Towards Understanding the Influence of SVM Hyperparameters

Charl J. van Heerden
Human Language Technology Competency Area
CSIR Meraka Institute
Pretoria 0001, South Africa
Email: cvheerden@csir.co.za

Etienne Barnard
Multilingual Speech Technologies Group
North-West University
Vanderbijlpark 1900, South Africa
Email: etienne.barnard@nwu.ac.za

***Abstract*—We investigate the relationship between SVM hyperparameters for linear and RBF kernels and classification accuracy. The process of finding SVM hyperparameters usually involves a gridsearch, which is both time-consuming and resource-intensive. On large datasets, 10-fold cross-validation grid searches can become intractable without supercomputers or high performance computing clusters. We present theoretical and empirical arguments as to how SVM hyperparameters scale with N, the amount of learning data. By using these arguments, we present a simple algorithm for finding approximate hyperparameters on a reduced dataset, followed by a focused line search on the full dataset. Using this algorithm gives comparable results to performing a grid search on complete datasets.**

I. Introduction

The Support Vector Machine (SVM) is a popular pattern recognition algorithm, first introduced in its current form in 1995 [1]. It entails the optimization of the following error function:

$$E_{svm} = \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + C\sum_i \xi_i \tag{1}$$

subject to the constraints

$$\begin{aligned} &y_i(\boldsymbol{\omega}^T\boldsymbol{x}_i + \omega_0) \geq 1 - \xi_i \\ &\xi_i \geq 0, i = 1, ..., n \end{aligned} \tag{2}$$

The dot product in Eq. (1) lends itself to the use of the Kernel trick:

$$\frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} = \sum_{i\in SV}\sum_{j\in SV}\alpha_i\alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{3}$$

Depending on which kernel K is used, SVMs usually have one or two hyperparameters which need to be set to appropriate values in order to achieve optimal classification accuracy. Choosing appropriate values can be a very expensive process, as one needs to iterate over a wide range of parameters or combinations thereof. For very large datasets such as the DFKI age-classification dataset, full grid-searches are intractable without large computing clusters [2], and can thus be a prohibiting factor in using SVMs.

Much work has been done to circumvent the expensive grid search approach by finding more efficient ways of choosing the SVM hyperparameters [3]–[7]. We continue this search by considering the problem from two different perspectives. (a) We believe that a better understanding of the role of the SVM hyperparameters can enable one to select better boundaries within which to search for the optimal hyperparameters and (b) we consider a simple algorithm where we use scaling arguments derived from the SVM error function to adapt hyperparameters obtained on a subset of the data.

The paper is organized as follows: In section II, we will give a brief overview of the role of the different SVM hyperparameters, and then investigate SVM behaviour over a wide range of hyperparameter values in section III. The relationship between the hyperparameters and the amount of training data, as well as a novel hyperparameter tuning strategy will be discussed in section IV, followed by experiments in section IV-D, testing the proposed tuning strategy.

II. Role of RBF and linear kernel hyperparameters

Two popular kernels and their corresponding hyperparameters will be discussed: the linear and radial basis function (RBF) kernels.

The linear SVM has no kernel parameters, hence the only parameter to be tuned is C from Eq. (1). C penalizes samples that are either misclassified, or which fall within the margin surrounding the separating hyperplane. High values of C would thus give more weight to the misclassification term in Eq. (1). Very large values of C would thus change the behaviour of an SVM to that of a perceptron algorithm, since the emphasis shifts to minimizing the sum of all errors. Small values of C give more weight to the margin term, with $C \to 0$ ensuring that the margin gets maximized ($C = 0$ is not sensible for non-separable problems).

The same arguments for C apply to the case where an RBF kernel is used. The RBF kernel

$$K(\boldsymbol{x_i}, \boldsymbol{x_j})_{RBF} = e^{-\gamma\|\boldsymbol{x_i}-\boldsymbol{x_j}\|^2} \tag{4}$$

allows the SVM to construct non-linear decision boundaries by transforming the data into some high dimensional feature space. In addition to C, one has to search for optimal values of γ, which controls the kernel width.

