Support Vector Pruning with SortedVotes for Large-Scale Datasets

Frerk Saxen, Konrad Doll and Ulrich Brunsmann
University of Applied Sciences Aschaffenburg, Germany
Email: {Frerk.Saxen, Konrad.Doll, Ulrich.Brunsmann}@h-ab.de

***Abstract*—High computational costs and large memory requirements which are particularly relevant in embedded systems often result from a large support vector (SV) set of a non linear kernel support vector machine (SVM). We propose effective and straight forward methods, an improved RANSAC-SVM algorithm, called *RANSAC-SV,* and a novel algorithm *SortedVotes,* for pruning support vectors of a non linear kernel support vector machine. We introduce a lower bound for the number of SVs defined by the classification accuracy of a linear SVM. Like RANSAC, our algorithm SortedVotes adjusts a parameter set in terms of an optimization goal. Thus, its application is not restricted to support vector pruning. Applied to SVMs, SortedVotes prunes an SV set initially obtained after training by weighting each SV with respect to a performance measure of the SVM classification and by sorting the SVs according to this weight. The final SV set contains the SVs with optimal weights. Taking the RANSAC-SVM algorithm as a reference, we evaluate our modified version RANSAC-SV and SortedVotes on the INRIA- and on the MNIST-database using a radial basis function (RBF) kernel. SortedVotes outperforms both algorithms at the lower bound for both databases. Whereas in our experiments RANSAC-SVM does not at all prune SVs without loss in classification accuracy, for the MNIST database, lossless pruning rates of $10\,\%$ and $65\,\%$ are obtained with RANSAC-SV and SortedVotes, respectively.**

***Index Terms*—Support Vector Machine (SVM); Pruning; Support Vector Reduction; RANSAC**

I. Introduction

SVMs have been widely accepted as a powerful method for real-world classification problems. For many applications SVMs have proven robust detection and high generalization ability, such as optical character recognition [1], [2], breast cancer detection [3], [4], face- and pedestrian-detection [5], [6]. During a training phase the decision boundary is determined directly from the training data by maximizing the separating margin of the decision boundary. The decision boundary, which depends on the SV set, is used in the test phase for classification.

SVMs can be divided into two families depending on the kernel function they use. In a linear kernel SVM (LK-SVM) the decision function to be evaluated during the test phase is just a scalar product, because all SVs can be linearly combined into a single weight vector. However, in non linear kernel SVMs (NLK-SVMs), which are used to map nonlinear decision boundaries into a higher dimensional space for linear separation [7], the computation of the decision function is not as computational effective as with an LK-SVM. Unfortunately in this case all SVs have to be taken into account individually.

978-1-61284-144-1/11/$26.00 ©2011IEEE

Due to the usual large number of SVs NLK-SVMs are considerably slower in the test phase than other learning machines such as neural network and decision trees [1], [2], [8], [9]. Computation time and memory requirements to classify an unknown sample are particularly relevant in embedded systems, e.g. for advanced driver assistance systems [10], [11]. Both quantities are directly proportional to the number of SVs. Additionally, potential overfitting often results from a large SV set, [12]–[15]. Pruning methods decrease the number of SVs yielding in better computational performance to classify unknown samples, higher generalization ability, and thus less overfitting.

Several approaches for SV pruning have been proposed. In [9], an approximation of the decision boundary in terms of a reduced set leads to less vectors. These vectors are no more SVs and decrease the computation time up to ten times. A similar approach reports up to twenty times cpu-time reduction [16]. In [17] a reduction of support vectors is achieved for LK-SVMs and for NLK-SVMs with polynomial and RBF kernel by eliminating linearly dependent SVs and adapting Lagrange multipliers. Furthermore, [4] excludes SVs which make the decision boundary highly convoluted and retrains with the remaining subset an SVM in a next step .

In [18], adjacent SVs belonging to the same class are replaced by a newly constructed vector. In [19], the RANSAC-SVM algorithm is used to exclude SVs. Another effective method from Liang [20] clusters SVs and performs orthogonal projections to find vectors that can be removed by others.

In this work we prune SVs and thereby reduce the computation time and the memory requirements of an NLK-SVM with respect to real-time embedded applications. Referring to this the classification accuracy of an LK-SVM defines a lower bound for reducing the number of SVs in an NLK-SVM.

