations in which one can either increase z on the cost of increasing ii or decrease ii on the cost of decreasing z we incorporate a strategic oscillation component [6] so that the algorithm strives to improve z for a given number of iterations, then switches to decreasing ii , then switches back to z improvements, etc. This trade-off is illustrated in Fig. 2.

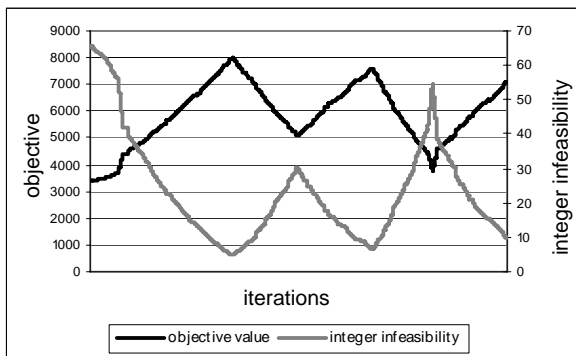


Fig. 2: Trade-off between TS moves that either improve ii or z .

To ensure that the algorithm always finds a solution for the MIP the TS starts with optimizing the integer infeasibility only. Once a MIP-feasible solution is found the strategic oscillation is activated. While the tabu status in our implementation is solely based on recency and frequency information (see above) we use an aspiration criterion to allow moves that lead to a new best feasible MIP solution even if they are currently tabu.

IV. EMPIRICAL STUDY

A. Overview

The empirical evaluation of DSVM strives to benchmark the predictive performance of DSVM in standard as well as cost-sensitive settings. We consider four data sets from the

Statlog project [27] and the UCI machine learning library [28] as a case study. The two credit scoring data sets Australian credit (ac) and German credit (gc) serve as examples from the field of managerial decision making while heart-disease (hrt) and Wisconsin breast cancer (wbc) exemplify cases of medical diagnosis. A brief description of data set characteristics is given in Table 1. Each data set has been partitioned into 2/3 training set for model building and 1/3 test set for out-of-sample evaluation.

TABLE 1:
DATA SET CHARACTERISTICS*

	#cases	#features	#class -1	#class +1
ac	690	14	307	383
gc	1000	24	700	300
hrt	270	13	150	120
wbc	683	10	239	444

* We use the pre-processed data sets that are available via the LIBSVM homepage [29].

Experiments have been conducted in a cluster of 35 workstations. While facilitating fast computation this set-up prohibits direct comparisons of training times due to varying machine configuration. Therefore, we have conducted some preliminary experiments on a single pc using the ac data set. For most parameter settings, the standard SVM classifier could be constructed in less than 30 sec. while DSVM required about 13 mins. However, this poor performance is partially due to using a slow simplex implementation so that we expect substantial improvements when incorporating CPLEX [20] in future work on DSVM, see also Section V.

B. Predictive accuracy of DSVM

Using the four data sets we compare DSVM versus SVM with linear (lin), radial (rad) and polynomial (poly) kernel. Since DSVM is a linear classifier, we use the decision tree approach (Section II.C) to construct hierarchical classifiers with tree depth of two (DSVM-DT2) and three (DSVM-DT3) levels. The balanced error rate (BER) is used to measure predictive accuracy. BER is calculated as:

$$BER = 0.5 \left(\frac{FP}{N_+} + \frac{FN}{N_-} \right), \quad (10)$$

In (10), FP denotes the number of false positive cases (number of false class +1 predictions) while FN (false negatives) measures the opposite error. We use N_+ / N_- to represent the overall number of positive/negative examples within a data set.

Hyperparameters, e.g. β and kernel parameters for standard SVM have been determined by cross-validation (CV) adopting a grid search strategy [3, 30]. For each classifier, the parameter setting providing the lowest 10-fold CV BER is selected and evaluated on the test set. Results are given in Table 2.