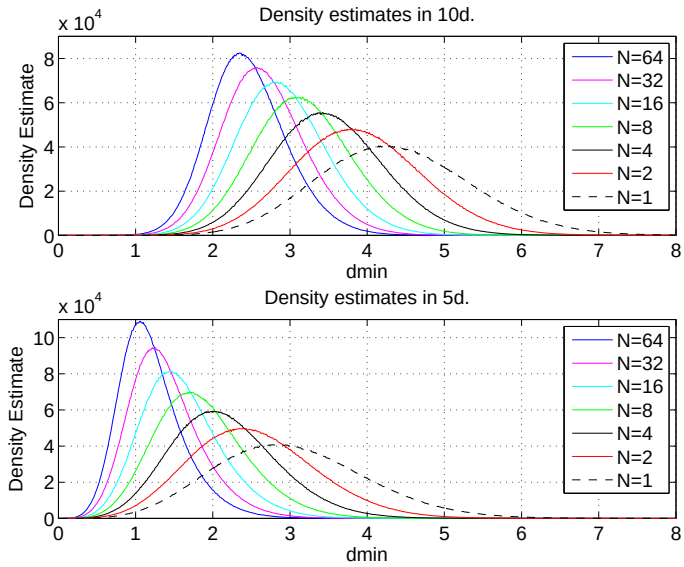


Fig. 2. Density estimates for the distance to the nearest neighbor when randomly sampling N points from a 5 and 10 dimensional normal distribution with zero mean and unit variance. Note the weak relationship between γ and N.

III. SVM BEHAVIOUR ACROSS A LARGE SPECTRUM OF HYPERPARAMETER VALUES

Keerthi et. al [8] investigated SVM behaviour at very small and large values of the SVM hyperparameters. In this section, we will extend and verify some of the insights they presented with the aim of identifying reasonable boundaries within which one could expect to find the optimal hyperparameter values.

A. Linear Kernels

Keerthi et. al observed that for linear SVMs, after some sufficiently high value of $C > C^*$, the cross-validation accuracy seems to converge to a value close to (if not at the) optimal accuracy. This implies that as $C \to \infty$, $E_{SVM} \to C \sum_i \xi_i$. We repeated and confirmed this observation on a number of datasets, as displayed in Fig. 3. The results also indicate that $C \to 0$ leads to poor classification accuracy. Fig. 4 gives some insight into why this is true in practice: the support vector machine learns very little for small values of C and assigns almost all training points as support vectors (severe underfitting). As the value of C is increased, the SVM starts to approximate the true decision boundary (see Fig. 1).

B. RBF Kernels

The results from the linear SVM seem to hold with regard to C for the RBF kernel. From fig. 5, it seems that for arbitrarily large C, a line search over γ would yield results close to that of the optimal accuracy one can obtain with an exhaustive grid search.

It is also evident that very large values of γ lead to severe overfitting and a steep corresponding drop in accuracy, even for large C. Small values of C lead to poor classification accuracy irrespective of the value of γ.

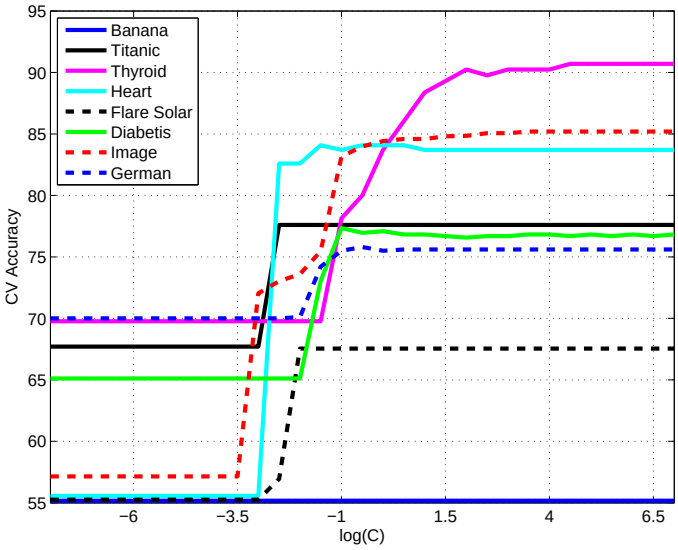


Fig. 3. 10-fold cross validation accuracy for linear SVMs against $log(C)$. All functions seem to converge after some sufficiently high value of C.

