Model Selection for Support Vector Machines via Uniform Design

Chien-Ming Huang[a], Yuh-Jye Lee[a,*], Dennis K. J. Lin[b], Su-Yun Huang[c]

[a]*Department of Computer Science & Information Engineering, National Taiwan University of Science and Technology, Taipei 106, Taiwan*

[b]*Department of Supply Chain and Information Systems, Pennsylvania State University, University Park, PA 16802, U.S.A.*

[c]*Institute of Statistical Science, Academia Sinica, Taipei 115, Taiwan*

Abstract

The problem of choosing a good parameter setting for a better generalization performance in a learning task is the so-called model selection. A nested uniform design (UD) methodology is proposed for efficient, robust and automatic model selection for support vector machines (SVMs). The proposed method is applied to select the candidate set of parameter combinations and carry out a k-fold cross-validation to evaluate the generalization performance of each parameter combination. In contrast to conventional exhaustive grid search, this method can be treated as a deterministic analogue of random search. It can dramatically cut down the number of parameter trials and also provide the flexibility to adjust the candidate set size under computational time constraint. The key theoretic advantage of the UD model selection over the grid search is that the UD points are "*far more uniform*" and "*far more space filling*" than lattice grid points. The better uniformity and space-filling phenomena make the UD selection scheme more efficient by avoiding wasteful function evaluations of close-by patterns. The proposed method is evaluated on different learning tasks, different datasets as well as different SVM algorithms.

Key words: discrepancy measure, Gaussian kernel, k-fold cross-validation, model selection, number-theoretic methods, quasi-Monte Carlo, support vector machine, uniform design.

[*] Corresponding author: 43 Keelung Rd., Sec. 4, Taipei 106, Taiwan. Tel: +886-2-27301066; Fax: +886-2-27301081.
Email address: yuh-jye@mail.ntust.edu.tw (Yuh-Jye Lee).

1 Introduction

In recent years, support vector machines (SVMs) with linear or nonlinear kernels (Vapnik, 1995; Cristianini and Shawe-Taylor, 2000) have become one of the most promising learning algorithms for classification as well as regression (Smola and Schölkopf, 2004). However, poor choice of parameter setting can dramatically decrease the generalization performance of SVMs. The problem of choosing a good parameter setting for a better generalization ability is the so called *model selection*. It will be desirable to have an effective and automatic model selection scheme to make SVMs practical for real life applications, in particular, for people who are not familiar with parameters tuning in SVMs. In this article, we develop a nested uniform design methodology for model selection in SVMs, which allows users to find a good parameter combination efficiently and automatically. We focus on selecting the *regularization* parameter and the Gaussian kernel *width* parameter. This problem can be treated as finding the maximum (or minimum) of a function which is only vaguely specified and has many local maxima (or minima). One standard method to deal with the model selection is to use a simple exhaustive grid search over the parameter space. It is obvious that the exhaustive grid search can not effectively perform the task of automatic model selection due to its high computational cost. Therefore, many improved model selection methods have been proposed to reduce the number of trials in parameter combinations (Keerthi and Lin, 2003). Chapelle et al. (2002) use a gradient-based approach to find the minimizing parameter setting for error bounds made by leave-one-out procedure. There are also other gradient-based approaches in the literature (Larsen et al., 1998; Bengio, 2000). Although the gradient-based methods present impressive gain in time complexity, they have a great chance of falling into bad local minima. In contrast to conventional exhaustive grid search, our proposed method can be treated as a deterministic analogue of random search known as quasi-Monte Carlo (Niederreiter, 1992). We first give a heuristic for setting up a two-dimensional search box in the parameter space, which is able to automatically scale the distance factor in the Gaussian kernel. Regardless of the search scheme, it is always important to set up a proper search region. Once the search region is determined, we apply the 2-stage uniform design methodology to select the candidate set of parameter combinations and perform a k-fold cross-validation to evaluate the generalization performance of each parameter combination. The 2-stage uniform design procedure first sets out a crude search for a highly likely candidate region of global optimum and then confines a finer second-stage search therein. We test our method on different learning tasks and different datasets, as well as on several SVM algorithms. Numerical results and comparisons show that our method can effectively find a good parameter combination in a few trials. Our model selection scheme is robust and efficient and can be carried out fully automatically. The nice feature of our model selection scheme is that it provides the flexibility to adjust the candidate size under computational cost constraint. In practice, it can be combined with variants of SVM implementations easily.

The article is organized as follows. Section 2 provides a brief introduction to formulations of SVMs for classification and regression. The model performance measure is discussed in Section 3. Section 4 presents the uniform design methodology. Section 5 describes our nested uniform design scheme for model selection. All numerical results are presented in Section 6. Section 7 concludes the article.

A word about our notation is given below. All vectors will be column vectors unless otherwise specified or transposed to a row vector by a superscript T. For a matrix $A \in R^{m\times d}$, A_i is the ith *row* of A. A column vector of ones of arbitrary dimension will be denoted by $\mathbf{1}$. For $A \in R^{m\times d}$ and $B \in R^{d\times l}$, the kernel $K(A, B)$ maps $R^{m\times d} \times R^{d\times l}$ into $R^{m\times l}$. In particular, if x and y are column vectors in R^d then, $K(x^T, y)$ is a real number, $K(A, x) = K(x^T, A^T)^T$ is a *column* vector in R^m and $K(A, A^T)$ is an $m \times m$ matrix.

