Machine Learning, 54, 195–209, 2004

A Meta-Learning Method to Select the Kernel Width in Support Vector Regression

CARLOS SOARES csoares@liacc.up.pt
PAVEL B. BRAZDIL pbrazdil@liacc.up.pt
LIACC/Faculty of Economics, University of Porto, Portugal

PETR KUBA xkuba@informatics.muni.cz
Masaryk University, Brno, Czech Republic

Editor: Peter Flach

Abstract. The Support Vector Machine algorithm is sensitive to the choice of parameter settings. If these are not set correctly, the algorithm may have a substandard performance. Suggesting a good setting is thus an important problem. We propose a meta-learning methodology for this purpose and exploit information about the past performance of different settings. The methodology is applied to set the width of the Gaussian kernel. We carry out an extensive empirical evaluation, including comparisons with other methods (fixed default ranking; selection based on cross-validation and a heuristic method commonly used to set the width of the SVM kernel). We show that our methodology can select settings with low error while providing significant savings in time. Further work should be carried out to see how the methodology could be adapted to different parameter setting tasks.

Keywords: meta-learning, parameter setting, support vector machines, Gaussian kernel, learning rankings

1. Introduction

Support Vector Machines (SVMs) have become widespread in the last decade, due to both sound theoretical foundations and good empirical results obtained in many problems. However, the SVM algorithm is very sensitive to the adequate choice of parameter values (Cristianini & Shawe-Taylor, 2000). According to Müller et al. (2001), the most common approach consists of pre-selecting, according to some heuristic, a set of candidates, estimating the error of each of them, and choosing the one with lowest expected error. This is often rather time consuming and the lack of sufficient resources (e.g., time or computational power) may prevent the user from finding the appropriate set of parameter values. Consequently, the user may need to content himself with suboptimal results.

Here we explore the possibility of applying the meta-learning approach to the problem of determining good parameter settings for SVMs with Gaussian kernel in the context of regression. Our results show that it is possible to use information about the past performance of different settings of the kernel width to predict their relative performance on new problems. We also show that our methodology provides recommendations that enable the user to obtain good performance in terms of error, while achieving significant gains in time by reducing the number of alternatives tried out.

The paper is organized as follows. As SVMs are well documented in the literature we will not describe the algorithm here, but rather survey current approaches to the problem of selecting parameter settings (Section 2). In Section 3 we describe our meta-learning methodology for setting the kernel width of SVMs with Gaussian kernel. The results of the experimental evaluation are given in Section 4. In Section 5 we discuss the sensitivity of the approach to some choices that need to be made. The results of our methodology are compared with an alternative heuristic approach in Section 6. Some limitations and future work are discussed in Section 7 and the conclusions are presented in Section 8.

2. Methods to set parameters of SVMs

There are several methods to set the parameters of SVMs. One approach is to choose the setting with the best generalization error, which is estimated from the empirical error. Three types of approaches are common: cross-validation (CV), the Bayesian evidence framework and the PAC framework, the first being the most common choice (Müller et al., 2001). Given that the SVM model has to be induced for each setting to estimate its empirical error and the computational requirements of SVMs are significant both in the training and in the test phases (Burges, 1998), these approaches to parameter selection are computationally demanding, especially when dealing with large data sets.

Recently, optimization approaches to set a single parameter (Cristianini, Shawe-Taylor, & Campbell, 1998) or multiple parameters (Chapelle et al., 2002) have been proposed. Again, these approaches can be computationally very expensive because the SVM algorithm must be executed for each value selected by the optimization method.

To avoid this problem, the choices concerning parameter settings are often driven by heuristics. An example of a heuristic to select the class of kernel is given by Smola and Schölkopf (1998): the Gaussian kernel is a good choice when only the smoothness of the data can be assumed. A different heuristic is used by Jaakkola, Diekhans, and Haussler (1999) to set the width of the Gaussian kernel beforehand, which is based of the distances between the examples in attribute space. This approach is discussed in more detail in Section 6.

One approach to setting the kernel width using meta-learning was described by Kuba et al. (2002). The authors used regression to predict the appropriate kernel width (σ) from a set of characteristics of data sets. The meta-data used consists of the performance of a set of σ values on previously processed data sets. Although the initial results were promising, it was difficult to demonstrate that the method could consistently lead to better settings than a given baseline (e.g., a given default setting). This led us to investigate a different meta-learning approach, based on rankings of the candidate values (Brazdil, Soares, & Costa, 2003). This approach is described in the remainder of this paper.