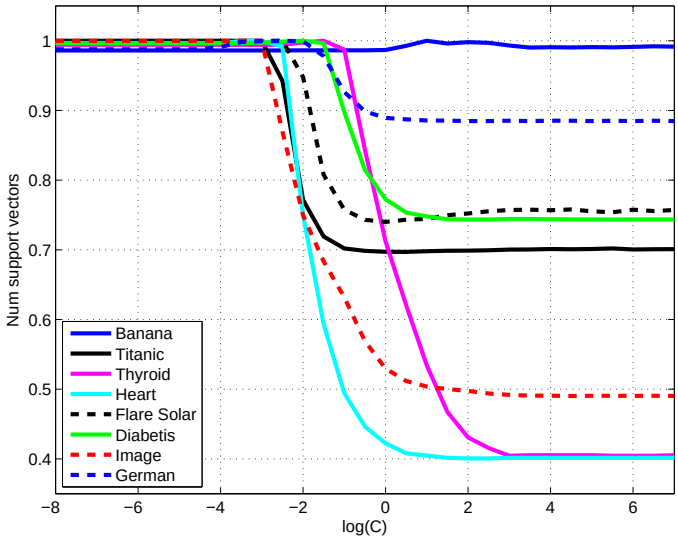
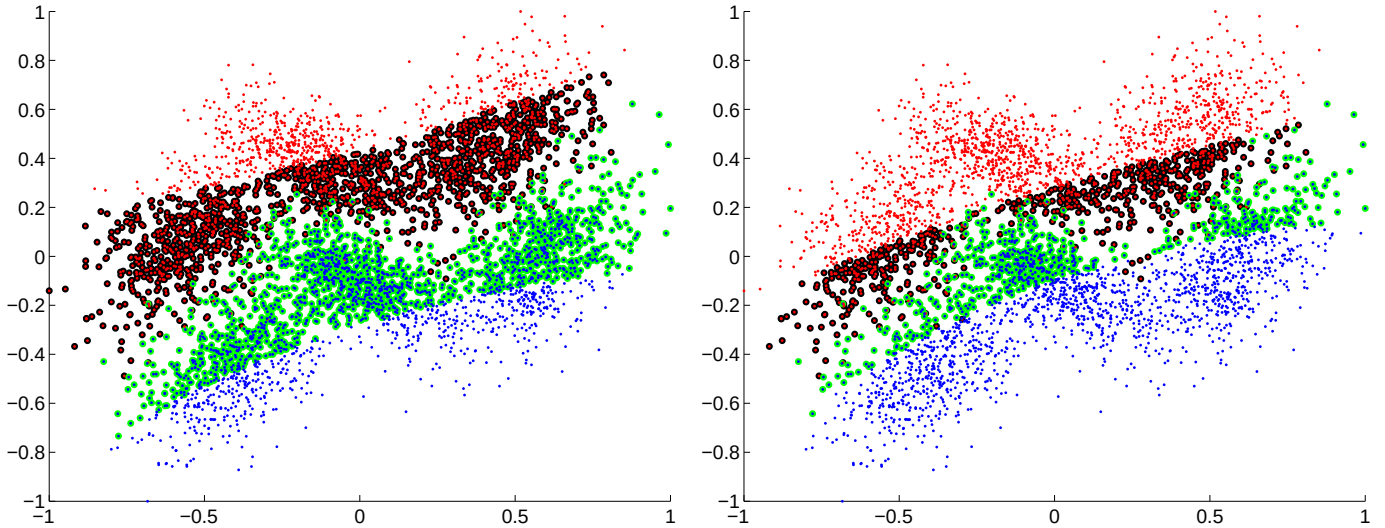


Fig. 4. Normalized number of support vectors vs $log(C)$. It is clear that for small C, the algorithm does not learn much and assigns almost all points as support vectors.

The optimal region within which to search for the hyperparameters values is evidently where C is large and γ is small.