To verify this behavior and to demonstrate the qualitiy of the proposed algorihms we present the results of the original RANSAC-SVM, the improved RANSAC-SV algorithm and our novel SV pruning method SortedVotes. Like RANSAC-SVM [19] SortedVotes is a genetic and iterative algorithm which estimates parameters of a mathematical model. The RANSAC-SVM algorithm iteratively selects random samples, estimates model parameters, calculates the classification rate and retains or rejects the samples according to the classification rate. If some samples are contaminated and therefore rejected, the RANSAC-SVM algorithm does not use this information in further iterations. Our SortedVotes algorithm,

however, weights each sample according to the classification rate and uses this valuable information later on.

The remainder of the paper is organized as follows. In Section II the underlying basics of SVMs are given. Section III describes the pruning methods RANSAC-SVM, RANSAC-SV and SortedVotes. In Section IV experimental results are presented. Section V summarizes the conclusions.

II. Support Vector Machines

SVMs belong to statistical classification methods [8], [13]. Given N training samples $\mathbf{x}_m$ with corresponding class labels y_m the set

$$\mathbf{X} = \{(\mathbf{x}_m, y_m) | \mathbf{x}_m \in \mathbb{R}^P, y_m \in \{-1, 1\}, m = 1 \dots N\} \quad (1)$$

denotes the training set. During the training phase a decision boundary that has a maximimum distance to the separating margin is generated. Unknown samples $\mathbf{u}$ can be classified with the decision function as follows:

$$F(\mathbf{u}) = sgn\left(b + \sum_{m=1}^{N} \alpha_m \cdot y_m \cdot K(\mathbf{x}_m, \mathbf{u})\right) \quad (2)$$

The parameters determined during the training phase are the Lagrange multipliers α_m and the bias b. Those samples $\mathbf{x}_m$ with nonzero Lagrange multiplier α_m are called "support vectors". $K(\mathbf{x}_m, \mathbf{u})$ is a positive semidefinite kernel function which may implicitly map the samples to a higher dimensional feature space. A widely used non linear kernel function of an NLK-SVM is the radial basis function (RBF): $K(\mathbf{x}_m, \mathbf{u}) = e^{-\gamma \cdot ||\mathbf{x}_m - \mathbf{u}||^2}$, where $||.||$ denotes the Euclidean norm and γ is a positive parameter controlling the curvature of the hyperplane.

For the linear kernel function $K(\mathbf{x}_m, \mathbf{u}) = \mathbf{x}_m^T \cdot \mathbf{u}$ we get an LK-SVM. The decision function can be simplified as follows:

$$F(\mathbf{u}) = sgn\left(\mathbf{w}^T \cdot \mathbf{u} + b\right) \quad (3)$$

$$\mathbf{w} = \sum_{m=1}^{N} \alpha_m \cdot y_m \cdot \mathbf{x}_m \quad (4)$$

III. Pruning Methods

In this section we present the details of the RANSAC-SVM algorithm, our improved version, RANSAC-SV, and our novel algorithm SortedVotes. The ratio between the number of SVs after pruning and the number of SVs before pruning defines the pruning rate. As validation criteria we use the classification rate C_r which is equivalent to classification accuracy and defined as ratio of the sum of true positives TP and true negatives TN to the total number of positive P and negative N samples.

$$C_r = \frac{TP + TN}{P + N} \quad (5)$$

A. RANSAC-SVM

The RANSAC-SVM [19] implementation starts with the whole training set $\mathbf{X}$. Subsets $\mathbf{X_l}$, $l = 1, 2, \dots, L$ of the training data (containing pos. and neg. examples) are randomly chosen from the training set. With each subset $\mathbf{X_l}$ an NLK-SVM is trained including variation of parameters (e.g. grid search for the margin parameter C and γ of a RBF kernel SVM) and validated by the complete training set $\mathbf{X}$. Out of two randomly selected subsets (e.g. $\mathbf{X_3}$ and $\mathbf{X_8}$) with classification rate C_r as selection weights two new subsets are generated by randomly selecting samples (e.g. $\mathbf{X_{\hat{3}}}$ and $\mathbf{X_{\hat{8}}}$, $\mathbf{X_{\hat{3}}} \subset \mathbf{X_3} \cup \mathbf{X_8}$, $\mathbf{X_{\hat{8}}} \subset \mathbf{X_3} \cup \mathbf{X_8}$, $\mathbf{X_{\hat{3}}} \cap \mathbf{X_{\hat{8}}} = \emptyset$). This is done until all subsets are processed. The new subsets form the basis for the next iteration. Again with each subset an SVM is trained by variation of parameters. The SVM with the highest classification rate C_r becomes the final classifier.