3. Recommendation of the kernel width with meta-learning

Here we describe the meta-learning methodology for recommendation of the kernel width parameter of SVMs. In this paper, we focus on regression tasks and SVMs with Gaussian kernel. This kernel can be represented by:

$$e^{-\frac{\|\mathbf{x}\cdot\mathbf{y}\|^2}{2\sigma^2}}$$

where **x** and **y** are the attribute vectors for two examples and the numerator is the squared 2-norm of the two vectors. Given appropriate parameter settings, any function can be represented by this type of kernel (Cristianini & Shawe-Taylor, 2000). Its parameter is the width of the distribution (σ), which controls the linearity of the induced model: the larger the width, the more linear the function (Burges, 1998).

The methodology involves (1) the generation of meta-data concerning the performance of a set of σ settings on existing data sets, and (2) the use of the meta-data generated to obtain a prediction concerning which σ setting to use on a new data set. Both points will be described in detail in the following sections.

3.1. Generation of meta-data

Obtaining the meta-data requires two different types of operations, namely running experiments on the available data sets with the given algorithm and selected parameter settings, and then computing a set of meta-features describing those data sets. Both of them will be discussed in the following sections.

3.1.1. Selecting parameter settings. As σ is a continuous parameter, we cannot compute the performance for all the possible settings and, so, a finite subset must be chosen. The number of alternatives must be carefully defined taking into account a compromise between the quality of results and computational requirements. On one hand, we want to have a sufficient number of settings, enabling us to identify, in every case, one that is not far from the best possible setting. On the other hand, if too many options were considered, the computational requirements to run all associated experiments and collect the meta-data necessary for meta-learning could be too high.

Analysis of the performance of SVMTorch (Collobert & Bengio, 2001) on 16 data sets has led us to fix the range of σ values to [0.25, 256000]. From this range, we have selected 11 alternatives, following a geometric progression with factor 4.[1] The choice of a geometric progression was motivated by the observation that, in the 16 data sets, a small change in the value of σ had a larger effect on the performance for small σ values. So, the final set of values is the sequence 0.25, 1,…, 256, 1000, 4000,…, 256000. Further analysis regarding the choice of 11 values is given in Section 5. Additionally, we have set the width of the error tube $\epsilon = 0.008$ and the regularization parameter $C = 100$, based on the same preliminary study.

3.1.2. Collecting performance data. Various performance measures exist to evaluate regression algorithms. One commonly used measure is the Normalized Mean Squared Error:

$$NMSE = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y}_i)^2}$$

where n is the number of cases, y_i and $\hat{y}_i$ are the target and the predicted values for case i, and $\bar{y}_i$ is the mean of the target values. The values of NMSE range from 0 to 1. The

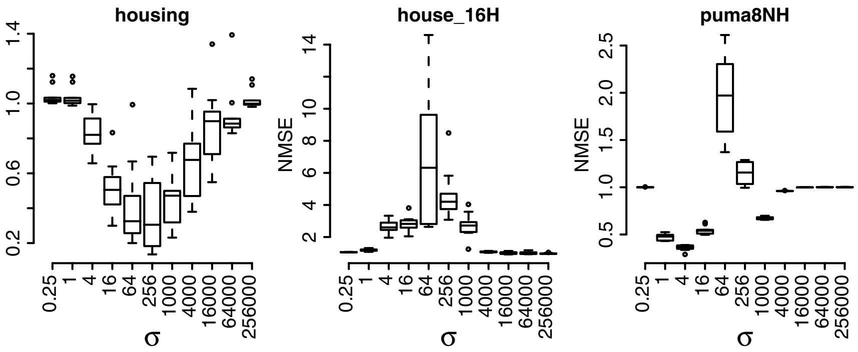

Figure 1. Distributions of NMSE for the selected σ values on data sets *housing*, *house_16H* and *puma8NH*.

performance of a baseline strategy of predicting the mean target value achieves a NMSE of 1. One property of NMSE which is essential for our purpose is that the values are comparable across different data sets. Performance of SVMTorch was estimated by 10-fold cross-validation.[2]

Let us examine how the error varies as the kernel width is varied for some data sets. The box-and-whisker plots[3] of the error for three data sets are presented in figure 1. The error on *housing* is U-shaped, having higher errors for the extreme values of σ. However, in some data sets there are high error peaks in the mid-range. This is one of the reasons why it is difficult to establish a good default setting for this parameter.