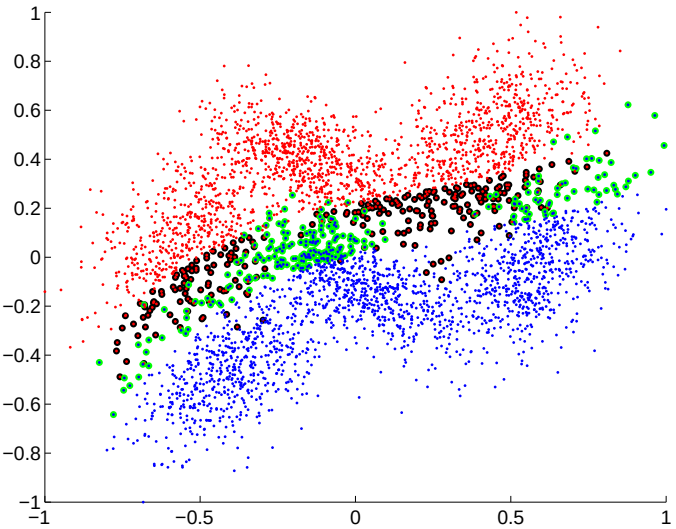
IV. HYPERPARAMETERS VS N

In this section we will discuss the relationship between the SVM hyperparameters and N, the amount of training data. We will show that there is a useful relationship between C and N which can be exploited in cases where there is too much training data to perform a normal grid search in an acceptable amount of time (an extensive grid search on the DFKI problem for example will take approximately 4 months if performed on a single PC).

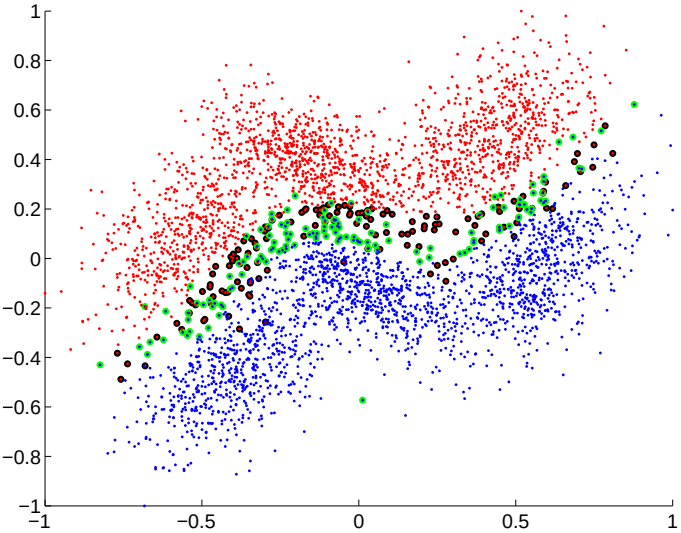


(a) $C = 10^{-2}$, $\gamma = 10^{-0.5}$, Acc=94.4

(b) $C = 10^{-1}$, $\gamma = 10^{-0.5}$, Acc=94.9



(c) $C = 10^{0.5}$, $\gamma = 10^{-0.5}$, Acc=95.55



(d) $C = 10^{4}$, $\gamma = 10^{-0.5}$, Acc=96.8

Fig. 1. Graphical illustration of support vectors (highlighted) on an artificial dataset as C is increased from 10^{-2} to 10^{4}, while γ is kept constant at $10^{-0.5}$. It is clear that as C is increased, the SVM starts to approximate the true boundary between the classes, as the support vectors become more concentrated on that boundary.