2 Support Vector Machines

In this section we give a very brief introduction to nonlinear SVMs for classification and regression. A nonlinear SVM model can be defined by a kernel mixture. One of the most popular kernel functions is the Gaussian kernel (also known as the radial basis function) defined by $K(u^T, v) = e^{-\gamma\|u-v\|_2^2}$, where $u, v \in R^d$, and γ is the *width* parameter of the Gaussian kernel. The value of $K(u^T, v)$ represents the *inner product* of $\Phi(u)$ and $\Phi(v)$ in some high dimensional feature space $\mathcal{F}$, where $\Phi(\cdot) : R^d \mapsto \mathcal{F}$ is a nonlinear map that we do not have to know explicitly in SVM formulations. Besides, values of $K(u^T, v)$ drop off at a rate determined by $\|u - v\|_2^2$ and γ. Note that the parameter γ is the key performance factor for SVMs (Lee and Mangasarian, 2001a, 2001b; Lee, Hsieh and Huang, 2005). It is also well known in statistical kernel smoothing literature that the kernel window size, corresponding to $1/\sqrt{2\gamma}$ here, plays a major influential role for the final appearance of the fitted curve or model (Silverman, 1986). Too large or too small of a γ value will lead to overfitting or underfitting, respectively. We will give a method for determining a search range for γ in Section 5.

2.1 SVMs for Classification

Consider the problem of classifying points into two classes, A_- and A_+. Given a training dataset $\{(x^i, y_i)\}_{i=1}^m$, where $x^i \in \mathcal{X} \subset R^d$ is the vector of the ith input data point and $y_i \in \{-1, 1\}$ is its class label, indicating one of the two classes, A_- and A_+, to which the input point belongs, we represent these data points by an $m \times d$ matrix A, where the ith row A_i corresponds to the ith input data point. We use alternately A_i (a row vector) and x^i (a column vector) for the same ith data point depending on the convenience. The SVM classifier $f(x)$ is of the following form:

$$f(x) = \alpha^T DK(A, x) + b = \sum_{j=1}^{m} y_j \alpha_j K(A_j, x) + b, \tag{1}$$

where $K(z^T, x)$ is a kernel function, and D is an $m \times m$ diagonal matrix with class labels, y_i's, along its diagonal. Conventionally, the coefficients α_j are obtained by solving the following dual maximization problem:

$$\begin{aligned} \max_{\alpha \in R^m} \quad & -\tfrac{1}{2}\alpha^T DK(A, A^T)D\alpha + \mathbf{1}^T\alpha \\ \text{subject to} \quad & \mathbf{1}^T D\alpha = 0, \\ & \mathbf{0} \leq \alpha \leq C\mathbf{1}, \end{aligned} \tag{2}$$

where $[K(A, A^T)]_{ij} = K(A_i, A_j^T)$. The term b in (1) can be determined by the Karush-Kuhn-Tucker (KKT) conditions (Vapnik, 1995; Cristianini and Shawe-Taylor, 2000). Most of the SVM packages, such as LIBSVM (Chang and Lin, 2005) and SVM^{light} (Joachims, 2002), implement SVM by solving the maximization problem above. An alternative smoothing strategy (Lee and Mangasarian, 2001b) has been proposed and solved by a fast Newton-Armijo algorithm that converges globally and quadratically.

2.2 ϵ-Insensitive Support Vector Regression (ϵ-SVR)

In the regression problem, $y_i \in R$ is the response observed at x^i. The aim is to find a linear or nonlinear regression function, $f(x)$, tolerating a small error in fitting the training dataset. This can be achieved by utilizing the ϵ-insensitive loss function that sets an ϵ-insensitive "tube" around residuals. The tiny errors that fall within the tube are discarded. Also, applying the idea of SVMs, the function $f(x)$ is made as flat as possible in fitting the training dataset. Similar to the formulation of SVM classification, we consider a function $f(x)$ of the following form:

$$f(x) = u^T K(A, x) + b = \sum_{j=1}^{m} u_j K(A_j, x) + b. \tag{3}$$

The coefficients u and b can be determined by solving an unconstrained minimization problem given as follows:

$$\min_{(u,b)\in R^{m+1}} \frac{1}{2}u^T u + C\mathbf{1}^T|\xi|_\epsilon, \tag{4}$$

where $|\xi|_\epsilon \in R^m$, and $(|\xi|_\epsilon)_i = \max\{0, |f(x^i) - y_i| - \epsilon\}$, which are the ϵ-insensitive fitting errors. The positive control parameter C here weights the trade-off between the fitting errors and the flatness of $f(x)$. Conventionally, problem (4) is reformulated as a convex quadratic programming problem (Smola and Schölkopf, 2004). A smoothing strategy for the ϵ-SVR is derived and solved, again, by a fast Newton-Armijo algorithm (Lee, Hsieh and Huang, 2005). When dealing with large data problems, classification or regression, a reduced kernel technique for support vector machine (RSVM) can be applied to cut down the computational cost as well as the model complexity (Lee and Mangasarian, 2001a; Lee and Huang, 2007). In our numerical study, the reduced kernel approximation will be applied to some experiments.