3.1.3. Data set characterization. For the purpose of algorithm recommendation, an adequate set of *meta-features*, i.e., data set characteristics, must satisfy the following two conditions. First, it must capture information that is useful for determining the relative performance of the individual learning algorithms. Second, it should not be too difficult to calculate them. To be more precise, it should be computationally cheaper to calculate the measures than to run the individual candidate algorithms, otherwise one might just as well run them (Pfahringer, Bensusan, & Girand-Carrier, 2000). In this paper, we have used the meta-features described in Kuba et al. (2002), which are listed in Table 1.[4]

3.2. Prediction with the k-NN ranking method

The use of the k-NN ranking method to provide recommendation of values for the kernel width parameter involves the following steps

- compute the meta-features for the data set in question,
- search for the k nearest neighbors among the existing data sets,
- retrieve the rankings of parameter settings on the nearest neighbors and aggregate this information to generate the recommended ranking.

The meta-features are the data characteristics of the data set discussed earlier. To identify the k nearest neighbors, the distance between the meta-features of the data set in question

Table 1. Measures used to characterize data sets (*meta-features*).

Number of examples
Number of attributes
Proportion of symbolic attributes
Ratio of the number of examples to the number of attributes
Proportion of the attributes with outliers
Coefficient of variation of the target (ratio of the standard-deviation to the mean)
Sparsity of the target (coefficient of variation discretized into 3 values)
Presence of outliers in the target
Stationarity of the target (the standard-deviation is larger than the mean)
R^2 coefficient of linear regression (without symbolic attributes)
R^2 coefficient of linear regression (with binarized symbolic attributes)
Average absolute correlation between numeric attributes
Average absolute correlation of numeric attributes to the target
Average dispersion gain

and all the others is calculated, and k data sets with the smallest distance are selected. The function used here is a version of the unweighted L_1 norm, adapted to measure the distance between characterizations of data sets (Brazdil, Soares, & Costa, 2003).

The k target rankings, corresponding to each of the k retrieved cases, are aggregated to generate the predicted ranking for the new case using the Average Ranks (AR) method. Let r_j^i be the rank of the candidate setting $j = 1..n$ on the neighboring case $i = 1..k$. The average rank for each setting is $\overline{r_j} = \sum_i r_j^i / k$. The final ranking is obtained by ordering the $\overline{r_j}$ values and assigning ranks to the candidates accordingly. An example of a recommendation for the data set *house_8L* is given in the second row of Table 2.

4. Experiments

In this section we describe the results of the empirical evaluation of the methodology proposed. We describe the experimental setup, including the baselines against which we

Table 2. Example rankings. Target ranking for data set *house_8L*, based on CV estimates (first row), recommended ranking obtained with the 1-NN ranking method for the same data set (second row) and the default ranking used as baseline, based on the mean NMSE across all data sets (last row). The k in the σ values represents 1000. The calculation of the correlation is explained in Section 4.2.

	σ											
Ranking	0.25	1	4	16	64	256	k	$4k$	$16k$	$64k$	$256k$	Correlation to target
Target	1	2	11	5	6	8	9	10	7	4	3	1.00
Recommended	1	4	2	3	5	6	7	8	9	10	11	0.06
Default	11	7	8	6	10	4	2	1	3	5	9	−0.59

compare our results (Section 4.1). The first evaluation measure considered is *ranking accuracy*, which is based on the correlation between the recommended and the target rankings (Section 4.2). Next, we apply the top-N evaluation methodology, which takes into account not only the error but also the computational costs (Section 4.3).

4.1. *Experimental setup and baselines*

We have used a total of 42 data sets, some of which are publicly available benchmarks while others were donated to the METAL Project (2002). Whenever the data sets contained a very large number of examples, a random sample of size 10.000 was used to limit the computational time. As the SVM algorithm was designed to work with numerical attributes only, all symbolic attributes were binarized by creating $k - 1$ attributes for an attribute with k values.[5] The values of the target attribute were normalized to make them comparable across different data sets: $y_i = (y_i - \overline{y_i})/std(y)$, where y_i is the target value for case i, and $\overline{y_i}$ and $std(y_i)$ are the mean and standard deviation of the target values, respectively. Estimates of meta-learning performance were obtained with a leave-one-out procedure.

We consider three baselines: a default ranking, the choice of the best setting using cross-validation of all alternatives and random selection. The default ranking of the variants considered (i.e., settings of the kernel width) is based on the mean NMSE across all data sets. The default ranking for the experimental setup adopted here is shown in the third row of Table 2. The best setting is $\sigma = 4000$. This value is followed by the neighboring settings (1000 and 16000, ranked 2nd and 3rd, respectively), and so on.[6] This indicates that the behavior of the error is generally smooth, although, as shown in figure 1, this is not always true.

Cross-validation consists of executing all the alternatives and selecting the one with the lowest error. This strategy minimizes the error (the mean NMSE across all data sets is 0.30) but it can be very time consuming, depending on the number of examples and attributes in the data set, and on the number of σ values considered.

