Selecting Classification Algorithms with Active Testing on Similar Datasets

Rui Leite[1] and **Pavel Brazdil**[2] and **Joaquin Vanschoren**[3]

Abstract. Given the large amount of data mining algorithms, their combinations (e.g. ensembles) and possible parameter settings, finding the most adequate method to analyze a new dataset becomes an ever more challenging task. This is because in many cases *testing* all possibly useful alternatives quickly becomes prohibitively expensive. In this paper we propose a novel technique, called *active testing*, that intelligently selects the most useful cross-validation tests. It proceeds in a tournament-style fashion, in each round selecting and testing the algorithm that is *most likely to outperform the best algorithm of the previous round* on the new dataset. This 'most promising' competitor is chosen based on a history of prior duels between both algorithms on *similar* datasets. Each new cross-validation test will contribute information to a better estimate of dataset similarity, and thus better predict which algorithms are most promising on the new dataset. We also follow a different path to estimate dataset similarity based on data characteristics. We have evaluated this approach using a set of 292 algorithm-parameter combinations on 76 UCI datasets for classification. The results show that active testing will quickly yield an algorithm whose performance is very close to the optimum, after relatively few tests. It also provides a better solution than previously proposed methods. The variants of our method that rely on cross-validation tests to estimate dataset similarity provides better solutions than those that rely on data characteristics.

1 Background and Motivation

In many data mining applications, an important problem is selecting the best algorithm for a specific problem. Especially in classification, there are hundreds of algorithms to choose from. Moreover, these algorithms can be combined into composite learning systems (e.g. ensembles) and often have many parameters that greatly influence their performance. This yields a whole spectrum of methods and their variations, so that *testing* all possible candidates on the given problem, e.g., using cross-validation, quickly becomes prohibitively expensive.

The issue of selecting the right algorithm has been the subject of many studies over the past 20 years [17, 3, 23, 20, 19]. Most approaches rely on the concept of *metalearning*. This approach exploits characterizations of datasets and past performance results of algorithms to recommend the best algorithm on the current dataset. The term *metalearning* stems from the fact that we try to learn the function that maps *dataset characterizations* (meta-data) to *algorithm performance estimates* (the target variable).

The earliest techniques considered only the dataset itself and calculated an array of various simple, statistical or information-theoretic properties of the data (e.g., dataset size, class skewness and signal-noise ratio) [17, 3]. Another approach, called *landmarking* [2, 12], ran simple and fast versions of algorithms (e.g. decision stumps instead of decision trees) on the new dataset and used their performance results to characterize the new dataset. Alternatively, in *sampling landmarks* [21, 8, 14], the complete (non-simplified) algorithms are run on small samples of the data. A series of sampling landmarks on increasingly large samples represents a partial learning curve which characterizes datasets and which can be used to predict the performance of algorithms significantly more accurately than with classical dataset characteristics [13, 14]. Finally, an 'active testing strategy' for sampling landmarks [14] was proposed that actively selects the most informative sample sizes while building these partial learning curves, thus reducing the time needed to compute them.

Motivation. All these approaches have focused on dozens of algorithms at most and usually considered only default parameter settings. Dealing with hundreds, perhaps thousands of algorithm-parameter combinations[4], provides a new challenge that requires a new approach. First, distinguishing between hundreds of subtly different algorithms is significantly harder than distinguishing between a handful of very different ones. We would need many more data characterizations that relate the effects of certain parameters on performance. On the other hand, the latter method [14] has a scalability issue: it requires that pairwise comparisons be conducted between algorithms. This would be rather impractical when faced with hundreds of algorithm-parameter combinations.

To address these issues, we propose a quite different way to characterize datasets, namely through the *effect that the dataset has on the relative performance of algorithms run on them*. As in landmarking, we use the fact that each algorithm has its own learning bias, making certain assumptions about the data distribution. If the learning bias 'matches' the underlying data distribution of a particular dataset, it is likely to perform well (e.g., achieve high predictive accuracy). If it does not, it will likely under- or overfit the data, resulting in a lower performance.

As such, we *characterize a dataset based on the pairwise performance differences between algorithms run on them*: if the same algorithms win, tie or lose against each other on two datasets, then the data distributions of these datasets are likely to be similar as well, at least in terms of their effect on learning performance. It is clear that the more algorithms are used, the more accurate the characterization

[1] LIAAD-INESC Porto L.A./Faculty of Economics, University of Porto, Portugal, rleite@fep.up.pt
[2] LIAAD-INESC Porto L.A./Faculty of Economics, University of Porto, Portugal, pbrazdil@inescporto.pt
[3] LIACS - Leiden Institute of Advanced Computer Science, University of Leiden, Nederlands, joaquin@liacs.nl

[4] In the remainder of this text, when we speak of *algorithms*, we mean *fully-defined algorithm instances* with fixed components (e.g., base-learners, kernel functions) and parameter settings.

will be. While we cannot run all algorithms on each new dataset because of the computational cost, we can run a fair amount of CV tests to get a reasonably good idea of which prior datasets are most similar to the new one.