A. C vs N

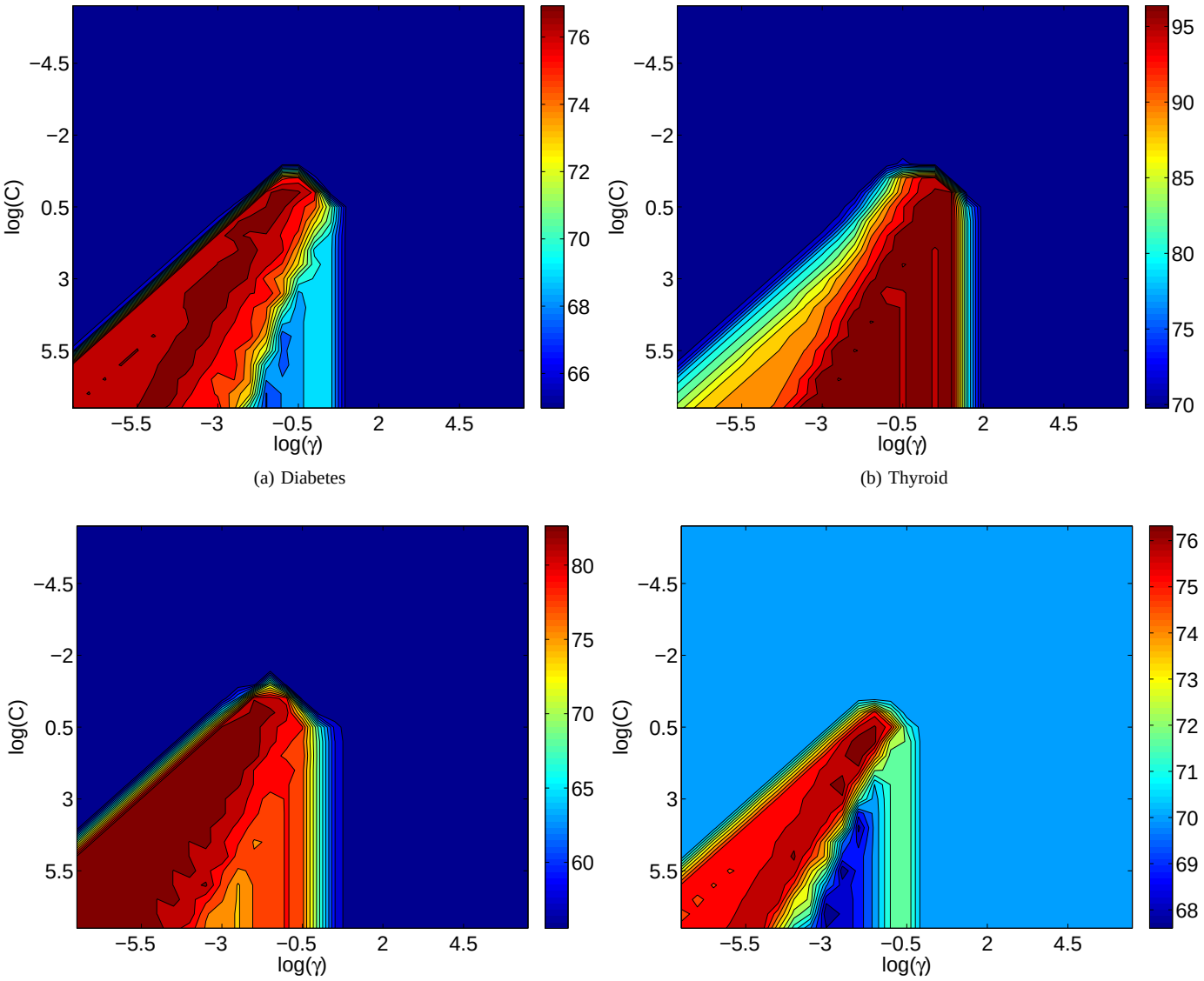
Consider the SVM error function in Eq. (1): as the number of training samples is increased, the width of the optimal separating boundary, and thus the first term in that equation, should remain approximately constant. Since the fraction of marginal or misclassified samples will also depend only weakly on N for large enough N, the summation in the second term will grow linearly with N. Hence, C should be inversely proportional to N to maintain a constant balance between the two terms.

From Fig. 6, this relationship can be seen to hold on a sufficiently large dataset (in this case the image classification dataset from the UCI database). In particular, notice how the lower range for C from within which one can obtain optimal accuracy systematically increases as the amount of data is decreased.

The larger the dataset, the more reliable one can expect this relationship to be. Care should be taken in cases where a high classification accuracy can be obtained though (hence a small percentage of misclassifications), since the probability of selecting a subset with a representative distribution of samples which will be misclassified becomes less likely.