3 Performance Measure for SVM Model Selection

The most common performance assessment method is probably the k-fold cross-validation (Stone, 1974) and the leave-one-out procedure. Both require that the learning engine be trained multiple times in order to obtain a performance measure for each parameter combination. In a k-fold cross-validation, the training data is randomly split into k mutually exclusive subsets (the folds) of approximately equal sizes. The resulting SVM model (i.e., the decision rule or the regression function) is obtained by training on $k-1$ subsets and then the model is tested on the remaining one subset. This procedure is repeated k times and in this fashion each subset is used for testing only once. By averaging the test errors over the k trials it gives an estimate of the expected generalization error. The leave-one-out procedure can be viewed as an extreme form of the k-fold cross-validation with k equal to the number of examples. Although the leave-one-out is known as an unbiased estimation method, it is computationally much more expensive than a k-fold cross-validation.

One standard method to deal with the model selection problem is to use a simple grid search on the parameter domain of interest. In this article, we consider the parameter space consisting of the regularization parameter C and the Gaussian kernel width parameter γ. For ϵ-insensitive support vector regression, we leave the parameter ϵ as user pre-specified. That is, our search region is a two-dimensional box. It is obvious that the exhaustive grid search can not do automatic model selection effectively due to its high computational cost. For example, for a grid search with 20×20 mesh parameter combinations (400 trials) in a 5-fold cross-validation, it will take 2000 times of SVM trainings to select the best parameter combination. Therefore, many improved model selection methods have been proposed to reduce the number of trials in parameter combinations. Chapelle et al. (2002) use a gradient-based approach to find the minimal error bound made by leave-one-out procedure. To show the model performance of some various parameter combinations, Figure 1 plots the 5-fold average test set accuracy for three public available datasets, `banana`, `waveform` and `splice` in a three dimensional surface, where the x-axis and the y-axis are $\log_2 C$ and $\log_2 \gamma$, respectively. The z-axis is the 5-fold average test accuracy. Each mesh point in the (x, y)-plane stands for a parameter combination and the z-axis indicates the model performance measure. It is easy to see that there are many local maxima. Thus, the gradient-based methods have a great chance of being trapped into (bad) local maxima. Also these plots show that these surfaces have low-degree of regularity, which further hinders the use of gradient-based methods.

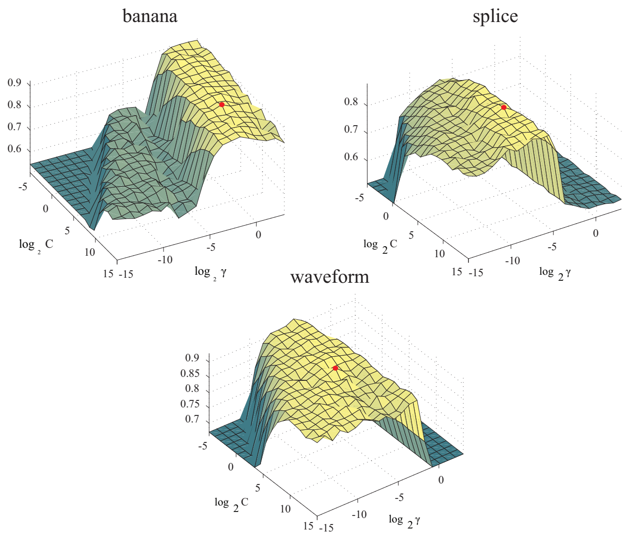


Fig. 1. *Three examples of search space of model selection*

Below we introduce the use of nested uniform designs (UDs) for model selection in SVMs. Basically the UD finds good representative points uniformly scattered over the parameter domain to replace the lattice grid points for a much more efficient parameter search.

4 Uniform Design (UD)

The uniform experimental design is one kind of space filling designs that can be used for computer and industrial experiments. The UD seeks its design points to be uniformly scattered on the experimental domain. Suppose there are s parameters of interest over a domain C^s. The goal here is to choose a set of m points $P_m = \{\theta_1, \ldots, \theta_m\} \subset C^s$ such that these points are uniformly scattered on C^s. Let $F(\theta)$ be the cumulative uniform distribution function over C^s and $F_m(\theta)$ be the empirical cumulative distribution function of P_m. Let the L_2-discrepancy of nonuniformity of P_m be defined as

$$D_2(C^s, P_m) = \left[\int_{C^s} |F_m(\theta) - F(\theta)|^2 \, d\theta \right]^{1/2} . \tag{5}$$

Here we consider only the case that C^s is a cube and for convenience a unit cube, i.e., $C^s = [0, 1]^s$. The search for UDs with minimum L_2-discrepancy is an NP-hard problem. Approximation is used to find low-discrepancy design close to the minimum discrepancy UD. A modification to the L_2-discrepancy is the centered L_2-discrepancy, which considers the uniformity not only of P_m over C^s, but also of all the projection uniformity of P_m over C^u, where C^u is a $|u|$-dimensional unit cube involving only $|u|$-many coordinates. The UD tables in the UD-web are all constructed under the centered L_2-discrepancy. See Fang and Lin (2003). These are the UDs that we will employ in this article.