Moreover, we can use these same performance results to establish which (yet untested) algorithms are likely to perform well on the new dataset, i.e., those algorithms that outperformed or rivaled the currently best algorithm on similar datasets in the past. As such, we can intelligently select the most promising algorithms for the new dataset, run them, and then use their performance results to gain increasingly better estimates of the most similar datasets and the most promising algorithms.

Key concepts. There are two key concepts used in this work. The first one is that of the *current best candidate algorithm* which may be challenged by other algorithms in the process of finding an even better candidate.

The second is the pairwise performance difference between two algorithms running on the same dataset, which we call *relative landmark*. A collection of such relative landmarks represents a history of previous 'duels' between two algorithms on prior datasets. The term itself originates from the study of landmarking algorithms: since absolute values for the performance of landmarkers vary a lot depending on the dataset, several types of *relative* landmarks have been proposed, which basically capture the relative performance difference between two algorithms [12]. In this paper, we extend the notion of relative landmarks to *all* (including non-simplified) classification algorithms.

The history of previous algorithm duels is used to select the most promising challenger for the current best candidate algorithm, namely the method that most convincingly outperformed or rivaled the current champion on prior datasets *similar* to the new dataset.

Approach. Given the current best algorithm and a history of relative landmarks (duels), we can start a tournament game in which, in each round, the current best algorithm is compared to the next, most promising contender. We select the most promising challenger as discussed above, and run a CV test with this algorithm. The winner becomes the new current best candidate, the loser is removed from consideration. We will discuss the exact procedure in Section 3.

We call this approach *active testing* (AT)[5], as it actively selects the most interesting CV tests instead of passively performing them one by one: in each iteration the *best competitor* is identified, which determines a new CV test to be carried out. Moreover, the same result will be used to further characterize the new dataset and more accurately estimate the similarity between the new dataset and all prior datasets.

Evaluation. By intelligently selecting the most promising algorithms to test on the new dataset, we can more quickly discover an algorithm that performs very well. Note that running a selection of algorithms is typically done anyway to find a suitable algorithm. Here, we optimize and automate this process using historical performance results of the candidate algorithms on prior datasets.

While we cannot possibly guarantee to return the absolute best algorithm without performing all possible CV tests, we can return an algorithm whose performance is either identical or very close to the truly best one. The difference between the two can be expressed in terms of a *loss*. Our aim is thus to *minimize* this loss using a *minimal number of tests*, and we will evaluate our technique as such.

In all, the research hypothesis that we intend to prove in this paper is: *Relative landmarks provide useful information on the similarity of datasets and can be used to efficiently predict the most promising algorithms to test on new datasets.*

We will test this hypothesis by running our active testing approach in a leave-one-out fashion on a large set of CV evaluations testing 292 algorithms on 76 datasets. The results show that our AT approach is indeed effective in finding very accurate algorithms with a limited number of tests.

We also present an adaptation of method AT that uses data characteristics to define the similarity of the datasets. Our purpose was to compare the relative landmark versus data measures approaches to select classification algorithms.

Roadmap. The remainder of this paper is organized as follows. First, we formulate the concepts of relative landmarks in Section 2 and active testing in Section 3. Next, Section 4 presents the empirical evaluation and Section 5 presents an overview of some work in other related areas. The final section presents conclusions and future work.

2 Relative Landmarks

In this section we formalize our definition of relative landmarks, and explain how are used to identify the most promising competitor for a currently best algorithm.

Given a set of classification algorithms and some new classification dataset d_{new}, the aim is to identify the potentially best algorithm for this task with respect to some given performance measure M (e.g., accuracy, AUC or rank). Let us represent the performance of algorithm a_i on dataset d_{new} as $M(a_i, d_{new})$. As such, we need to identify an algorithm $a*$, for which the performance measure is maximal, or $\forall a_i M(a*, d_{new}) \geq M(a_i, d_{new})$. The decision concerning $\geq$ (i.e. whether $a*$ is at least as good as a_i) may be established using either a statistical significance test or a simple comparison.