TABLE 2:
BALANCED ERROR RATE OF SVM AND DSVM

	Training ¹			Test		
	BER	FP	FN	BER ²	FP	FN
ac						
SVM (lin)	0,13	43	18	0,15	26	10
SVM (rad)	0,14	48	17	0,14	29	6
SVM (poly)	0,09	15	24	0,20	16	28
DSVM	0,13	46	15	0,19	31	8
DSVM-DT2	0,10	26	20	0,19	25	17
DSVM-DT3	0,09	26	15	0,19	28	14
gc						
SVM (lin)	0,32	46	98	0,34	13	66
SVM (rad)	0,32	30	63	0,34	17	63
SVM (poly)	0,32	30	72	0,32	14	61
DSVM	0,30	37	106	0,33	17	70
DSVM-DT2	0,28	37	106	0,31	14	65
DSVM-DT3	0,28	37	106	0,31	14	65
hrt						
SVM (lin)	0,14	12	14	0,21	6	10
SVM (rad)	0,15	11	17	0,22	4	12
SVM (poly)	0,13	10	14	0,20	5	10
DSVM	0,12	8	15	0,20	8	8
DSVM-DT2	0,08	8	7	0,21	7	10
DSVM-DT3	0,05	5	5	0,21	7	10
wbc						
SVM (lin)	0,03	5	8	0,04	3	3
SVM (rad)	0,03	4	9	0,03	2	5
SVM (poly)	0,02	4	4	0,07	7	4
DSVM	0,02	10	4	0,04	7	2
DSVM-DT2	0,02	5	5	0,04	4	3
DSVM-DT3	0,02	5	4	0,04	5	3

¹ Results on the training set are averaged over the ten CV folds.

² We use bold face to highlight the classifier that provides the lowest BER for the respective data set.

From Table 2 we conclude that the predictive performance of DSVM’s is competitive to standard SVM and other classifiers that have been evaluated in previous benchmarking studies on the same data [3, 5, 31]. However, considering the increased complexity of DSVM training a competitive performance is not sufficient to justify the application of this method. In fact, these results suggest that the potential negative effect of approximating classification error in standard SVM is minor. On the other hand, real world applications regularly involve asymmetric misclassification costs and the impact of error approximation might be larger under these constraints. Therefore, we consider cost-sensitivity in the following experiments.

C. Cost-efficiency of DSVM

DSVM allows an explicit integration of misclassification costs. Therefore, we expect it to outperform standard SVMs in cost-sensitive settings and compare them under varying cost-distributions. We consider applications where FP is less severe than FN. Let C^+ denote the cost for a FN error, e.g. classifying a bad credit risk as credit-worthy, and C^- the costs for FP respectively. Without loss of generality we can set C^- to one and scale C^+ accordingly. Obviously, there is some cost asymmetry where the classifier avoids the more expensive error type FN completely and always predicts the positive class. A pre-test revealed that this point is reached at the latest at a ratio of $C^+ : C^- = 50 : 1$ for our data. Conse-

quently, we consider cost distributions from $C^+ : C^- = 2 : 1$ to 50:1 and repeat the previous benchmark (Table 2) for each distribution. Again, free parameters have been determined by 10-fold cross-validation, this time using misclassification costs as performance metric. That is, the classifier providing minimal misclassification costs within 10-fold CV is selected and evaluated on the test set. To increase readability we aggregate our results into five groups of increasing cost asymmetry. Since aggregation prohibits the calculation of real cost values we report the average number of false positive and false negative predictions within each group. Results are presented in Table 3.

TABLE 3:
RESULTS FOR SVM AND DSVM UNDER DIFFERENT COST-DISTRIBUTIONS

Data set	Results for csSVM				Results for DSVM			
	training		Test		training		test	
	FP	N	FP	N	FP	FN	FP	FN
ac								
2:1-5:1	50	10	33	7	50	10	34	6
6:1-10:1	80	0	46	6	73	7	45	4
11:1-15:1	80	0	46	6	113	2	52	3
16:1-20:1	80	0	46	6	113	2	63	2
25:1-50:1*	80	0	46	6	142	1	65	1
gc								
2:1-5:1	170	4	76	26	238	23	71	17
6:1-10:1	350	0	168	2	377	3	148	4
11:1-15:1	360	0	175	1	395	3	162	1
16:1-20:1	360	0	175	1	400	2	168	0
25:1-50:1*	360	0	175	1	400	2	168	0
hrt								
2:1-5:1	20	0	12	7	27	4	13	7
6:1-10:1	20	0	13	6	34	2	16	6
11:1-15:1	20	0	13	6	45	1	21	3
16:1-20:1	20	0	13	6	60	0	28	0
25:1-50:1*	20	0	13	6	60	0	28	0
wbc								
2:1-5:1	0	0	4	4	10	4	4	2
6:1-10:1	0	0	4	4	21	2	9	2
11:1-15:1	0	0	4	4	29	1	12	1
16:1-20:1	0	0	4	4	29	1	12	1
25:1-50:1*	0	0	4	4	29	1	12	1

* We used a step size of 5 to increase asymmetry in cost distributions from a ratio of 20:1 onwards. Consequently, this group contains approximately the same number of elements as the other groups.