For implementing the uniform design in SVM model selection problem, the following steps are necessary:

(1) Choose a parameter search domain, determine a suitable number of levels for each parameter (or factor in design terminology).
(2) Choose a suitable UD table to accommodate the number of parameters and levels. This can be easily done by visiting the UD-web.
`http://www.math.hkbu.edu.hk/UniformDesign`
(3) From the UD table, randomly determine the run order of experiments and conduct the performance evaluation of each parameter combination in the UD.
(4) Fit the SVM model.
(5) The last step is a knowledge discovery step from the built model. That is, to find the best combination of the parameter values that maximizes the performance measure.

For a detailed discussion, literature review and recent development on uniform design, see Fang and Lin (2003).

5 Nested Uniform Designs for Model Selection in SVMs

We describe our search strategy based on nested UDs (Niederreiter and Peart, 1986; Fang et al., 2000; Fang and Lin, 2003) to reduce the number of trials in parameter combinations. Regardless of the search scheme, it is always important to set up a proper search region. Many numerical experiments and past experience have indicated that the width parameter γ is the key factor in SVMs model selection . Hence, the appropriate γ range must be made prior to parameter search. We note that the function value of the Gaussian kernel not only depends on γ but also on the distance between two data points. The magnitude of the distance between a pair of data points also depends on the input space dimension. Here, we propose a heuristic for determining the search range of γ, which is able to automatically scale the distance factor in the Gaussian kernel. Let $A_{\bar{i}}^*$ and $A_{\bar{j}}^*$ be a pair of the *closest distinct points* in the training dataset and let $\rho = \|A_{\bar{i}}^* - A_{\bar{j}}^*\|_2^2$, i.e., $\rho = \min_{A_{\bar{i}} \neq A_{\bar{j}}} \|A_{\bar{i}} - A_{\bar{j}}\|_2^2$. We confine the kernel function value of this pair of points to the range $[0.150, 0.999]$. That is,

$$0.150 \leq e^{-\gamma\|A_{\bar{i}}^* - A_{\bar{j}}^*\|_2^2} = e^{-\gamma\rho} \leq 0.999, \tag{6}$$

which can be converted to $L_\gamma \leq \gamma \leq U_\gamma$ with

$$L_\gamma = -\frac{\ln(0.999)}{\rho} \approx \frac{10^{-3}}{\rho} \quad \text{and} \quad U_\gamma = -\frac{\ln(0.150)}{\rho} \approx \frac{1.90}{\rho}. \tag{7}$$

The lower bound for γ, leading to a lower bound for underfitting, comes from the inequality $e^{-\gamma\rho} \leq 0.999$, which basically says that the window parameter γ should be able to sustain the closest neighboring pair to have at most 0.999 similarity measure but not higher. Pairs of data with similarity measure greater than 0.999 are considered "being lumped together" by the kernel smoothing with lower bound width parameter L_γ. The upper bound for γ can be transformed into a lower bound for σ via $\sigma = \sqrt{1/(2\gamma)}$. Then,

$$\frac{\min_{A_{\bar{i}} \neq A_{\bar{j}}} \|A_{\bar{i}} - A_{\bar{j}}\|_2}{\sqrt{2 \times 1.90}} \leq \sigma.$$

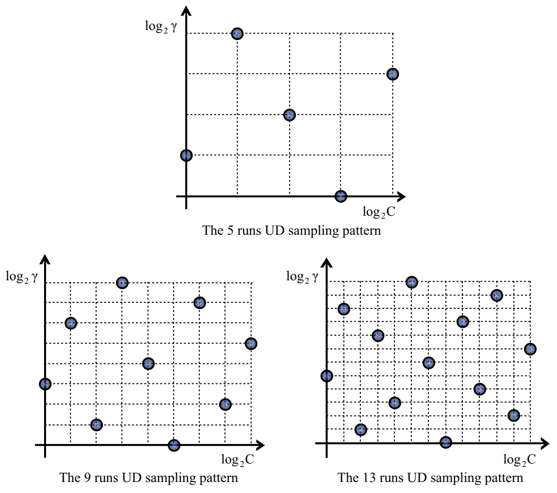
The denominator is roughly about 2. That is, in terms of σ, the kernel window size is at least about half of the minimum nearest neighbor distance. The k-nearest neighbor kernel smoothing with $k = 1/2$ is highly overfitting (Silverman, 1986). In brief, the lower bound value L_γ will lead to inadequately underfitting, while the upper bound value U_γ will lead to highly overfitting. Thus, the search range $[L_\gamma, U_\gamma]$ has covered the extent from underfitting to overfitting.

In practice, finding the *closest distinct points* in a massive training dataset is very time consuming. We suggest the following scheme for the upper and lower bound estimates based on a random subset. First, randomly sample a small subset from the entire dataset, next calculate the upper and lower bounds using this random subset, and finally adjust the bounds by a multiplicative factor $(m/\bar{m})^{2/(4+d)}$, where $\bar{m}$ is the subset size and d is the dimension of x. For assessing the search range of γ using a reduced set, it should be adjusted accordingly to account for the effect caused by using only a fraction $\bar{m}/m$ of data. It is well known in the nonparametric literature that an ideal window width σ is of order $\sigma = O(m^{-1/(4+d)})$, or equivalently $\gamma = O(m^{2/(4+d)})$ (Stone, 1984; Silverman, 1986). Thus, if only a fraction $\bar{m}/m$ of data is used, a multiplicative factor $(m/\bar{m})^{2/(4+d)}$ adjustment should be adopted. This adjustment factor is often close to one in the reduced set case with a moderate to large sized d.