However, instead of searching exhaustively for $a*$, we aim to find a near-optimal algorithm, $\hat{a*}$, which has a high probability $P(M(\hat{a*}, d_{new}) \geq M(a_i, d_{new}))$ to be optimal, ideally close to 1.

As in other work that exploits metalearning, we assume that $\hat{a*}$ is likely better than a_i on dataset d_{new} if it was found to be better on a similar dataset d_j (for which we have performance estimates):

$$P(M(\hat{a*}, d_{new}) \geq M(a_i, d_{new})) \sim P(M(\hat{a*}, d_j) \geq M(a_i, d_j)) \quad (1)$$

The latter estimate can be maximized by going through all algorithms and identifying the algorithm $\hat{a*}$ that satisfies the $\geq$ constraint in a maximum number of cases. However, this requires that we know which datasets d_j are most similar to d_{new}. Since our definition of similarity requires CV tests to be run on d_{new}, but we cannot run all possible CV tests, we use an iterative approach in which we repeat this scan for $\hat{a*}$ in every round, using only the datasets d_j that seem most similar at that point, as dataset similarities are recalculated after every CV test.

Initially, having no information, we deem all datasets to be similar to d_{new}, so that $\hat{a*}$ will be the globally best algorithm over all prior datasets. We then call this algorithm the *current best algorithm* a_{best} and run a CV test to calculate its performance on d_{new}. Based on this, the dataset similarities are recalculated (see Section 3), yielding a possibly different set of datasets d_j. The best algorithm on this new

[5] Note that while the term 'active testing' is also used in the context of actively selected sampling landmarks [14], there is little or no relationship to the approach described here.

set becomes the *best competitor* a_k (different from a_{best}), calculated by counting the number of times that $M(a_k, d_j) > M(a_{best}, d_j)$, over all datasets d_j.

We can further refine this method by taking into account how large the performance differences are: the larger a difference was in the past, the higher chances are to obtain a large gain on the new dataset. This leads to the notion of relative landmarks RL, defined as:

$$RL(a_k, a_{best}, d_j) = i(M(a_k, d_j) > M(a_{best}, d_j)) * \\ *(M(a_k, d_j) - M(a_{best}, d_j)) \quad (2)$$

The function $i(test)$ returns 1 if the $test$ is true and 0 otherwise. As stated before, this can be a simple comparison or a statistical significance test that only returns 1 if a_k performs significantly better than a_{best} on d_j. The term RL thus expresses how much better a_k is, relative to a_{best}, on a dataset d_j. Experimental tests have shown that this approach yields much better results than simply counting the number of wins.

Up to now, we are assuming a dataset d_j to be either similar to d_{new} or not. A second refinement is to use a gradual (non-binary) measure of similarity $Sim(d_{new}, d_j)$ between datasets d_{new} and d_j. As such, we can weigh the performance difference between a_k and a_{best} on d_j by how similar d_j is to d_{new}. Indeed, the more similar the datasets, the more informative the performance difference is. As such, we aim to optimize the following criterion:

$$a_k = \arg\max_{a_i} \sum_{dj \in D} RL(a_i, a_{best}, d_j) * Sim(d_{new}, d_j)) \quad (3)$$

in which D is the set of all prior datasets d_j.

To calculate the similarity $Sim()$, we use the outcome of each CV test on d_{new} and compare it to the outcomes on d_j.

In each iteration, with each CV test, we obtain a new evaluation $M(a_i, d_{new})$, which is used to recalculate all similarities $Sim(d_{new}, d_j)$. In fact, we will compare four variants of $Sim()$, which are discussed in the next section. With this, we can recalculate equation 3 to find the next best competitor a_k.

3 Active Testing

In this section we describe the active testing (AT) approach, which proceeds according to the following steps:

1. Construct a global ranking of a given set of algorithms using performance information from past experiments (metadata).
2. Initiate the iterative process by assigning the top-ranked algorithm as a_{best} and obtain the performance of this algorithm on d_{new} using a CV test.
3. Find the most promising competitor a_k for a_{best} using relative landmarks and all previous CV tests on d_{new}.
4. Obtain the performance of a_k on d_{new} using a CV test and compare with a_{best}. Use the winner as the current best algorithm, and eliminate the losing algorithm.
5. Repeat the whole process starting with step 3 until a stopping criterium has been reached. Finally, output the current a_{best} as the overall winner.

Step 1 - Establish a Global Ranking of Algorithms. Before having run any CV tests, we have no information on the new dataset d_{new} to define which prior datasets are similar to it. As such, we naively assume that all prior datasets are similar. As such, we generate a global ranking of all algorithms using the performance results of all algorithms on all previous datasets, and choose the top-ranked algorithm as our initial candidate a_{best}. To illustrate this, we use a toy example involving 6 classification algorithms, with default parameter settings, from Weka [10] evaluated on 40 UCI datasets [1], a portion of which is shown in Table 1.