B. RANSAC-SV

Since the number of iterations of the RANSAC-SVM implementation in [19] is deterministic and does not depend on the evaluation results, RANSAC-SVM does not really apply the RANSAC algorithm which is probabilistic. We instead use the original RANSAC algorithm [21] and apply it to the SVs and not to the whole training set. This decreases the computation time without loosing classification performance.

Initially, an NLK-SVM T_0 is trained based on the training set $\mathbf{X}$ and optimized by variation of parameters. The SVs and the corresponding class labels obtained by the training step form the vector set $\mathbf{S}$ which is a subset of the training set $\mathbf{X}$. Depending on the desired pruning rate, M vectors are randomly selected from vector set $\mathbf{S}$ and used to train an NLK-SVM T_1 using the parameters of T_0. The SVM T_1 is validated by the vector set $\mathbf{S}$. This is done iteratively by randomly selecting new M vectors from the vector set $\mathbf{S}$ to train T_i. The classification rate C_r is taken as consensus measure to determine the number of iterations (according to RANSAC [21]). Since we apply the RANSAC algorithm to the SVs instead of the whole training set, we avoid dealing with vectors which are far away from the decision boundary and thus do not affect the decision function. Furthermore, dealing with SVs decreases the computation time for large scale datasets.

The algorithm is described briefly as follows:

- Train NLK-SVM on training set $\mathbf{X}$ and search optimal SVM parameters (e.g. C_{opt} and γ_{opt} for RBF kernel). To decrease the computation time a representative subset of the training set is selected.
- Train NLK-SVM T_0 with optimal parameters (e.g. C_{opt} and γ_{opt}) on complete training set $\mathbf{X}$ and obtain the N_S SVs and class labels which build the vector set $\mathbf{S} \subset \mathbf{X}$.
- Choose number of desired SVs M which is inversely proportional to the desired pruning rate.
- Preset initial number of iterations I (just a conservative guess), set $i = 1$.
- *While* $i \leq I$

1) Randomly select M vectors and corresponding labels from $\mathbf{S}$ to train an NLK-SVM T_i (use the above determined parameters).
2) Evaluate the NLK-SVM on vector set $\mathbf{S}$ by calculating the classification rate.
3) Adjust I (number of iterations): $I = \frac{log(0.05)}{log(1-(C_r)^M)}$
4) $i = i + 1$

- Adopt SVM with highest classification rate as reduced SVM model.

C. SortedVotes

As in the RANSAC-SV algorithm initially an NLK-SVM T_0 is trained and optimized by variation of parameters. The SVs obtained from the training step form the vector set $\mathbf{S}$ of size N_S which is a subset of the training set $\mathbf{X}$. The SVs in $\mathbf{S}$ are arbitrarily arranged. In the first step, depending on the desired pruning rate, the first M samples of $\mathbf{S}$ are selected to train an NLK-SVM T_1 using the determined parameters from T_0. The trained SVM T_1 is validated by the vector set $\mathbf{S}$. The first M samples of $\mathbf{S}$ are weighted (voted) by the validation criteria (e.g. classification rate C_r). The next M samples (step size = M) are selected to train another NLK-SVM T_2 and are weighted with the validation criteria. The higher the weight, the higher the propability that a sample is uncontaminated. If the end of the vector set $\mathbf{S}$ is reached, the vector set is sorted according to the weights to enhance uncontaminated samples. The following steps are quite similar to the first step, only the step size selecting samples from vector set $\mathbf{S}$ is modified. In the second step we use $\frac{M}{2}$, in the third $\frac{M}{3}$, and so on. Increasing the number of steps increases the accuracy but also the computation time.

The algorithm is described briefly as follows:

