Learning to Predict the Leave–one–out Error of Kernel Based Classifiers

Koji Tsuda[1,3], Gunnar Rätsch[1,2], Sebastian Mika[1], Klaus-Robert Müller[1,2]

[1] GMD FIRST, Kekuléstr. 7, 12489 Berlin, Germany
[2] University of Potsdam, Am Neuen Palais 10, 14469 Potsdam
[3] AIST Computational Biology Research Center, 2-41-6, Aomi, Koutou-ku, Tokyo, 135-0064 , Japan
{tsuda, raetsch, mika, klaus}@first.gmd.de

Abstract. We propose an algorithm to predict the leave-one-out (LOO) error for kernel based classifiers. To achieve this goal with computational efficiency, we cast the LOO error approximation task into a *classification* problem. This means that we need to *learn* a classification of whether or not a given training sample – if left out of the data set – would be misclassified. For this learning task, simple data dependent features are proposed, inspired by geometrical intuition. Our approach allows to reliably select a good model as demonstrated in simulations on Support Vector and Linear Programming Machines. Comparisons to existing learning theoretical bounds, e.g. the span bound, are given for various model selection scenarios.

1 Introduction

Numerous methods have been proposed [7, 8, 3, 5, 9] for model selection of kernel-based classifiers such as Support Vector Machines (SVMs) [7] and Linear Programming Machines (LPMs) [1]. They all try to find a reasonably good estimate of the generalization error to select the proper hyperparameters. The *data dependent* LOO error would in principle be ideal for selecting hyperparameters of learning machines, as it is an (almost) unbiased estimator of the true generalization error [6]. Its computation is, unfortunately, for most practical cases prohibitively slow.

There have been several attempts to approximate the leave-one-out error in closed-form for SVM classifiers [8, 3, 5]. For example, a new type of bound was proposed that relies on the *span* of the SVs and was empirically found to perform best among the learning theoretical bounds [8]. However, such approximations are limited to a special learning machine, i.e. SVM, and it seems difficult to provide a useful approximation that is valid for a more general class of classifiers.

In this work, we introduce a learning approach for approximating the LOO error of general kernel classifers such as SVMs or Linear Programming Machines (LPM). We propose to use geometrical features to cast the leave-one-out error approximation problem for kernel classifiers into a – fast solvable – *classification* problem. Thus, our LOO error approximation problem reduces to *learn a classification of whether or not a training sample, if left out the data set, will be* *misclassified.* This task is referred to as a *meta* learning problem because we try to learn about a learning machine. The meta classification task is data rich as a large number of training patterns can be generated from all sorts of different classification problems. Note that we are using features that are meant to reflect the local difficulty or complexity of the meta learning problem across a large set of possible data. In experiments, we show that our approach works well both for SVM and LPM.

After reviewing some popular learning theoretical LOO error bounds we describe the features used as inputs for the meta learning problem and solve it by using a classification approach. Subsequently we perform simulations showing the usefulness of our LOO error approximation scheme in comparison to other bounds and finally conclude with some remarks.

2 Reviewing SVMs, LPMs and Selected LOO Bounds

When learning with SVMs [7] and LPMs [1], one is seeking for the coefficients of a linear combination of kernel functions $K(\boldsymbol{x}_i, \cdot)$, i.e. $f_{\boldsymbol{\alpha},b}(\boldsymbol{x}) = b + \sum_{i=1}^{\ell} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x})$. This is done by solving the following type of optimization problem [4]:

$$\min_{\boldsymbol{\alpha},b,\boldsymbol{\xi}\geq\mathbf{0}} \|\boldsymbol{\alpha}\|_P + C\textstyle\sum_{i=1}^{\ell}\xi_i \quad \text{with} \quad y_i f_{\boldsymbol{\alpha},b}(\boldsymbol{x}_i) \geq 1 - \xi_i, \quad i = 1, \ldots, \ell, \tag{1}$$

where $\|\cdot\|_P$ is the 2-norm of $\boldsymbol{\alpha}$ in feature space for SVMs and the 1-norm of $\boldsymbol{\alpha}$ (in the coefficient space) for LPMs, respectively. The data and labels are denoted by $\boldsymbol{x}_i \in \mathbb{R}^n, y \in \{1, -1\}$ respectively. C is the regularization parameter. When solving (1), one usually introduces Lagrange multipliers λ_i, $i = 1, \ldots, \ell$ for the constraints in (1), which are zero if the constraint is not active. For SVMs they turn out to be equal to α_i. For LPM there is no such correspondence.