The SVMs regularization parameter C is another challenge in SVM model selection. The parameter C determines the trade-off between minimizing the training error and reducing the model complexity. The range of C depends on the underlying SVM learning algorithm being used. Our empirical observation suggests that the most appropriate C range for SVMs is between 10^{-2} and 10^{4} except for the case of RSVM (Lee and Mangasarian, 2001a; Lee, Hsieh and Huang, 2005). The justification is that the reduced kernel technique has dramatically reduced the model complexity. Hence, it usually requires a larger C to obtain a good resulting model. The appropriate C range for RSVM is between 10^{0} and 10^{6}. The conventional ϵ-SVR implemented in LIBSVM will take a long CPU time to obtain the resulting model when C is large. For this reason, we use the range of C between 10^{-2} and 10^{2} in the ϵ-SVR model selection experiment for LIBSVM. Once we have the two-dimensional parameter search box we are ready for describing our nested UDs scheme.

One reminder for the reader is that the range for the search box is based on several subjective choices of settings. It remains somehow arbitrary, but it is probably the best one can do here. One simple remedy is, one can make the search range conservative to include some extent of coverage for the first stage UD search and then the "nesting mechanism" shrinks the range for the second stage UD search. In other words, the method of nested designs first sets out a crude search for a wide range of candidate region and then confines a finer second-stage search therein.

Figure 2 shows the UD sampling patterns, where N runs means that we distribute N trial points of parameter combinations uniformly on the predetermined search domain. Note that each $\log_2 C$ or $\log_2 \gamma$ value is used at most once in the nested UD based method, and there is no point placed on the corners. These characteristics are very important for efficient model selection. The corner points on the search domain often cause the overfitting or underfitting phenomena, and should be avoided. The parameter points from the UD sampling patterns are wisely chosen by the number-theoretic methods (Fang and Wang, 1994) to make them "uniform" and "space-filling". This UD methodology is a deterministic analogue of random search known as quasi-Monte Carlo. It is



$\log_2 \gamma$
$\log_2 C$
The 5 runs UD sampling pattern
$\log_2 \gamma$
$\log_2 C$
The 9 runs UD sampling pattern
$\log_2 \gamma$
$\log_2 C$
The 13 runs UD sampling pattern

Fig. 2. *UD sampling patterns*

known that a quasi-Monte Carlo method with judiciously chosen deterministic points usually leads to a faster rate of convergence than a corresponding Monte Carlo method and lattice grid method (Niederreiter, 1992).

We perform the UD based method in two stages. At the first stage, we use a 13-runs UD sampling pattern (see Figure 3) in the appropriate search range proposed above. At the second stage, we halve the search range for each parameter coordinate in the log-scale and let the best point from the first stage be the center point of the new search box. We do allow the second stage UD points to fall outside the prescribed search box. Then we use a 9-runs UD sampling pattern in the new range. The total number of parameter combinations is 21 (the duplicate point, i.e., the center point at the second stage, is trained and counted only once). Moreover, to deal with large sized datasets, we combine a 9-runs and a 5-runs sampling pattern at these two stages, as pictured in Figure 4. The total number of parameter combinations is reduced to 13 (again, the duplicate point, i.e., the center point at the second stage, is trained and counted only once), and the UD based method can still make the resulting SVM model perform well. In next section, the numerical results will show merits of the nested UD model selection method.

The method of nested UDs is not limited to 2 stages and can be applied in a sequential manner and one may consider a finer net of UDs to start with. The reason that we use a crude 13-runs or a 9-runs design at the first stage is that it is simply enough for the purpose of model selection in the real data SVM problems in our experiments. See Section 6 for more numerical details. The nested uniform designs for model selection has some intrinsic connection with the popular and widely used quasi-Monte Carlo methods (Niederreiter, 1992; Fang and Wang, 1994). This UD quasi-random search can be considered as a localization-of-search type algorithm (Niederreiter and Peart, 1986) to speed up the decision of model selection in SVM.

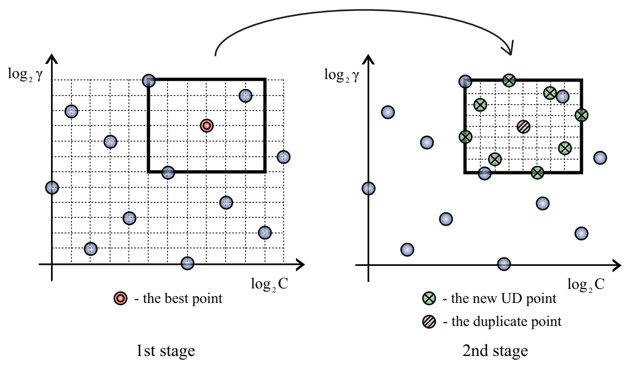


Fig. 3. *The nested UD model selection with a 13-points UD at the first stage and a 9-points UD at the second stage*