B. γ vs N

The relationship between optimal kernel widths and N is known to be weak in well-studied problems such as kernel density estimation (see section 3.4, [9], where a relationship $\gamma \propto N^{1/5}$ is derived). This can be understood from the weak



(a) Diabetes

(b) Thyroid

(c) Heart

(d) German

Fig. 5. Contour plots depicting the CV accuracy over a wide range of $log(C)$ and $log(\gamma)$ for the UCI diabetes, Thyroid, Heart and German datasets.

dependence of nearest-neighbour distances on N. Typical examples are shown in Fig. 2, where we see that a 64-fold increase in N only increases the median of the nearest-neighbour distance by a factor of 3 (five dimensions) and a factor of 2 (ten dimensions). We therefore expect a similarly weak relationship between the optimal γ and N for SVM training. For the purposes of this paper we consequently assume that a narrow line search around a value obtained on a subset of the data will suffice to obtain the optimal kernel width

C. Algorithm for finding optimal hyperparameters on a subset of the data

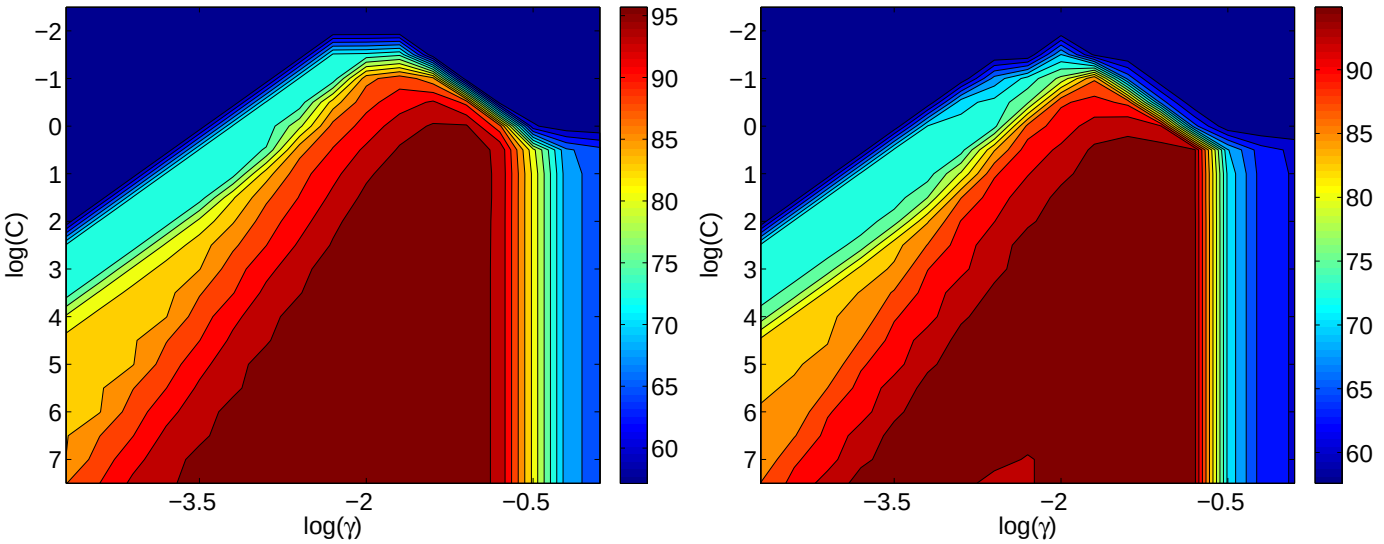
Given the relationships mentioned in sections IV-A and IV-B, we propose the following strategy for finding the optimal hyperparameters on very large datasets:

- Select a subset of the training data N_{sub} and find the optimal hyperparameters using 10-fold cross-validation
- Adapt C_{sub} by scaling as follows: $C^*_{full} = C_{sub} \cdot \frac{N_{sub}}{N_{full}}$
- Using C^*_{full}, do a line-search in a narrow region around γ_{sub} to find γ_{full}

D. Experimental Analysis

All the experiments reported in this paper were conducted on the IDA benchmark repository (available at http://www.fml.tuebingen.mpg.de/Members/raetsch/benchmark) and the results compared with those reported in [7] and [10]. We did not however follow the experimental setup they proposed, since it is not clear that the training and test sets were independent.