We will now review some bounds on the LOO error of SVMs that have been proposed. A more complete presentation can be found in e.g. [2,8]. Let Z be a sample of size ℓ, where each pattern is defined as $z_i = (\boldsymbol{x}_i, y_i)$. Furthermore, define $Z^p = \{z_i \in Z, i \neq p\}$ and $f^p = \mathcal{L}(Z^p)$, i.e. f^p is the decision obtained when learning with the p-th sample left out of the training set. The LOO error is defined as

$$\epsilon^{loo}(Z) = \frac{1}{\ell}\sum_{p=1}^{\ell} \Psi(-y_p f^p(\boldsymbol{x}_p)), \tag{2}$$

where $\Psi(z) = 1$ for $z > 0$ and $\Psi(z) = 0$ otherwise. As $-y_p f^p(\boldsymbol{x}_p)$ is positive only if f^p commits an error on $\boldsymbol{x}_p$, $\epsilon^{loo}(Z)$ is the average number of patterns which are misclassified when they are left out.

Support Vector Count: SVMs have several useful properties that can be exploited for a LOO prediction: The first is that patterns which are not Support Vectors (SVs) do not change the decision surface and are always correctly classified. Therefore, one has to consider *only the SVs* in the LOO procedure and the LOO error can be easily bounded by

$$\epsilon^{loo}(Z) \leq \frac{\#SV}{\ell}, \tag{3}$$

where $\#SV$ is the number of SVs. However, this is a very rough estimate, because not all SVs will be misclassified when they are removed from the training set. For LPMs, (3) also holds, if one defines the SVs to be the patterns $\boldsymbol{x}_i$ whose expansion coefficients α_i or corresponding Lagrange multipliers λ_i are non-zero.

Jaakkola-Haussler Bound: In [3], Jaakkola and Haussler proposed a tighter bound than Eq.(3)

$$\epsilon^{loo}(Z) \leq \frac{1}{\ell}\sum_{i=1}^{\ell} \Psi(\alpha_p K(\boldsymbol{x}_p, \boldsymbol{x}_p) - y_p f(\boldsymbol{x}_p)), \tag{4}$$

where α_i is the weight coefficient of SVM and K is the kernel function.

Span Predictions: Recently, a sophisticated way of predicting the LOO error using the "span" of support vectors has been proposed [8]. Under the assumption that the set of SVs *does not change* during the LOO procedure, the LOO error can be exactly rewritten as

$$\epsilon^{loo}(Z) = \frac{1}{\ell}\sum_{p \in SV} \Psi(-y_p f(\boldsymbol{x}_p) + \alpha_p S_p^2), \tag{5}$$

where SV denotes the set of support vectors and S_p is a geometric value called "span" of a support vector [8]. Unfortunately, in practice the above assumption made is not always satisfied, however, experimentally it is shown that this approximation works very well [8]. For computing the span one needs to solve an optimization problem which can be very expensive, if the number of SVs is high.

3 Meta Learning: Predicting the Generalization Error from Empirical Data

We will now outline a learning framework to predict the LOO error. The LOO error ϵ^{LOO} depends on the data set Z and the parameters of the learning machine $\boldsymbol{\theta}$. We assume that the LOO errors could be measured for several data sets and various combinations $(Z_1, \boldsymbol{\theta}_1), \cdots, (Z_m, \boldsymbol{\theta}_m)$. Then, a meta-learning machine is trained to predict ϵ^{LOO} on unseen data based on appropriate features extracted from $(Z, \boldsymbol{\theta})$.

Predicting the LOO-Error as a classification problem Recall that the LOO error can be represented by each LOO result $r(\boldsymbol{x}_p)$:

$$\epsilon^{LOO} = \frac{1}{\ell}\sum_{p=1}^{\ell} \Psi(-y_p f^p(\boldsymbol{x}_p)) := \frac{1}{\ell}\sum_{p=1}^{\ell} \Psi(-r(\boldsymbol{z}_p)) \tag{6}$$

where $r(\boldsymbol{z}_p) = \mathrm{sgn}(y_p f^p(\boldsymbol{x}_p))$. So, to predict the LOO error, it is sufficient to predict the result of each LOO procedure, i.e. *whether or not* an error will be made, if a certain pattern is left out. This meta learning problem is a binary classification problem.

For kernel classifiers, the learning scheme can be designed as Fig.1: Here, a coefficient α_i is attached to every training sample $\boldsymbol{x}_i$. Features are extracted for the left-out sample $(\boldsymbol{x}_p, \alpha_p)$ as well as for the neighboring samples. The neighbors are included since they are likely to affect the LOO result as shown in Fig. 2. In taking the neighbors, we do not care whether they are support vectors or not. So, the features also include information from non-support vectors whereas the span bound (5) is derived from the support vectors only.