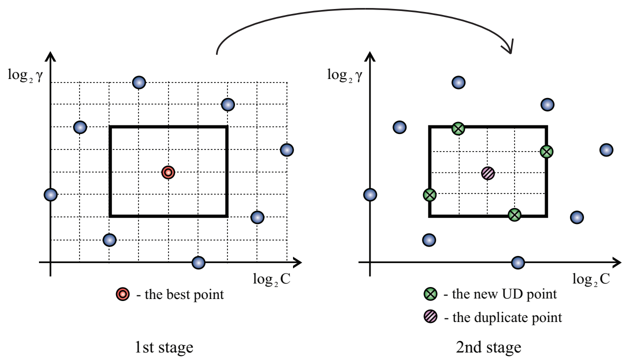


Fig. 4. *The nested UD model selection with a 9-points UD at the first stage and a 5-points UD at the second stage*

6 Numerical Results

In this section, we will apply our model selection method not only to different datasets but also to different SVM packages, including LIBSVM (Chang and Lin, 2005), SSVM (Lee and Mangasarian, 2001b), ϵ-SSVR (Lee, Hsieh and Huang, 2005) and RSVM (Lee and Mangasarian, 2001a; Lee and Huang, 2007). LIBSVM is probably one of the most popular libraries for conventional support vector classification and regression. SSVM and ϵ-SSVR are alternative SVM implementations, which utilize a smoothing technique and are solved by a fast Newton-Armijo algorithm. RSVM utilizes the reduced kernel approximation to reduce the computational complexity. We implement our model selection method in MATLAB. All experiments are run on a personal computer consisting of a 3.0 GHz Pentium-4 processor and a one-gigabytes memory.

In binary classification experiments, we follow the same procedures and datasets in Keerthi and Lin (2003), Rätsch (1999), and Newman et al. (1998). All datasets have their own partitions into training set and test set, and they are directly used as given in the above mentioned references without any further normalization or scaling. Table 1 shows the summary of binary classification datasets. The terms TrnA and TrnB stand for the numbers of positive and negative training examples, respectively. Similarly, the terms TstA and TstB stand for the numbers of the positive and negative test (validation) examples. The number of input predictor variables is denoted by the term *no. variables*.

Problem	Size				
	TrnA	TrnB	TstA	TstB	no. variables
banana	217	183	2159	2741	2
image	740	560	580	430	18
splice	483	517	1044	1131	60
waveform	132	268	1515	3085	21
tree	198	502	3050	8642	18
adult	395	1210	7175	22414	123
web	72	2405	1094	37900	300

Table 1
Summary of classification datasets.

For the model selection procedure in classification problems, a 5-fold cross-validation is used to obtain estimates of generalization error in the training set (TrnA+TrnB). For each dataset, the parameter pair (C, γ) with the smallest 5-fold cross-validation error rate among all UD pairs are chosen as the final parameter estimates and are used in SVM training with all the training set (TrnA+TrnB). The resulting model is used to predict the test set (TstA+TstB), and the test set error rates are shown in Table 2. Note that the test set does not enter the training and model selection procedures.

In Table 2, the term UD1 stands for the strategy of using 13 and 9 trial pairs of UDs at the first stage and the second stage, respectively. The term UD2 denotes the strategy of using 9 and 5 trial pairs of UDs at the respective two stages. The numbers of trial pairs of the Grid method and Keerthi and Lin's method are 441 and 54. However, we only use 21 or 13 total trial pairs in the UD1 or UD2 method, respectively. Keerthi and Lin's search scheme is confined to a good parameter region and then do twice one-dimensional search therein, while ours strikes for most economic scattering points through mathematically rigorous number-theoretic approach. The Grid and Keerthi & Lin columns are the test set error from Keerthi and Lin (2003). The UD1 and UD2 columns are the average test set error $\pm$ standard deviation from 10 times repeated experiments of our model selection method. We obtain comparable test set errors for LIBSVM and SSVM. However, our proposed methods have used a lot fewer trial pairs. The fewer trial pairs take less computing time in model selection procedures. We further apply our model selection method to RSVM. The RSVM randomly selects 5% training set to form the reduced kernel and

Problem	LIBSVM			
	Grid (441)	Keerthi & Lin (54)	UD1 (21)	UD2 (13)
banana	0.1235	0.1178	0.1128±0.0038	0.1121±0.0054
image	0.0248	0.0248	0.0244±0.0013	0.0246±0.0025
splice	0.0970	0.1011	0.1044±0.0039	0.1017±0.0065
waveform	0.1078	0.1078	0.1071±0.0038	0.1120±0.0044
tree	0.1132	0.1246	0.1157±0.0047	0.1168±0.0041
adult	0.1614	0.1614	0.1618±0.0032	0.1602±0.0013
web	0.0222	0.0222	0.0210±0.0015	0.0212±0.0004
Problem	SSVM		RSVM	
	UD1 (21)	UD2 (13)	UD1 (21)	UD2 (13)
banana	0.1219±0.0070	0.1185±0.0070	0.1229±0.0077	0.1239±0.0053
image	0.0307±0.0040	0.0279±0.0061	0.0437±0.0082	0.0429±0.0081
splice	0.1005±0.0019	0.1003±0.0030	0.1346±0.0041	0.1360±0.0053
waveform	0.1055±0.0035	0.1087±0.0053	0.1138±0.0040	0.1121±0.0039
tree	0.1171±0.0026	0.1189±0.0029	0.1193±0.0054	0.1178±0.0040
adult	0.1605±0.0020	0.1611±0.0021	0.1614±0.0019	0.1625±0.0016
web	0.0236±0.0014	0.0229±0.0020	0.0248±0.0014	0.0258±0.0020

Table 2
The average test set error rate of classification problems. The Grid and Keerthi & Lin columns are the test set error from Keerthi and Lin (2003). The UD1 and UD2 columns are the average test set error ± standard deviation by 10 *times repeated experiments of our model selection method. RSVM randomly selects* 5% *training set to form the reduced kernel. The number in parentheses denotes the number of trial pairs.*

also obtains comparable test set errors. Moreover, RSVM saves even more computing time in every single training run.