- Train NLK-SVM on training set $\mathbf{X}$ and search optimal SVM parameters (e.g. C_{opt} and γ_{opt} for RBF kernel SVM). To decrease the computation time a representative subset of the training set can be selected.
- Train NLK-SVM T_0 with optimal parameters (e.g. C_{opt} and γ_{opt}) on complete training set $\mathbf{X}$ and obtain the N_S SVs and class labels which form the vector set $\mathbf{S} \subset \mathbf{X}$.
- Choose number of desired SVs M which is inversely proportional to the desired pruning rate.
- Initialize votes $\mathbf{v}$ with $\vec{0}$, set $i = 1$.
- *For* $l = 1$ *to* 5
  - Sort vector set $\mathbf{S}$ according to corresponding vector $\mathbf{v}$.
  - *For* $P = 1$ *to* $N_S - M$ *step* $\frac{M}{l}$
    1) Select M samples from $\mathbf{S}$ (from P to $P + M$) to train an NLK-SVM T_i (use the optimal SVM parameters from T_0).
    2) Validate T_i on $\mathbf{S}$ by calculating a validation criteria (e.g. classification rate C_r).
    3) Weight all selected samples with respect to the validation criteria and save these weights in $\mathbf{v}$.
    4) $i = i + 1$
- Adopt SVM with best validation criteria as reduced SVM model.

IV. Experimental Results

In our study we evaluate the RANSAC-SVM, RANSAC-SV and SortedVotes algorithms with two different datasets: the INRIA dataset for pedestrian detection and the MNIST dataset for character recognition. We trained our SVMs with SVM^{light} [22].

A. Lower bound for pruning SVs

In literature often pruning rates are described for a loss in classification performance of e.g. $1\,\%$ to compare different pruning methods and to present pruning results. Even a "minor" loss in absolute classification performance (e.g. $1\,\%$), however, may fall below the classificataion performance of the LK-SVM. Since LK-SVMs combine Lagrange multipliers with SVs to a single weight vector $\mathbf{w}$ (see Eq. 4), this classification performance indicates maximum meaningful decrease in classification performance for pruning methods considering memory requirements and speed of classification. With respect to different datasets, kernels and descriptors we propose to compare the decrease in classification results of NLK-SVMs after pruning with the linear classification results.

B. Results on INRIA dataset

The INRIA dataset, introduced by Dalal and Triggs [6], is a pedestrian database with $14,596$ training and $5,656$ testing images. We use histogram of oriented gradients (HOG) for feature extraction. With the Dalal and Triggs setting our descriptor has a length of $3,780$ elements for each image.

The LK-SVM (dashed horizontal line in Fig. 1) is trained with $C = 0.03118$ and limits the classification rate to $98.57\,\%$ for SV pruning of an NLK-SVM. It is only $0.62\,\%$ worse than the original RBF kernel SVM ($C = 256$ and $\gamma = 0.046997$) with a classification rate of $99.19\,\%$. The SortedVotes algorithm, our RANSAC-SV algorithm, and the RANSAC-SVM algorithm reach the classification rate of the LK-SVM at 394, 475, and 765 SVs, which label maximum reasonable pruning rates ($83.0\,\%$, $79.5\,\%$, and $66.9\,\%$) for the INRIA dataset.

Liang [20] evaluated ten different datasets and obtained avarage pruning rates of about $45\,\%$ with a loss of $1.8\,\%$ in classification rate. We show for the INRIA dataset in Fig. 1 that already a loss of $0.62\,\%$ results in linear classification results. The acceptable loss in classification rate obviously depends on the dataset. It may also depend on the descriptors chosen.

The original SVM has $2,311$ SVs. When pruning SVs, no loss in classification performance is reached at $2,137$ SVs with our RANSAC-SV algorithm and $2,221$ SVs with the SortedVotes algorithm ($7.5\,\%$ and $3.9\,\%$ pruning rates). The RANSAC-SVM algorithm does not at all reach lossless classification performance.

C. Results on MNIST dataset

The MNIST database of handwritten digits [23] has a training set of $60,000$ examples and a test set of $10,000$

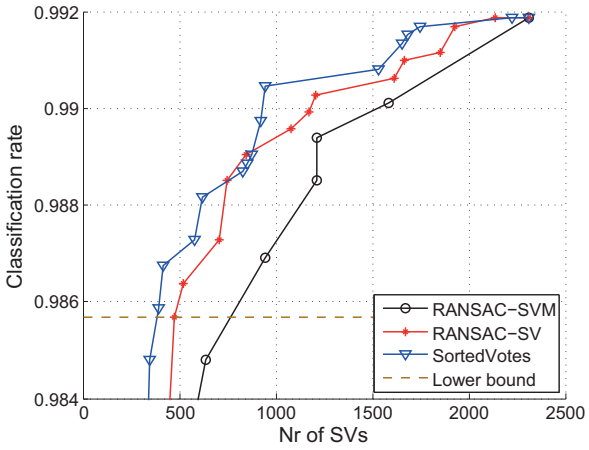

Fig. 1. Pruning results on INRIA dataset: classification rate as a function of the number of SVs. The graphs present the maximum performance of the algorithms. Results with more SVs at lower classification rate than others are not shown, since they would not be used in practice. The lower bound for pruning SVs is given by the linear classification results.