Both classifier react to increasing cost asymmetry with reducing the more “expensive” false negative error therewith accepting more false positives. This behavior is desirable and results in overall lower misclassification costs. It is noteworthy that DSVM consistently produces smaller FN errors than SVM which confirms our initial conjecture that this technique is well suited for cost-sensitive learning. Regarding less severe FP errors we cannot observe a clear trend. DSVM is better under some distributions while being inferior to standard SVM in other settings.

We conclude that DSVM is indeed a promising candidate classifier in scenarios involving imbalanced cost-distributions. This makes the technique particularly well suited for classification in medical settings. Clearly, missing a positive result when detecting tumors from Magnetic Resonance Imaging scans might induce dramatic conse-

quences while a small number of false alarms is tolerable if the scans are subsequently re-screened by medical personal. Therefore, a small reduction of FN errors can be a substantial improvement and very well compensate an increase in training times.

V. CONCLUSIONS

We argued that the approximate handling of errors in standard SVM might lead to inaccurate predictions. To verify this assumption we considered the DSVM classifier and proposed a new tabu search heuristic for solving the resulting MIP which exploited the specific structure of the DSVM training problem.

Empirical comparisons could not confirm previous findings regarding the superiority of DSVM over standard SVM [5]. However, while DSVM gave competitive results in standard settings it turned out to be a better choice when one error type is more severe than another. This makes DSVM a promising candidate for classification in medical diagnosis. Further more, business application, e.g. in the field of direct marketing, with severely asymmetric cost-distributions would benefit from this behaviour.

We will conduct an extended benchmarking study of DSVM in respective scenarios in future research. In addition, we plan the development of a novel heuristic for DSVM training, hybridizing elements from tabu search with a stochastic component. In this context we will investigate the solution space of DSVM and the path of TS through this space in more detail. In particular, the appropriateness of using an LP-optimal solution as starting point for the meta-heuristic requires further clarification. While we adopted this approach from the literature [6] some experiments suggest that it might be detrimental for overall training speed since the search oscillates back and forth in a region of attraction around the LP-optimal solution.