Finally, random selection consists of choosing a σ value for each data set randomly from the set of pre-selected values. The expected error and time of this strategy are the mean error and the mean time of all experiments, respectively. For the set of alternatives considered the mean error is 0.78 and the mean execution time is approximately 5 minutes.

4.2. *Evaluation of ranking accuracy*

The similarity between the recommended and the target rankings is measured with the Spearman rank correlation coefficient, $r_s = 1 - 6\sum_{i=1}^{m}(rr_i - tr_i)^2/(m^3 - m)$, where rr_i and tr_i are the recommended and target ranks of setting i respectively, and m is the number of parameter settings.[7] The value of 1 represents perfect agreement, and -1 perfect disagreement. A correlation of 0 means that the rankings are not related, which would be the expected score of the random ranking method. Full details on this evaluation methodology were described by in Brazdil, Soares, and Costa (2003).

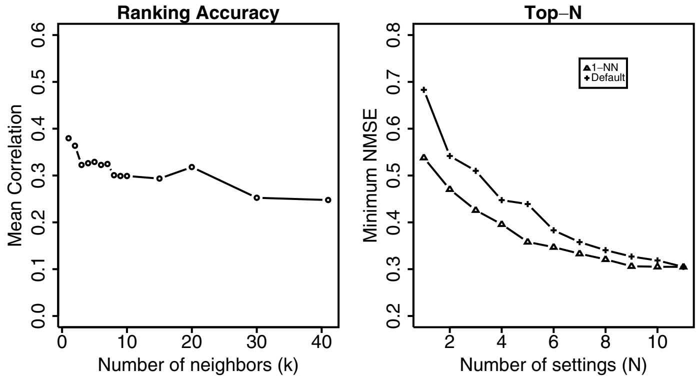

Figure 2. Ranking accuracy and Top-N evaluation of the k-NN ranking method.

Figure 2 (left-hand side) shows the mean ranking accuracy across all data sets for different number of neighbors (k), representing data sets. The point represented as $k = 41$ corresponds to the default ranking baseline. The results show that our k-NN ranking method is able to generate more accurate rankings than the default ranking. Additionally, we can observe a clear tendency for the accuracy to decrease with increasing number of neighbors.[8] This indicates that the data characteristics used contain useful information for ranking different parameter settings of the Gaussian kernel.

4.3. *Top-N evaluation*

One of the advantages of providing a recommendation in the form of a ranking is that the user may try several alternatives, thus increasing the probability of obtaining a good result. This may still lead to significant gains in time when compared to trying all the candidates. It is reasonable to expect that the order recommended will be followed, i.e., the candidate ranked at the top will be executed first, followed by the candidate ranked second, etc. As it is not known beforehand how many alternatives will be considered, we adopt the top-N evaluation: for different values of N, we simulate that the top N alternatives are executed. While varying N, the quality of the recommendation is defined as:

- the *minimum* error (NMSE) achieved with the settings tried out, and
- a measure of the *total* computational cost of executing them.

Top-N performance can be visualized by plotting the error achieved versus the number of alternatives executed. This kind of top-N evaluation is commonly used in other fields, like Information Retrieval and recommender systems (Järvelin & Kekäläinen, 2000; Karypis, 2001), where both benefits and costs of solutions are usually taken into account.

The results for 1-NN ranking are plotted in figure 2 (right-hand side). We have used $k = 1$ (i.e., the recommendation is based on a single neighboring data set) because it is generally expected to obtain good results (Ripley, 1996). The results indicate that our methodology is able to achieve better mean NMSE than the default ranking baseline. A good value of N, i.e., the number of algorithms in the ranking to execute, can be determined by identifying the point where the top-N curve levels off. In our case, any value $N \geq 5$ seems suitable for this purpose.

Additional evidence can be obtained by performing pairwise comparisons of the best alternatives found for different N. The statistical significance of the differences between two alternatives was assessed using a Wilcoxon Sign Rank test with a significance level of 0.05. Comparing top-1 for the recommended ranking and the default ranking, the former wins in 20 data sets, losing only 11 times. When comparing the top-5 settings, our methodology achieves 16 wins and 2 losses only when compared to the default ranking. This confirms that our methodology provides better recommendations than the default ranking.

Let us now compare the k-NN ranking method with the other two baselines defined in Section 4.1, namely cross-validation (CV) and random selection. CV is represented in figure 2 as top-11 (i.e., the last case in the sequence). We observe that the mean error of top-1 (0.538) is 0.23 higher than the error of CV (0.304), but it saves a lot of time and it is much better than the default σ value of SVMTorch (see Section 4.1), which obtains a mean NMSE close to 0.8.[9] Furthermore, it also represents a significant gain when compared to random selection (0.783). This means that the top-1 setting is systematically much better than one chosen at random.