We have thus taken the complete dataset (concatenation of the first train and test sets) and divided it into 10 test sets,



(a) 100% of data used

(b) 50% of data used

(c) 25% of data used

(d) 12.5% of data used

Fig. 6. Contour plots showing the results of a grid search with varying amounts of data. Fig. 6a shows a contour plot for all of the data, with every subsequent figure generated with half the amount of data of the previous plot. In this fashion, Fig. 6d is generated with an eighth of the amount of data used for Fig. 6a.

TABLE I
THE 10-FOLD CROSS-VALIDATION ERROR RATES OBTAINED USING SVMS WITH RBF KERNELS. * γ LINE SEARCH WITHOUT C ADAPTATION.

Dataset	**SVM Error (with approximate total CV experiment time in hh:mm in brackets)**					**Total # samples**
	Rätsch	**100%**	**50%**	**25%**	**12.5%**	
Image	2.7 ± 0.6	2.17 ± 0.23 (13:42)	1.95 ± 0.29 (4:20)	1.86 ± 0.38 (1:31)	2.34 ± 0.39 (0:43)	2310
Image*		2.08 ± 0.25	1.95 ± 0.29	1.91 ± 0.29	1.73 ± 0.31	2310
Splice	10.9 ± 0.7	3.40 ± 0.41 (113:35)	3.56 ± 0.50 (37:35)	3.84 ± 0.57 (14:36)	4.00 ± 0.60 (8:29)	3175
Splice*		3.53 ± 0.46	3.62 ± 0.49	3.55 ± 0.45	3.56 ± 0.49	3175
Waveform	9.9 ± 0.4	8.52 ± 0.43 (233:41)	8.64 ± 0.45 (40:2)	8.72 ± 0.46 (7:11)	8.44 ± 0.49 (3:12)	5000
Waveform*		8.42 ± 0.41	8.54 ± 0.38	8.62 ± 0.45	8.52 ± 0.44	5000
DFKI		54.98 (2883:41)	55.88 (430:9)	55.05 (97:45)	54.78 (38:6)	34843
DFKI*		55.19	55.08	54.95	55.00	34843

where for each test set, the rest of the data is considered the training set. For each fold, the training set was then again partitioned into 10 folds, each of which was used to find the optimal parameters with which to evaluate the corresponding

held-out test set. We thus performed 10 independent evaluations, with each evaluation possibly having different SVM hyperparameters. Our 10-fold cross-validation approach also necessarily assigned more training data to each model than was the case in the original partitions from [10] (we thus have a $90-10$ split where they aimed for $60-40$ in general).

The encoding of some of the categorical features in the IDA benchmark repository is also not well-suited to SVMs, in that some categories which are conceptually equidistant are encoded as being ranked. We present our best results *with* a proper encoding (in particular, each feature in the splice dataset was encoded as a 4-bit feature, which leads to significantly better classification accuracy).

The SVM error rates reported in [7] and [10] were averaged over 100 partitions of the dataset and are represented as the mean error observed with the corresponding standard deviation in the first column of table I. All our results are presented as the mean error together with the corresponding standard error.

Also in table I are results obtained when using the algorithm proposed in section IV-C as well as using only a line search, following a grid search. We tested this algorithm using randomly selected subsets of 50%, 25% and 12.5% of the full training set respectively. The results obtained were encouraging in that the SVMs were trained in a fraction of the time and using much less resources than what was the case with the full grid-search. This is especially useful where one has a large dataset (such as DFKI) but with limited time and resources. The results without C adaptation is also interesting in that it indicates that C adaptation sometimes hurts performance. This is especially true if a small optimal value of C, close to the edge of a steep corresponding drop on the contour, is found.

V. Conclusion

We presented theoretical and empirical arguments that gives one more insight as to how to make intelligent choices regarding the region within which to search for optimal hyperparameter values. We also presented a simple algorithm that uses scaling arguments derived from the SVM error function to find the SVM hyperparameters in much less time and requiring much less resources. The scaling arguments with regard to C and N are sensitive to underfitting in cases where subsets are selected from datasets that have little overlap, as can be seen in table I for the case where 12.5% of the data was selected for the image and splice datasets respectively (note the low error rates achievable on both datasets). By performing K-fold cross-validation on each folds training set in order to obtain the optimal hyperparameters, a too low value of K may result in an underestimation of the value of C. Fig. 7 shows how this happened for the case of one of the folds of the splice dataset. Notice that when leave-one-out (LOO) cross-validation was performed, the optimal value of C can be seen to be very large, whereas 10-fold cross-validation led to a complete underestimation of C (small peak before C converges to a slightly lower value).