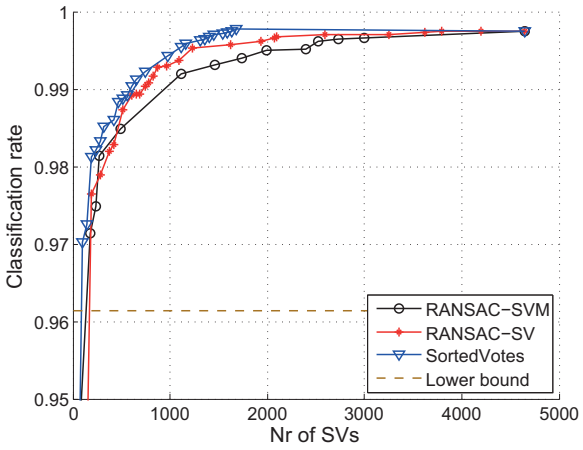

Fig. 2. Pruning results on MNIST dataset relabled to the class of digit three and others: classification rate as a function of the number of SVs. The graphs present the maximum performance of the algorithms. Results with more SVs at lower classification rate than others are not shown, since they would not be used in practice. The lower bound for pruning SVs is given by the linear classification results.

examples. We generate a two class classification problem by introducing the classes "digit three" and "non digit three". We scale each pixel in the image to the interval $[-1, 1]$ and concatenate the pixles to a descriptor with $28 \times 28 = 784$ elements.

The LK-SVM (dashed horizontal line in Fig. 2) is trained with $C = 6.268844221105526 \cdot 10^{-6}$ and limits the classification rate to $96.14\,\%$ for SV pruning. It is $3.61\,\%$ worse than the original RBF kernel SVM ($C = 2.82842712475$ and $\gamma = 0.00728932024638$ proposed by [24]) with $99.75\,\%$ in classification rate. The SortedVotes algorithm, the RANSAC-SV algorithm, and the RANSAC-SVM algorithm reach the classification rate of the LK-SVM at 97, 193, and 180 SVs which label maximum reasonable pruning rates ($97.9\,\%$, $95.8\,\%$, and $96.1\,\%$).

The original SVM has $4,649$ SVs. No loss in classification performance is reached at $1,631$ SVs with the SortedVotes algorithm and $4,203$ SVs with the RANSAC-SV algorithm ($64.9\,\%$ and $9.6\,\%$ pruning rates). Again, the RANSAC-SVM algorithm does not at all reach lossless classification performance.

As well Fig. 1 as Fig. 2 show that our SortedVotes and RANSAC-SV implementations deliver better results than the RANSAC-SVM algorithm.

We evaluated the SortedVotes, the RANSAC-SV, and the RANSAC-SVM algorithm on a desktop PC (Windows 7 with Intel(R) Core(TM) $i7$ CPU 930 with $2.8\,$GHz and $6\,$GB RAM). Table I shows the average calculation time per pruning procedure. The SortedVotes algorithm is more than 7 times faster than the RANSAC-SV algorithm and more than 73 times faster than the RANSAC-SVM algorithm.

SortedVotes	RANSAC-SV	RANSAC-SVM
$1,123\,$s	$8,360\,$s	$82,020\,$s
$\approx 19\,$min	$\approx 2.3\,$h	$\approx 22.8\,$h

TABLE I
AVERAGE CALCULATION TIME PER PRUNING PROCEDURE

V. CONCLUSIONS

In this paper we present effective and straight forward methods for pruning SVM classifiers. With respect to linear classification results our proposed method SortedVotes yields feasible pruning rates for the MNIST database between $65\,\%$ and $98\,\%$. A lossless pruning rate of $65\,\%$ is obtained with SortedVotes for this database. The computation time of the SortedVotes algorithm is 7 times faster than the RANSAC-SV implementation and more than 73 times faster than the RANSAC-SVM implementation. Just like the RANSAC algorithm SortedVotes estimates parameters of a mathematical model and thus can be adopted to other tasks than SV pruning. Experiments on the INRIA and the MNIST dataset show that linear classification results limit pruning rates. Overall SortedVotes provides better results than our RANSAC-SV and RANSAC-SVM implementation.