If we consider the top-5 alternatives, which means running less than half of the parameter settings, the mean error obtained is 0.358, which is only 0.05 higher than the mean error obtained with CV. Additionally, despite executing less than half the alternatives tried out with CV, our methodology achieves a statistically comparable performance on 35 out of the 42 data sets.

5. Robustness to changes in the set of σ values and the ϵ value

Earlier we have described the choice of 11 σ values and the ϵ value used in our meta-learning experiments (Section 3.1.1). In the following sections we discuss the issue of choosing particular settings in detail.

5.1. *Effect of changing the set of σ settings on SVM error*

We start by defining criteria to determine whether a set of pre-selected parameter settings is adequate for our purposes. Given a set of data sets and assuming that the pre-selected set of m parameter settings is $P = p_1, \ldots, p_m$, the requirements are:

1. Adequacy with respect to a baseline method: For every data set there should be a p_i that obtains an error which is lower than the error of a given baseline. Here, the baseline is the mean target value, which, by definition, obtains a NMSE of 1.
2. Relative adequacy: Given some pre-selected set P, the results cannot be further significantly improved by adding additional elements to it.

3. Relevance: For every element p_i, we should be able to identify at least one data set for which p_i is the best alternative from the pre-selected set, P.
4. Non-redundancy: For every p_i, there should not exist a p_j such that the performance of p_i is never significantly better than that of p_j for all data sets considered. In other words, some differences in performance should be statistically significant.

On the majority of the data sets (38), at least one of the σ values achieves an error that is less or equal to 0.8. On the remaining four data sets, the best error is never larger than 1. This means that valid patterns are generally being found in the data and it provides evidence that the first requirement is satisfied.

As for requirement 2), we consider what happens if other sets of values were used. The first set, σ^{21}, follows a geometric progression with factor 2, giving the sequence $0.25, 0.5, 1, \ldots, 512, 1000, \ldots, 256000$ with 21 elements,[10] which is a superset of the original set, referred to here as σ^{11}. Using this extended set of parameter settings only brings noticeable improvement in approximately 5% of the data sets and the mean error achieved only decreases by 0.008. The situation is different with σ^5, which contains five of the original 11 cases (1, 16, 256, 4000 and 64000). The reduction of alternatives lead to an increase in the error in approximately 25% of the data sets and leads to an increase in the mean error by 0.067. These results suggest that the set σ^{11} verifies the second requirement, while σ^5 does not. This is consistent with the results of Chapelle et al. (2002), which indicate that restricting the search for a good value of the kernel width to a subset of pre-defined values does not necessarily imply a significant increase in the error obtained.

Figure 3 (left-hand side) plots the number of data sets for which each setting is ranked in the first position. We observe not only that each variant is the best on at least one data set, but also that none of them is much better than all the others. This shows that the set of values selected verifies requirement 3).

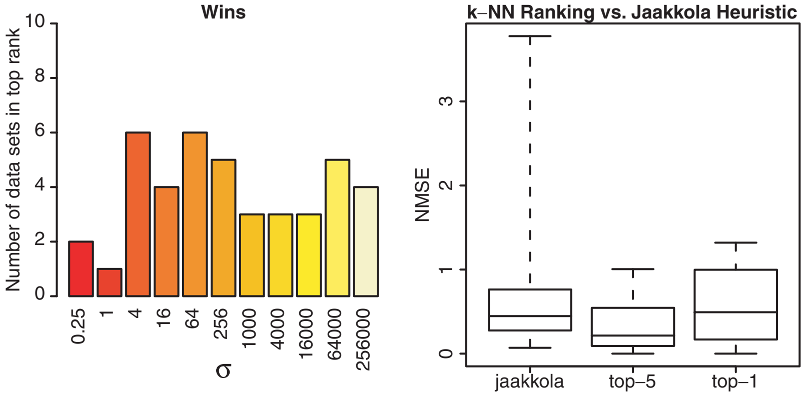

Figure 3. Number of times each parameter setting obtains the lowest error (left-hand side). Empirical distribution of error obtained by selecting parameters with the Jaakkola heuristic and the top-1 and top-5 strategies using 1-NN ranking (right-hand side). Here, the "whiskers" represent the minimum and maximum values.

Finally, requirement 4) is concerned with the non-redundancy of each of the settings selected. If we analyze the NMSE for different pairs of σ settings on individual data sets, we observe that the values will, in general, differ (see, e.g., the examples in figure 1). If we perform pairwise comparisons between the settings using the Wilcoxon Sign Rank test with a significance level of 0.05, we observe a low number of non-significant differences, namely 4.6 out of 55 on average. These results show that the settings are on the whole non-redundant.