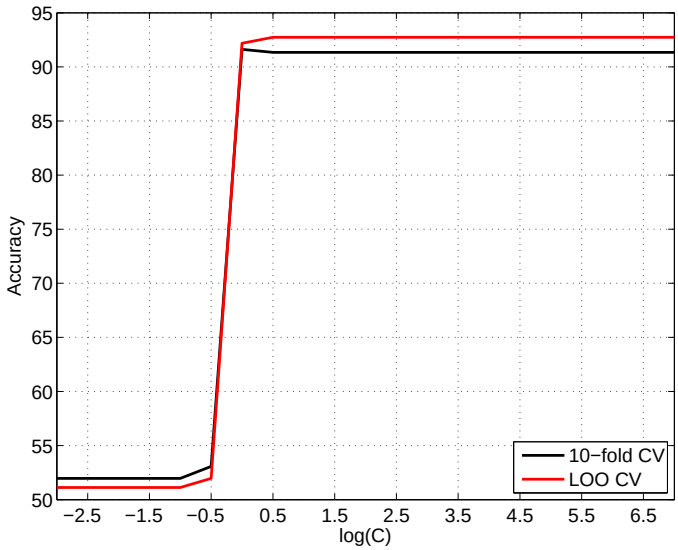


Fig. 7. Cross-section of the contour plot of hyperparameters vs accuracy for both 10-fold cross-validation and leave-one-out cross-validation. This particular cross-section was taken from one of the folds of the 12.5% splice subset and depicts varying C vs classification accuracy with γ fixed at 0.01. Notice how the LOO CV estimate has much less variance than the 10-fold cross-validation estimate.

Our results also indicate that a narrow line search over γ without C adaptation, (given initial parameters from a grid search) is the safest approach to SVM training when one has large amounts of training data. The influence of noise on the selection of optimal C needs to investigated further in order to fully exploit the relationship between C and N.

References

[1] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, September 1995.

[2] C. van Heerden, E. Barnard, M. Davel, C. van der Walt, E. van Dyk, M. Feld, and C. Müller, "Combining regression and classification methods for improving automatic speaker age recognition," in *ICASSP*, Texas, USA, March 2010, pp. 5174–5177.

[3] H. Fröhlich and A. Zell, "Efficient parameter selection for support vector machines in classification and regression via model-based global optimization," in *IJCNN*, vol. 3, Montréal, Québec, Canada, August 2005, pp. 1431–1436.

[4] K. Schittkowski, "Optimal parameter selection in support vector machines," *Journal of Industrial and Management Optimization*, vol. 1, no. 4, pp. 465–476, November 2005.

[5] A. B. Jiménez, J. L. Lázaro, and J. R. Dorronsoro, "Finding optimal model parameters by discrete grid search," ser. Advances in Soft Computing, E. Corchado, M. Juan, and A. Abraham, Eds. Berlin: Springer, December 2007, vol. 44, ch. Innovations in Hybrid Intelligent Systems, pp. 120–127.

[6] K.-P. Wu and S.-D. Wang, "Choosing the kernel parameters for support vector machines by the inter-cluster distance in the feature space," *Pattern Recognition*, vol. 42, no. 5, pp. 710–717, May 2009.

[7] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Machine Learning*, vol. 46, no. 1-3, pp. 131–159, January 2002.

[8] S. S. Keerthi and C.-J. Lin, "Asymptotic behaviours of support vector machines with gaussian kernel," *Neural Computation*, vol. 15, no. 7, pp. 1667–1689, July 2003.

[9] B. Silverman, *Density Estimation for Statistics and Data Analysis*. London, England: Chapman and Hall, 1986.

[10] T. O. G. Rätsch and K.-R. Müller, "Soft margins for adaboost," *Machine Learning*, vol. 42, no. 3, pp. 287–320, March 2001.