As said before, our approach is entirely independent from the exact evaluation measure used: the most appropriate measure can be chosen by the user in function of the specific data domain. In this example, we use *success rate (accuracy)*, but any other suitable measure of classifier performance, e.g. *AUC* (area under the ROC curve), precision, recall or F1 can be used as well.

Each accuracy figure shown in Table 1 represents the mean of 10 values obtained in 10-fold cross-validation. The ranks of the algorithms on each dataset are shown in parentheses next to the accuracy value. For instance, if we consider dataset $abalone$, algorithm MLP is attributed rank 1 as its accuracy is highest on this problem. The second rank is occupied by $LogD$, etc.

The last row in the table shows the *mean rank* of each algorithm, obtained by averaging over the ranks of each dataset: $R_{a_i} = \frac{1}{n}\sum_{d_j=1}^{n} R_{a_i,d_j}$, where R_{a_i,d_j} represents the rank of algorithm a_i on dataset d_j and n the number of datasets. This is a quite common procedure, often used in machine learning to assess how a particular algorithm compares to others [5].

The mean ranks permit us to obtain a global ranking of candidate algorithms, CA. In our case, $CA = \langle MLP, J48, JRip, LogD, IB1, NB\rangle$. It must be noted that such a ranking is not very informative in itself. For instance, statistical significance tests are needed to obtain a truthful ranking. Here, we only use this global ranking CA are a starting point for the iterative procedure, as explained next.

Step 2 - Identify the Current Best Algorithm. Indeed, global ranking CA permits us to identify the top-ranked algorithm as our initial best candidate algorithm a_{best}. In Table 1, $a_{best} = MLP$. This algorithm is then evaluated using a CV test to establish its performance on the new dataset d_{new}.

Step 3 - Identify the Most Promising Competitor. In the next step we identify a_k, the *best competitor* of a_{best}. To do this, all algorithms are considered one by one, except for a_{best} and the eliminated algorithms (see step 4).

For each algorithm we analyze the information of past experiments (meta-data) to calculate the relative landmarks, as outlined in the previous section. As equation 3 shows, for each a_k, we sum up all relative landmarks involving a_{best}, weighted by a measure of similarity between dataset d_j and the new dataset d_{new}. The algorithm a_k that achieves the highest value is the most promising competitor in this iteration. In case of a tie, the competitor that appears first in ranking CA is chosen.

To calculate $Sim(d_{new}, d_j)$, the similarity between d_j and d_{new}, we have explored six different variants, AT0, AT1, ATWs, ATx, ATdc, ATWs_k described below.

AT0 is a base-line method which ignores dataset similarity. It always returns a similarity value of 1 and so all datasets are considered similar. This means that the best competitor is determined by summing up the values of the relative landmarks.

AT1 method works as AT0 at the beginning, when no tests have been carried out on d_{new}. In all subsequent iterations, this

Table 1. Accuracies and ranks (in parentheses) of the algorithms 1-nearest neighbor (IB1), C4.5 (J48), RIPPER (JRip), LogisticDiscriminant (LogD), MultiLayerPerceptron (MLP) and naive Bayes (NB) on different datasets and their mean rank.

Datasets	IB1	J48	JRip	LogD	MLP	NB
abalone	.197 (5)	.218 (4)	.185 (6)	.259 (2)	.266 (1)	.237 (3)
acetylation	.844 (1)	.831 (2)	.829 (3)	.745 (5)	.609 (6)	.822 (4)
adult	.794 (6)	.861 (1)	.843 (3)	.850 (2)	.830 (5)	.834 (4)
...	...	...	...	...	...	...
Mean rank	4.05	2.73	3.17	3.74	2.54	4.78

method estimates dataset similarity using only the most recent CV test. Consider the algorithms listed in Table 1 and the ranking CA. Suppose we started with algorithm MLP as the current best candidate. Suppose also that in the next iteration $LogD$ was identified as the best competitor, and won from MLP in the CV test: $(M(LogD, d_{new}) > M(MLP, d_{new}))$. Then, in the subsequent iteration, all prior datasets d_j satisfying the condition $M(LogD, d_j) > M(MLP, d_j)$ are considered similar to d_{new}. In general terms, suppose that the last test revealed that $M(a_k, d_{new}) > M(a_{best}, d_{new})$, then $Sim(d_{new}, d_j)$ is 1 if also $M(a_k, d_j) > M(a_{best}, d_j)$, and 0 otherwise. The similarity measure determines which RL's are taken into account when summing up their contributions to identify the next best competitor.