In summary, the selected set of σ values is suitable for our meta-learning purposes. The following additional conclusion can be drawn from this analysis. We have determined a relatively small set of values for the width of the Gaussian kernel. We have shown that if these values are tried out before determining the setting, we can obtain better results than with any particular setting chosen apriori from this set. This suggests that algorithms could have an ordered set of default parameter settings, rather than a single recommended default setting. The algorithms could then be extended to run through the given sequence of settings, ordered by their overall performance. This can be seen as an intermediate step before exploiting the meta-learning approach for this purpose.

5.2. *Effect of changing the set of σ settings on meta-learning*

Let us examine how the ranking accuracy and the top-N performance vary for different sets of σ values, namely σ^5, σ^{11} and σ^{21}, described earlier. The results for different number of neighbors are plotted in figure 4. The plot on the left-hand side confirms that the k-NN ranking generates more accurate rankings than the default ranking (i.e., $k = 41$). Note that the behavior with the reduced set σ^5 is less stable than with the others. This can be explained by the fact that the effect of a given rank error on the value of the Spearman correlation is larger when the number of alternatives is smaller. For instance, the correlation between two rankings $1, 2, 3, \ldots, n$ and $2, 1, 3, \ldots, n$ is 0.90 and 0.99 for $n = 5$ and $n = 10$, respectively.

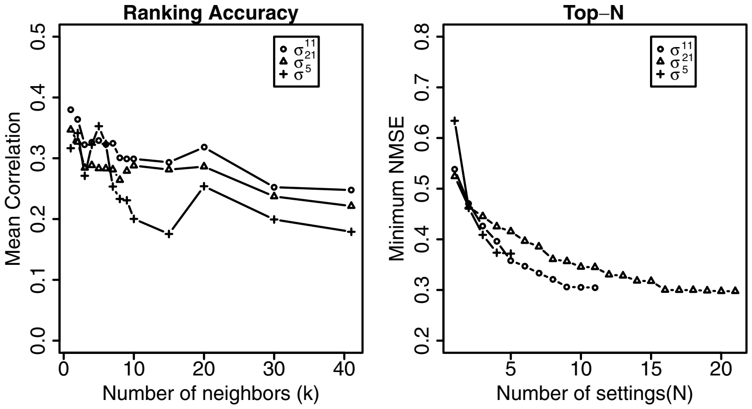

Figure 4. Ranking accuracy and Top-N evaluation for the three different sets of σ values.

The top-N results plotted on the right-hand side show that there is no advantage in either extending or reducing the set of σ values considered. The error obtained with the extended set σ^{21} converges toward the minimum error more slowly than the original set. Additionally, as shown above, the minimum errors are quite similar. Concerning the reduced set of values, the curve has a slightly steeper slope toward lower errors, which represents an advantage when compared with σ^{11}. However, the advantage of using the latter set of values on some data sets becomes evident if five (or more) alternatives are executed.

5.3. *Effect of changing the value of ϵ on meta-learning*

To test how robust the methodology is to changes in the target rankings, we have carried out experiments with the same set of 11 σ settings used earlier but with different values of the width of the error tube, namely $\epsilon = 0.001$ and $\epsilon = 0.128$. We have quantified the similarity between the target rankings for $\epsilon = 0.001$ and $\epsilon = 0.008$ using Spearman correlation, for each data set. The mean correlation is 0.99, which means that the target rankings are similar. We have performed meta-learning experiments, which show, as expected, that similar results are obtained for the two ϵ values, both in terms of ranking accuracy and top-N evaluation.

We have repeated the experiment above for $\epsilon = 0.128$ and quantified the similarity between $\epsilon = 0.008$. The mean correlation between the target rankings is -0.01, indicating that they are quite different. However, the meta-learning results are also similar for the two meta-data sets. Both results indicate that the methodology is robust to differences in the target rankings.

6. Comparison with the Jaakkola heuristic

A heuristic to set the width of the Gaussian kernel for SVM has been used by Jaakkola, Diekhans, and Haussler (1999). The width of the Gaussian kernel is set on the basis of the distances between examples in the attribute space. For each example it is necessary to calculate the minimum distance of a given case **x** to all the others:

$$d_{\mathbf{x}} = \min_{\mathbf{y}} \left(\sqrt{\sum_i (x_i - y_i)^2} \right)$$

where x_i and y_i represent the values of cases **x** and **y** for the independent attribute i. The value of the parameter is set to $\sigma_J = \overline{d_{\mathbf{x}}}$.

We have calculated σ_J for each of the 42 data sets. The values obtained with the Jaakkola heuristic range from 0.04 to 15625. They are consistent with the set of σ values we have used, which ranges from 0.25 to 256000, providing further evidence that our choice is adequate.