ACKNOWLEDGMENT

This work results from the joint project Ko-PER, which is part of the project initiative Ko-FAS, and has been funded by the German *Bundesministerium für Wirtschaft und Technologie* (Federal Department of Commerce and Technology) under grant number 19S9022B.

REFERENCES

[1] Y. Lecun, L. D. Jackel, H. A. Eduard, N. Bottou, C. Cartes, J. S. Denker, H. Drucker, E. Sackinger, P. Simard, and V. Vapnik, "Learning

algorithms for classification: A comparison on handwritten digit recognition," in *Neural Networks: The Statistical Mechanics Perspective*. World Scientific, 1995, pp. 261–276.

[2] C.-L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, "Handwritten digit recognition: benchmarking of state-of-the-art techniques," *Pattern Recognition*, vol. 36, no. 10, pp. 2271–2285, 2003.

[3] Y. A. Rejani and S. Selvi, "Early detection of breast cancer using svm classifier technique," *International Journal on Computer Science and Engineering*, vol. 1, pp. 127–130, 2009.

[4] Y. Zhan and D. Shen, "Design efficient support vector machine for fast classification," *Pattern Recognition*, vol. 38, no. 1, pp. 157–161, January 2005.

[5] E. E. Osuna, R. Freund, and F. Girosi, "Support vector machines: Training and applications," Massachusetts Institute of Technology Artificial Intelligence Laboratory and Center for Biological and Computational Learning Department of Brian and Cognitive Sciences, Tech. Rep., 1997.

[6] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. Washington, DC, USA: IEEE Computer Society, 2005, pp. 886–893.

[7] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000.

[8] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *5th Annual ACM Workshop on Computational Learning Theory*. ACM Press, 1992, pp. 144–152.

[9] C. J. Burges, "Simplified support vector decision rules," in *13th International Conference on Machine Learning*, 1996, pp. 71–77.

[10] T. P. Cao and G. Deng, "Real-time vision-based stop sign detection system on fpga," in *Digital Image Computing: Techniques and Applications*. IEEE Computer Society, 2008, pp. 465–471.

[11] S. Bauer, S. Koehler, K. Doll, and U. Brunsmann, "Fpga-gpu architecture for kernel svm pedestrian detection," in *CVPRW*. IEEE, 2010, pp. 61–68.

[12] C. J. C. Burges and B. Schölkopf, "Improving the accuracy and speed of support vector learning machines," in *Advances in Neural Information Processing Systems 9*. MIT Press, 1997, pp. 375–381.

[13] V. N. Vapnik, *The nature of statistical learning theory*, M. Jordan, J. F. Lawless, S. L. Lauritzen, and V. Nair, Eds. Springer-Verlag New York, Inc., 1995.

[14] E. E. Osuna and F. Girosi, *Advances in Kernel Methods - Support Vector Learning*. MIT-Press, 1999, ch. 16, pp. 271–284.

[15] X. Zeng and X. wen Chen, "Smo-based pruning methods for sparse least squares support vector machines," *IEEE Transactions on Neural Networks*, vol. 16, no. 6, pp. 1541–1546, 2005.

[16] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Massachusetts Institute of Technology, 2002.

[17] T. Downs, K. E. Gates, and A. Masters, "Exact simplification of support vector solutions," *Journal of Machine Learning Research*, vol. 2, pp. 293–297, 2001.

[18] D. Nguyen and T. Ho, "An efficient method for simplifying support vector machines," in *International Conference on Machine Learning*. ACM Press, 2005, pp. 617–624.

[19] K. Nishida and T. Kurita, "Ransac-svm for large-scale datasets," in *19th International Conference on Pattern Recognition (ICPR 2008)*, Tampa, FL, 2008, pp. 1–4.

[20] X. Liang, "An effective method of pruning support vector machine classifiers," *IEEE Transactions on Neural Networks*, vol. 21, no. 1, pp. 26–38, 2010.

[21] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Graphics and Image Processing*, vol. 24, pp. 381–395, 1981.

[22] T. Joachims, *Advances in Kernel Methods - Support Vector Learning*. MIT-Press, 1999, ch. 11, pp. 169–184.

[23] Y. LeCun and C. Cortes, "The mnist database of handwritten digits." [Online]. Available: http://yann.lecun.com/exdb/mnist/

[24] A. Müller, "Peekaboo - andy's computer vision and machine learning blog," thursday, September 23, 2010. [Online]. Available: http://peekaboo-vision.blogspot.com/2010/09/mnist-for-ever.html