In regression experiments, we follow the same procedures and datasets, Housing, Computer-Activity and Kin-fh, as in the article by Lee, Hsieh and Huang (2005). We also randomly select 1000 rows from Comp-Activ and Kin-fh, respectively, to form two smaller datasets: Comp-Active_1000 and Kin-fh_1000. In these experiments, the 2-norm relative error is used to evaluate the discrepancy between the predicted values and the observations. For an observation vector y and its prediction $\hat{y}$, the 2-norm relative error is defined as follows:

$$\frac{\|y-\hat{y}\|_2}{\|y\|_2}. \tag{8}$$

In order to evaluate how well each method generalizes to unseen data, we split the entire dataset into two parts, the training set and the test set. The training set is used to estimate the regression function; the test set, which is not involved in the training procedure nor the model selection, is used to evaluate the prediction ability of the resulting regression function. We have also used a *stratification* scheme in splitting the entire dataset to retain the "similarity" pattern between training and test datasets. That is, we try to make the training set and the test set to have similar observational distributions. A smaller test error indicates a better prediction ability. We perform a 10-fold cross-validation on each dataset and report the average test error in Table 4.

Problem	Size		
	Training	Test	no. variables
housing	456	50	13
Comp-Activ_1000	900	100	21
Kin-fh_1000	900	100	32
Comp-Activ	7373	819	21
Kin-fh	7373	819	32

Table 3
Summary of regression datasets.

In Lee, Hsieh and Huang (2005), the reduced kernel approach has been used for solving the two large datasets: Comp-Activ and Kin-fh. The term "N/A" in Table 4 indicates that there are no full kernel numerical results reported in Lee, Hsieh and Huang (2005). Their experiments have used a manual grid method with many trial pairs for parameters. However, the UD-based automatic model selection method can obtain comparable 2-norm relative errors for LIBSVM and ϵ-SSVR in much smaller numbers of trial pairs. The number of trial pairs is 21 in the UD1 method and 13 in the UD2. Moreover, the UD-based model selection works well successful for ϵ-SSVR with the reduced kernel.

In summary, our numerical experiments show that our method can find a good parameter combination in a small number of trial pairs. It can be applied to different SVM packages such as SSVM, ϵ-SSVR, RSVM and LIBSVM. The small overall standard deviations shown in Table 2 and Table 4 indicate the robustness of our method. We also report in Table 5 the average CPU time for a single training run.

Problem	LIBSVM		
	Lee	UD1 (21)	UD2 (13)
housing	0.1168	0.1206±0.0041	0.1196±0.0042
Comp-Activ_1000	0.0307	0.0317±0.0002	0.0322±0.0002
Kin-fh_1000	0.1403	0.1363±0.0004	0.1365±0.0004
Comp-Activ	0.0280	0.0284±0.0001	0.0287±0.0001
Kin-fh	0.1322	0.1319±0.0001	0.1319±0.0001
Problem	ϵ-SSVR		
	Lee	UD1 (21)	UD2 (13)
housing	0.1171	0.1158±0.0026	0.1187±0.0033
Comp-Activ_1000	0.0300	0.0317±0.0002	0.0317±0.0003
Kin-fh_1000	0.1385	0.1356±0.0004	0.1358±0.0003
Problem	RSVM		
	Lee	UD1 (21)	UD2 (13)
housing	N/A	0.1619±0.0042	0.1632±0.0047
Comp-Activ_1000	N/A	0.0339±0.0007	0.0340±0.0009
Kin-fh_1000	N/A	0.1357±0.0003	0.1360±0.0004
Comp-Activ	0.0299	0.0287±0.0001	0.0289±0.0002
Kin-fh	0.1319	0.1321±0.0003	0.1321±0.0003

Table 4
The average 2-norm relative error (8) of regression problems. The Lee column is the 2-norm relative error from Lee, Hsieh and Huang (2005). The UD1 and UD2 columns are the average error ± standard deviation by 10 times repeated experiments of our model selection method. RSVM randomly selects 5% training set to form the reduced kernel. The number in parentheses denotes the number of trial pairs.

7 Conclusion and Future Work

We have developed a nested uniform design methodology for model selection in SVMs, which can find a good parameter combination in an efficient and fully automatic way. We believe that this is a novel application of uniform design in computer experiments. In practice, our proposed method can be combined with any SVM packages easily. Further research will be applying the nested uniform design methodology to model selection problems involving more parameters, e.g., parameters in polynomial kernels or the ϵ-insensitivity in ϵ-SVR. One advantage of using UDs, or nested UDs, over the grid search, is that, as the dimension increases, the grid search is getting less feasible.