In figure 3 (right-hand side) we observe that the error achieved with the top-5 settings is lower than the error achieved by selecting σ with the Jaakkola heuristic. Most importantly, we observe that the NMSE error obtained with the latter strategy is larger than 1 in 7 data sets. This means that the heuristic performs worse than predicting the mean target value on those data sets. This never happens if the top-5 settings are executed.

Even if the algorithm ranked in the top position of the recommendation is the only one tried out, the error achieved is, on average, very similar to the error obtained with the Jaakkola heuristic. However, our methodology appears to be more robust, in the sense that its maximum error is much lower. Additionally, if we perform pairwise comparisons between the parameter setting suggested with the Jaakkola heuristic and the best setting suggested by our methodology (using the Wilcoxon Sign Rank test with a significance level of 0.05), the results show a clear advantage of our methodology. Top-1 wins 22 times and loses 13 times and top-5 wins 32 times and only loses on two data sets.

In terms of computational effort, the Jaakkola heuristic is equivalent to the 1-NN algorithm. This is much more demanding than the meta-learning procedure.

7. Limitations and future work

In this paper we have addressed a specific instance of the problem of recommending parameter settings, focusing on the kernel width of SVM with Gaussian kernel. In principle, the methodology proposed here could be adapted and/or extended to the following situations: (1) Other SVM parameters (e.g., the regularization constant or the width of the error tube) and other types of kernels; (2) Handling multiple parameters at the same time; (3) Other algorithms. Handling multiple parameters may imply a significant increase in the computational requirements, as the meta-learning method requires that different test results be available beforehand. Further work is needed to verify how good the recommendations generated in each case are. We plan to investigate some of these cases in future and compare our approach to other general parameter setting methods (Kohavi & John, 1995).

Given that the goal is to obtain the best possible error while maximizing the savings in experimentation time, another possible improvement to this work is to perform multicriteria ranking, taking also time into account (Brazdil, Soares, & Costa, 2003).

One important aspect we did not investigate here is the selection of appropriate data characteristics that provide information about the relative performance of the candidates. In future we aim to improve the set of measures used here, taking into account the specific problem of selecting parameter settings for SVM. One obvious possibility would be to use the value obtained with the Jaakkola heuristic.

8. Conclusions

In this paper we have investigated how to set the width of the Gaussian kernel. We have adapted previous work in the area of meta-learning that uses information about the past performance of different parameter settings.

We have carried out an extensive empirical evaluation, which involved dozens of regression data sets and a leave-one-out evaluation methodology. We have compared our methodology to the method based on cross-validation. We have shown that good performance can be achieved by executing less than half of the alternatives in the ranking. Thus our methodology can achieve significant savings in time.

We have also compared the recommended rankings of settings to the corresponding target rankings (i.e., reflecting their true relative performance). The results show that the meta-learning methodology is able to predict the target ranking better than the baseline method based on a fixed ordering of the alternatives. So, with our ranking method fewer alternatives need to be tested to achieve a similar error.

We have examined different choices for the set of σ parameter values considered and have shown that our methodology is relatively robust to changes in granularity and that our initial choice of values was satisfactory.

We have compared the methodology proposed to one common strategy in SVM, the Jaakkola heuristic. Our methodology achieves better or comparable results and it appears to be more robust.

One additional observation is concerned with default parameter settings. Algorithm providers usually supply a single default setting per parameter. Our work shows that it would be advantageous to provide a ranking of different defaults and we have shown how these can be determined.

The work described here is by no means complete. Further work should be carried out to verify how the method could be adapted to other similar parameter setting tasks and extended to deal with combinations of parameters.

Appendix A: Detailed results

In this paper we have summarized the results of an extensive body of experiments. More details can be found in an on-line appendix. It can be obtained from this journal website (located at the time of this writing at `http://www.kluweronline.com/issn/0885-6125`).

Acknowledgments

We would like to thank the anonymous referees for their constructive comments which helped to improve this paper. We are also grateful to those who have donated data sets and to Rita Ribeiro for providing us with her data characterization functions in R (Ihaka & Gentleman, 1996). The financial support from ESPRIT project METAL, FEDER, Programa de Financiamento Plurianual de Unidades de I&D and from the Faculty of Economics of Porto is gratefully acknowledged.

Notes

[1] The 7th element was set to 1000 rather than to 1024.

[2] The MLEE package was used to estimate performance (Petrak, 2002).

[3] The box is defined by the first and third quartiles and the middle line represents the median. Points outside the “whiskers” are considered outliers, assuming a normal distribution.