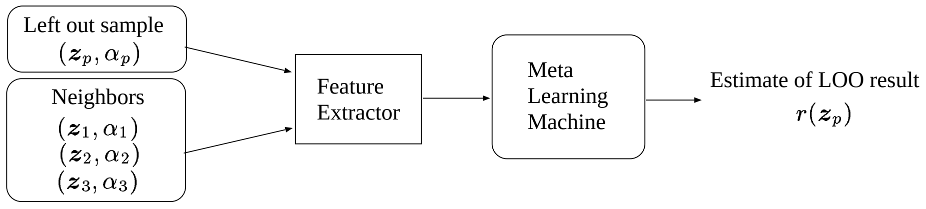
Left out sample
$(\boldsymbol{z}_p, \alpha_p)$
Neighbors
$(\boldsymbol{z}_1, \alpha_1)$
$(\boldsymbol{z}_2, \alpha_2)$
$(\boldsymbol{z}_3, \alpha_3)$
Feature Extractor
Meta Learning Machine
Estimate of LOO result
$r(\boldsymbol{z}_p)$

Fig. 1: Learning scheme for predicting the LOO result.

Features for meta classification To include the local geometry around a left-out sample, the features for LOO are extracted as follows: for the left-out sample $\boldsymbol{x}_p$, we use

– Weight α_p, Dual weight λ_p (LPM only), Margin $f(\boldsymbol{x}_p)/\|\boldsymbol{w}\|^2$.

Additionally, we calculate the following quantities from the 3 nearest neighbors $\boldsymbol{x}_k$ of $\boldsymbol{x}_p$:

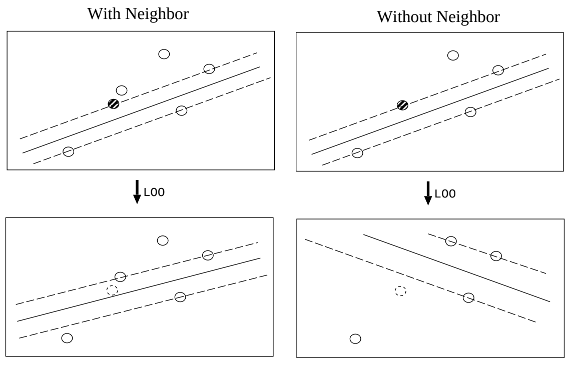

Fig. 2: Consider the case with a non-SV near the SV of interest as in the left panel. In the leave-one-out procedure, the boundary is re-estimated without this SV, but the non SV takes its part and the boundary does not change significantly. The LOO boundary could show a large difference, if there was no close non-SV neighbor as in the right panels.

- Distance in input space $\|\boldsymbol{x}_k - \boldsymbol{x}_p\|/D_i$,
- Distance in feature space $\|\Phi(\boldsymbol{x}_k) - \Phi(\boldsymbol{x}_p)\|/D_f$,
- Weight α_k, Dual weight λ_k (LPM only), Margin $f(\boldsymbol{x}_k)/\|\boldsymbol{w}\|^2$,

where D_i, D_f are the maximum distances between training samples in the input space and the feature space, respectively. All these features of $\boldsymbol{x}_p$ form a vector $\boldsymbol{v}_p$ which is used together with the label $r(\boldsymbol{z}_p)$ to *learn* the LOO error prediction $\{\boldsymbol{v}_p\} \rightarrow \{r(\boldsymbol{z}_p)\}$ for all p patterns. In this work we built the meta-classifier as a linear programming machine with polynomial kernel of degree 2, which employs the 1-norm regularization (in feature space) leading to sparse weight vectors. This turns out to be beneficial as the (meta) LPM automatically selects the relevant features.

4 Experiments

The motivation of the following experiment is to answer the following questions: (1) Does the LOO error predictor learn from a given data set to generalize well on unseen data sets? (2) Is the prediction good enough for reliable model selection?

4.1 Two Class Benchmarks

In our study, we considered three data sets[1]: `twonorm, ringnorm` and `heart`. Here, `twonorm` and `ringnorm` are quite similar because the input dimensionality is 20 and the number of training samples is 400 for both data sets. But, `heart` is a quite different data set, where the input dimensionality is 13 and the number of training samples is only 170. For the evaluation of our meta classifier we use the following experimental setup: For each data set we considered ten realizations (train/test-splits, used for averaging and obtaining error bars). On each realization, we trained SVMs and LPMs for wide ranges of the regularization constant C and the (RBF) kernel parameter σ, i.e. $K(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\|\boldsymbol{x} - \boldsymbol{y}\|/\sigma^2)$. For each training sample we extracted the features described in Section 3. These features and the corresponding labels (which have been computed by the actual LOO procedure) are used for training and testing our classifier. We learned from two data sets and tested on the third. To evaluate the performance, the *model selection error* is used: First, the kernel parameter σ and regularization constant C are selected where the predicted LOO error is minimized. The model selection error is then defined as the classification error on the test set at the chosen parameters.