Problem	SSVM/ϵ-SSVR		RSVM	
	UD1(105)	UD2(65)	UD1(105)	UD2(65)
banana	0.348±0.013	0.423±0.020	0.005±0.000	0.008±0.001
image	15.180±0.213	45.670±17.575	0.068±0.001	0.069±0.002
splice	3.157±0.091	2.998±0.216	0.113±0.002	0.119±0.008
waveform	0.217±0.015	0.220±0.017	0.007±0.001	0.010±0.001
tree	1.579±0.100	1.670±0.042	0.016±0.001	0.020±0.001
adult	15.149±0.316	14.944±0.073	0.264±0.006	0.263±0.008
web	75.023±2.937	72.098±0.779	1.470±0.032	1.498±0.050
housing	0.545±0.015	0.562±0.009	0.009±0.000	0.009±0.000
Comp-Activ_1000	2.779±0.021	2.851±0.029	0.036±0.001	0.037±0.001
Kin-fh_1000	2.610±0.030	2.711±0.026	0.040±0.001	0.041±0.001
Comp-Activ	N/A	N/A	8.609±0.096	8.342±0.129
Kin-fh	N/A	N/A	6.756±0.167	6.946±0.373

Table 5
The average CPU time in second ± standard deviation by 10 times repeated experiments. The number in parentheses denotes the number of SVM or SVR trainings in a single experiment for model selection with a 5-fold cross-validation. RSVM randomly selects 5% training set to form the reduced kernel.

However, the impact of dimensionality to UD is rather minor. UDs in high dimensional boxes can be looked up in the UD-web. The UD-web has tabulated the UDs for dimensionality as high as 20-30. However, one should be cautious for using high dimensional UDs. As the dimensionality increases, scattering points becomes sparse, and it is true for UD points, too. Due to the efficient UD model selection, we are able to build a web-based data analysis system that will allow users to upload their datasets via web browsers, and the system will return the resulting SVM trained models with testing results to the users in a fully automatic way. The web-based data analysis system is available at `http://dmlab1.csie.ntust.edu.tw/WDAS`. The automatic UD-based model selection is implemented in MATLAB, named "`hibiscus`" in the SSVM toolbox available at: `http://dmlab1.csie.ntust.edu.tw/downloads/`.

References

[1] Y. Bengio. Gradient-based optimization of hyper-parameters. *Neural Computation*, 12(8):1889–1900, 2000.

[2] C.-C. Chang and C.-J. Lin. *LIBSVM: a Library for Support Vector Machines*, April 2005.

[3] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3):131–159, 2002.

[4] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, Cambridge, MA, USA, 2000.

[5] K.-T. Fang and D. K. J. Lin. Uniform experimental designs and their applications in industry. In R. Khattree and C.R. Rao, editors, *Handbook of Statistics*, 22, 131–170, North-Holland, Amsterdam, 2003.

[6] K.-T. Fang, D. K. J. Lin, P. Winker, and Y. Zhang. Uniform design: theory and applications. *Technometrics*, 42:237–248, 2000.

[7] K.-T. Fang and Y. Wang. *Number-theoretic Methods in Statistics*. Chapmman & Hall, London, 1994.

[8] T. Joachims. SVM^{light}, 2002. http://svmlight.joachims.org.

[9] S. S. Keerthi and C.-J. Lin. Asymptotic behaviors of support vector machines with Gaussian kernel. *Neural Computation*, 15(7):1667–1689, 2003.

[10] J. Larsen, C. Svarer, L. N. Andersen and L. K. Hansen. Adaptive regularization in neural network modeling. In G. B. Orr and K. R. Müller, eds. *Neural Networks: Trick of the Trade*. Berlin, Springer, 1998.

[11] Y.-J. Lee, W.-F. Hsieh, and C.-M. Huang. ϵ-SSVR: a smooth support vector machine for ϵ-insensitive regression. *IEEE Transactions on Knowledge and Data Engineering*, 17:678–685, 2005.

[12] Y.-J. Lee and S.-Y. Huang. Reduced support vector machines: a statistical theory. *IEEE Transactions on Neural Networks*, 18:1–13, 2007.

[13] Y.-J. Lee and O. L. Mangasarian. RSVM: reduced support vector machines. In *First SIAM International Conference on Data Mining*, Chicago, 2001a.

[14] Y.-J. Lee and O. L. Mangasarian. SSVM: a smooth support vector machine. *Computational Optimization and Applications*, 20:5–22, 2001b.

[15] D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz. UCI Repository of machine learning databases, 1998.
http://www.ics.uci.edu/∼mlearn/MLRepository.html. Irvine, CA: University of California, Department of Information and Computer Science.

[16] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 1992.

[17] H. Niederreiter and P. Peart. Localization of search in quasi-Monte Carlo methods for global optimization. *SIAM Journal on Scientific and Statistical Computing*, 7(2):660–664, 1986.

[18] G. Rätsch. Benchmark repository of Intelligent Data Analysis Group (IDA). http://ida.first.fraunhofer.DE/projects/bench/benchmarks.htm.

[19] B. W. Silverman. *Density Estimation for Statistics and Data Analysis.* Chapmman & Hall, London, 1986.

[20] A. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14:199–222, 2004.

[21] C. J. Stone. An asymptotically optimal window selection rule for kernel density estimates. *Annals of Statistics*, 12:1285–1297, 1984.

[22] M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society, B*, 36(1):111–147, 1974.

[23] V. N. Vapnik. *The Nature of Statistical Learning Theory.* Springer, New York, USA, 1995.