4. To obtain more details concerning these measures, as well as the experiments described in this paper, see Appendix A.
5. The kth value is represented by zeros on all $k - 1$ binary attributes.
6. According to a personal communication by Ronan Collobert, the default value for the width of the Gaussian kernel in SVMTorch is $\sigma = 10$. The default ranking indicates that a better default value could have been used because two close values, namely 4 and 16, are assigned low ranks (8th and 6th, respectively).
7. This is a standard approximation to the Spearman coefficient, adopted for computational reasons (Neave & Worthington, 1992)
8. Note that the aggregation does not weigh the contribution of each neighbor by its distance.
9. We do not have results for the default parameter value $\sigma = 10$, but the mean NMSE of $\sigma = 4$ and $\sigma = 16$ are 0.83 and 0.78, respectively. We expect the mean error of the default parameter value to be similar.
10. The 13th element was set to 1000 rather than 1024.

References

Brazdil, P., Soares, C., & Costa, J. (2003). Ranking learning algorithms: Using IBL and meta-learning on accuracy and time results. *Machine Learning, 50:3*, 251–277.

Burges, C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery, 2*, 121–167.

Chapelle, O., Vapnik, V., Bousquet, O., & Mukherjee, S. (2002). Choosing multiple parameters for support vector machines. *Machine Learning, 46:1*, 131–159. Available from `http://www.kernel-machines.org`.

Collobert, R., & Bengio, S. (2001). SVMTorch: Support vector machines for large-scale regression problems. *Journal of Machine Learning Research*, *1*, 143–160.

Cristianini, N., & Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press.

Cristianini, N., Shawe-Taylor, J., & Campbell, C. (1998). Dynamically adapting kernels in support vector machines. In M. Kearns, S. Solla, & D. Cohn (Eds.), *Advances in Neural Information Processing Systems* (vol. 11, pp. 204–210). MIT Press. Available from `http://www.kernel-machines.org`.

Ihaka, R., & Gentleman, R. (1996). R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics, 5:3*, 299–314.

Jaakkola, T., Diekhans, M., & Haussler, D. (1999). Using the Fisher kernel method to detect remote protein homologies. In T. Lengauer, R. Schneider, P. Bork, D. Brutlag, J. Glasgow, M. Mewes, & R. Zimmer (Eds.), *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology* (pp. 149–158), AAAI Press. Available from `http://www.kernel-machines.org`.

Järvelin, K., & Kekäläinen, J. (2000). IR evaluation methods for retrieving highly relevant documents. In N. Belkin, P. Ingwersen, & M.-K. Leong (Eds.), *Proceedings of the 23th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (ACM SIGIR '00)* (pp. 41–48). ACM Press.

Karypis, G. (2001). Evaluation of item-based top-N recommendation algorithms. In *Proceedings of the Tenth International Conference on Information and Knowledge Management* (pp. 247–254). ACM.

Kohavi, R., & John, G. (1995). Automatic parameter selection by minimizing estimated error. In A. Prieditis & S. Russell (Eds.), *Machine Learning: Proceedings of the Twelfth International Conference*. Morgan Kaufmann.

Kuba, P., Brazdil, P., Soares, C., & Woznica, A. (2002). Exploiting sampling and meta-learning for parameter setting support vector machines. In F. Garijo, J. Riquelme, & M. Toro (Eds.), *Proceedings of the Workshop de Minería de Datos Y Aprendizaje of (IBERAMIA 2002)*. pp. 217–225.

METAL Project (2002). Esprit Project METAL (#26.357). `http://www.metal-kdd.org`.

Müller, K.-R., Mika, S., Rätsch, G., Tsuda, K., & Schölkopf, B. (2001). An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks, 12:2*, 181–201. Available from `http://www.kernel-machines.org`.

Neave, H., & Worthington, P. (1992). *Distribution-Free Tests*. Routledge.

Petrak, J. (2002). MLEE Machine learning experimentation environment. Austrian Institute for Artificial Intelligence. `http://www.ai.univie.ac.at/~johann/mlee.html`.

Pfahringer, B., Bensusan, H., & Giraud-Carrier, C. (2000). Tell me who can learn you and I can tell you who you are: Landmarking various learning algorithms. In P. Langley (Ed.), *Proceedings of the Seventeenth International Conference on Machine Learning (ICML2000)* (pp. 743–750). Morgan Kaufmann.

Ripley, B. (1996). *Pattern Recognition and Neural Networks*. Cambridge.

Smola, A., & Schölkopf, B. (1998). From regularization operators to support vector kernels. In *Advances in Neural Information Processing Systems*. Available from `http://www.kernel-machines.org`.

Received October 3, 2002
Revised November 26, 2003
Accepted December 5, 2003
Final manuscript December 5, 2003