The results for SVM and LPM are shown in Fig. 3 and 4, respectively. Results on the span bound and JH bound are not available for LPMs, as the respective bounds simply do not exist. The experiments show that in most cases our LOO predictor performs almost as well as the actual (highly expensive) LOO calculation. Compared to the bounds we observe that both methods achieve similar

[1] The data sets incl. training/test splits and the LOO results can be obtained at `http://ida.first.gmd.de/~raetsch`.

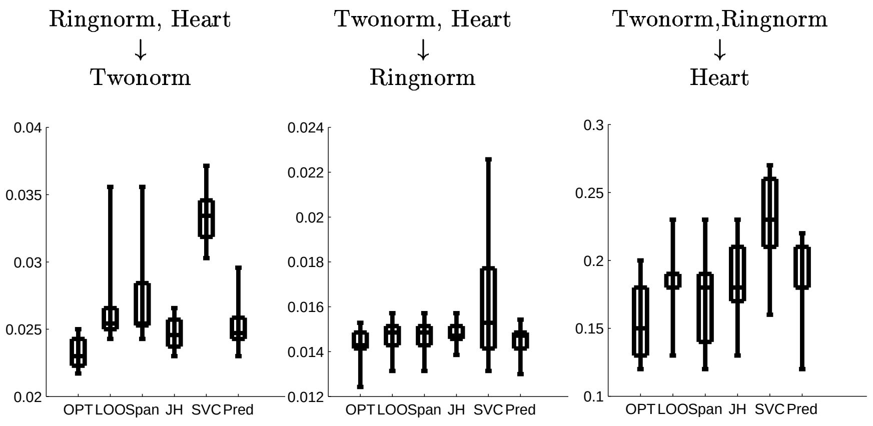

Fig. 3: Model selection errors in SVM. The labels denote: OPT: optimal choice based on the test error, LOO: actual leave-one-out calculation, Span: span bound, JH: Jaakkola-Haussler Bound, SVC: Support Vector Count, Pred: our method. Two data sets are used for training and the test error is assessed on the third one.

performance, note however that our method can also be applied to LPMs. Comparing the three cases in Fig. 4, our method performs slightly worse when `heart` is used for testing in comparison with the other two cases. This shows the tendency that LOO error prediction works well if data with a similar characteristics is contained in training set. This indicates also that the statistics of the feature set still varies considerably for different data sets. However, it is still surprising that our simple features can show such a good generalization performance in benchmark problems.

4.2 A Multiclass Example

As one application of our approach, we consider multi-class problems. In solving c-class problems, it is common, e.g. for SVMs, to use c single (two-class) classifiers, each of which is trained to discriminate one class from all the other classes. Here, the hyperparameters of each SVM have to be properly set by some model selection process. Clearly, it takes prohibitively long to perform leave-one-out procedures in all SVMs for all possible hyperparameter settings. To cope with this problem, the leave-one-out procedure is performed with respect to only one of the c classification problems and our meta classifier is then trained based on this result. Then, the hyperparameters of the other SVMs can be efficiently selected according to the LOO error predictions from the meta classifier.
We performed an experiment with the `3dobj` data set[2] with 8 classes and 640 samples (i.e. 80 samples per class). The task is here to choose a proper value of kernel width σ from a prespecified set of 11 values. The sample is randomly divided into 320 training and 320 test samples for obtaining error bars on our

[2] This dataset will be added to the IDA website.