REFERENCES

- [1] V. Vapnik, S. Kotz, Estimation of Dependences Based on Empirical Data, 2.ed., Springer, New York, 2006.
- [2] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, 2.ed., Wiley, New York, 2001.
- [3] B. Baesens, T. Van Gestel, S. Viaene, M. Stepanova, J. Suykens, J. Vanthienen, Benchmarking state-of-the-art classification algorithms for credit scoring, *Journal of the Operational Research Society* 54 (6) (2003), pp. 627-635.
- [4] A. Bertoni, R. Folgieri, G. Valentini, Bio-molecular cancer prediction with random subspace ensembles of support vector machines, *Neurocomputing* 63 (2005), pp. 535-539.
- [5] C. Orsenigo, C. Vercellis, Discrete support vector decision trees via tabu search, *Computational Statistics & Data Analysis* 47 (2) (2004), pp. 311-322.
- [6] A. Lokketangen, F. Glover, Solving zero-one mixed integer programming problems using tabu search, *European Journal of Operational Research* 106 (2-3) (1998), pp. 624-658.
- [7] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and other Kernel-based Learning Methods, Cambridge University Press, Cambridge, 2000.
- [8] V.N. Vapnik, The Nature of Statistical Learning Theory, Springer, New York, 1995.
- [9] Veropoulos, N. Cristianini, C. Campbell, Controlling the Sensitivity of Support Vector Machines. In: Proc. of the 16th Intern. Joint Conf. on Artificial Intelligence, 1999, pp. 55-60.
- [10] Y. Lin, Y. Lee, G. Wahba, Support vector machines for classification in nonstandard situations, *Machine Learning* 46 (1-3) (2002), pp. 191-202.
- [11] J.C. Platt, Fast Training of Support Vector Machines using Sequential Minimal Optimization. In: B. Schölkopf, C.J.C. Burges, A.J. Smola (Eds.), *Advances in Kernel Methods: Support Vector Learning*, MIT Press, Cambridge, 1999, pp. 185-208.
- [12] T. Joachims, Making Large-Scale SVM Learning Practical. In: B. Schölkopf, C.J.C. Burges, A.J. Smola (Eds.), *Advances in Kernel Methods: Support Vector Learning*, MIT Press, Cambridge, 1999, pp. 169-184.
- [13] E.E. Osuna, F. Girosi, Reducing the Run-time Complexity in Support Vector Machines. In: B. Schölkopf, C.J.C. Burges, A.J. Smola (Eds.), *Advances in Kernel Methods: Support Vector Learning*, MIT Press, Cambridge, 1999, pp. 271-284.
- [14] A.H. Land, A.G. Doig, An automatic method for solving discrete programming problems, *Econometrica* 28 (1960), pp. 497-520.
- [15] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, B. Schölkopf, An introduction to kernel-based learning algorithms, *IEEE Transactions on Neural Networks* 12 (2) (2001), pp. 181-201.
- [16] K.P. Bennet, O.L. Mangasarian, Robust linear programming discrimination of two linearly inseparable sets, *Optimization Methods and Software* 1 (1992), pp. 23-24.
- [17] K.P. Bennett, S. Wu, L. Auslender, On Support Vector Decision Trees for Database Marketing. In: Proc. of the Intern. Joint Conf. on Neural Networks, 1999, pp. 904-909.
- [18] P. Bradley, O. Mangasarian, W. Street, Feature selection via mathematical programming, *INFORMS Journal on Computing* 10 (2) (1998), pp. 209-217.
- [19] C.-W. Hsu, C.-C. Chang, C.-J. Lin, A Practical Guide to Support Vector Classification Department of Computer Science and Information Engineering National Taiwan University, Taipei, Taiwan, 2003 (www.csie.ntu.edu.tw/~cjlin/guide/guide.pdf).
- [20] ILOG, CPLEX Library, 2006 (www.ilog.com/products/cplex).
- [21] F. Glover, Tabu search - wellsprings and challenges, *European Journal of Operational Research* 106 (2-3) (1998), pp. 221-225.
- [22] F. Glover, M. Laguna, *Tabu Search*, Kluwer, Boston, 1997.
- [23] F. Glover, Tabu search: part 1, *ORSA Journal on Computing* 1 (3) (1989), pp. 190-206.
- [24] F. Glover, Tabu search: part 2, *ORSA Journal on Computing* 2 (4) (1990), pp. 4-32.
- [25] J.A. Blue, K.P. Bennett, Hybrid extreme point tabu search, *European Journal of Operational Research* 106 (2-3) (1998), pp. 676-688.
- [26] A. Lokketangen, F. Glover, Candidate List and Exploration Strategies for Solving 0/1 MIP Problems using a Pivot Neighborhood. In: S. Voß, S. Martello, I.H. Osman, C. Roucairol (Eds.), *Meta-Heuristics. Advances and Trends in Local Search Paradigms for Optimization*, Kluwer, Boston, 1999.
- [27] D. Michie, D.J. Spiegelhalter, C.C. Taylor, *Machine Learning, Neural and Statistical Classification*, Horwood, New York, 1994.
- [28] D.J. Newman, S. Hettich, C.L. Blake, C.J. Merz, *UCI Repository of Machine Learning Databases*. Department of Information and Computer Science, University of California, Irvine, CA, 1998.
- [29] C.-C. Chang, C.-J. Lin, LIBSVM - A Library for Support Vector Machines, 2001 (www.csie.ntu.edu.tw/~cjlin/libsvm).
- [30] T. van Gestel, J.A.K. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B. de Moor, J. Vandewalle, Benchmarking least squares support vector machine classifiers, *Machine Learning* 54 (1) (2004), pp. 5-32.
- [31] T.-S. Lim, W.-Y. Loh, Y.-S. Shih, A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms, *Machine Learning* 40 (3) (2000), pp. 203-228.