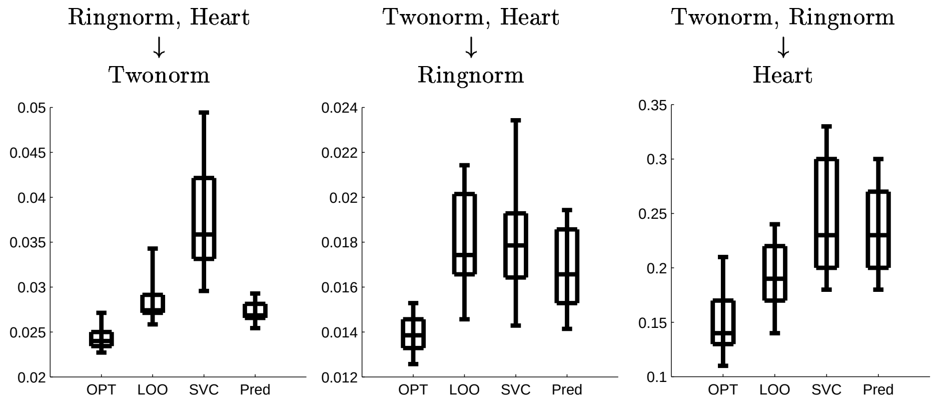

Fig. 4: Model selection errors in LPM.

results. One class is chosen for the training and the remaining classes are used for testing. For each σ in $\Sigma = \{0.4, 0.6, \cdots, 2.4\}$, we computed the LOO error predictions using the meta classifier.

Figure 5 shows the model selection errors of SVMs and LPMs. Here, our method performed as well as the actual LOO calculation for both: SVMs and LPMs. The task of model selection in multiclass problems appears very suitable for our method, because the data sets for training and testing are considered to have similar statistical properties.

5 Conclusion

To train a learning machine that learns about the generalization behavior of a set of learning machines seems like an appealing idea and introduces a *meta* level of reasoning. Our goal in this work was to obtain an effective meta algorithm for predicting the LOO error from past experience, i.e. from a variety of data sets. By

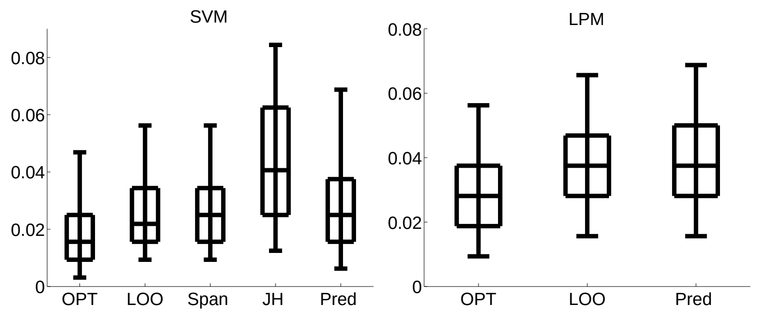

Fig. 5: Model selection errors in the multiclass problem.

adding the twist of casting the meta learning problem into a specific classification task allowed to achieve an accurate and fast empirical estimate of the LOO error of SVM and LPM. The crucial point was to use simple geometrical features for classifying whether a given training sample would be misclassified, if left out. Once given a reliable LOO error estimate it can easily be used for model selection. Careful simulations (using approximately 1.5 CPU years of 800Mhz Pentium III mostly for obtaining the actual LOO error) show that our meta learning framework compares favorably over the conventional bounds. Apparently, our heuristic geometrically motivated features have a good generalization ability for different data sets. We speculate that using these features could provide new ways to improve bounds, in a way, by integrating particularly meaningful features from our meta learning problem into new learning theoretical bounds. Future research will be dedicated to a further exploration of the meta learning idea also for other learning machines and to gaining a better learning theoretical understanding of our findings.

Acknowledgments We thank for valuable discussions with Jason Weston, Olivier Chapelle and Bernhard Schölkopf. This work was partially funded by DFG under contract JA 379/-71, -91, MU 987/1-1.

References

1. K.P. Bennett and O.L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, 1:23–34, 1992.
2. O. Chapelle and V.N. Vapnik. Choosing kernel parameters for support vector machines. Personal communication, mar 2000.
3. T.S. Jaakkola and D. Haussler. Probabilistic kernel regression models. In *Proceedings of the 1999 Conference on AI and Statistics*, 1999.
4. K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 2001. in Press.
5. M. Opper and O. Winther. Gaussian processes and SVM: Mean field and leave-one-out. In A.J. Smola, P.L. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 311–326, Cambridge, MA, 2000. MIT Press.
6. V.N. Vapnik. *Estimation of Dependences Based on Empirical Data.* Springer-Verlag, Berlin, 1982.
7. V.N. Vapnik. *The nature of statistical learning theory.* Springer Verlag, New York, 1995.
8. V.N. Vapnik and O. Chapelle. Bounds on error expectation for support vector machines. *Neural Computation*, 12(9):2013–2036, September 2000.
9. G. Whaba, Y. Lin, and H. Zhang. Generalized approximate cross-validation for support-vector-machines: another way to look at margin-like quantities. Technical Report TR-1006, Dept. of Statistics, University of Wisconsin, April 1999.