Another variant of AT1 could use the difference between $RL(a_k, a_{best}, d_{new})$ and $RL(a_k, a_{best}, d_j)$, normalized between 0 and 1, to obtain a real-valued (non-binary) similarity estimate $Sim(d_{new}, d_j)$. In other words, d_j is *more similar* to d_{new} if the relative performance difference between a_k and a_{best} is about as large on both d_j and d_{new}. We plan to investigate this in our future work.

ATWs is similar to AT1, but instead of only using the last test, it uses *all* CV tests carried out on the new dataset, and calculates the Laplace-corrected ratio of corresponding results. For instance, suppose we have conducted 3 tests on d_{new}, thus yielding 3 pairwise algorithm comparisons on d_{new}. Suppose that 2 tests had the same result on dataset d_j (i.e. $M(a_x, d_{new}) > M(a_y, d_{new})$ and $M(a_x, d_j) > M(a_y, d_j)$), then the frequency of occurrence is $2/3$, which is adjusted by Laplace correction to obtain an estimate of probability $(2 + 1)/(3 + 2)$. As such, $Sim(d_{new}, d_j) = \frac{3}{5}$.

ATx is similar to ATWs, but requires that all pairwise comparisons yield the same outcome. In the example used above, it will return $Sim(d_{new}, d_j) = 1$ only if all three comparisons lead to same result on both datasets and 0 otherwise.

ATdc is similar to ATWs, but uses a different similarity function. The idea of using this variant was to test if data characteristics (e.g. *number of examples*) could provide better information to identify the most similar datasets (w.r.t. d_{new}). We define the similarity between two datasets d_{new} and d_j using the following expression:

$$Sim(d_{new}, d_j) = 1 - \frac{1}{|Z|} \sum_{z \in Z} \frac{|z(d_{new}) - z(d_j)|}{max(z) - min(z)} \qquad (4)$$

The symbol z represents a generic data measure used to characterize the datasets ($z(d)$ is the value of characteristic z for dataset d). Z is the set of measures used.

ATWs_k is similar to ATWS, but only consider the k most similar datasets (those with highest Sim values). The similarity for all the other datasets are set to 0. The idea is to avoid the situation where the similarity values show very small variations, making unusefull the information in the relative landmarks.

Step 4 - Determine which of the Two Algorithms is Better. Having found a_k, we can now run a CV test and compare it with a_{best}. The winner (which may be either the current best algorithm or the competitor) is used as the new current best algorithm in the new round. The losing algorithm is eliminated from further consideration.

Step 5 - Repeat the Process and Check the Stopping Criteria. The whole process of identifying the best competitor (step 3) of a_{best} and determining which one of the two is better (step 4) is repeated until a stopping criterium has been reached. For instance, the process could be constrained to a fixed number of CV tests: considering the results presented further on in Section 4, it would be sufficient to run at most 20% of all possible CV tests. Alternatively, one could impose a fixed CPU time, thus returning the best algorithm in h hours, as in budgeted learning. In any case, until aborted, the method will keep choosing a new competitor in each round: there will always be a next best competitor. In this respect our system differs from, for instance, hill climbing approaches which can get stuck in a local minimum.

Discussion - Comparison with Active Learning: The term *active testing* was chosen because the approach shares some similarities with *active learning* [7]. The concern of both is to speed up the process of improving a given performance measure. In active learning, the goal is to select the most informative data point to be labeled next, so as to improve the predictive performance of a supervised learning algorithm with a minimum of (expensive) labelings. In active testing, the goal is to select the most informative CV test, so as to improve the prediction of the best algorithm on the new dataset with a minimum of (expensive) CV tests.

4 Empirical Evaluation

4.1 Evaluation Methodology and Experimental Set-up

The proposed method AT was evaluated using a *leave-one-out* method [18]. The experiments reported here involve D datasets and so the whole procedure was repeated D times. In each cycle, all performance results on one dataset were left out for testing and the results on the remaining $D - 1$ datasets were used as metadata to determine the best candidate algorithm.

This study involved 292 algorithms (algorithm-parameter combinations), which were extracted from the experiment database for machine learning (ExpDB) [11, 22]. This set includes many different algorithms from the Weka platform [10], which were varied by assigning different values to their most important parameters. It includes SMO (a support vector machine, SVM), MLP (Multilayer Perceptron), J48 (C4.5), and different types of ensembles, including RandomForest, Bagging and Boosting. Moreover, different

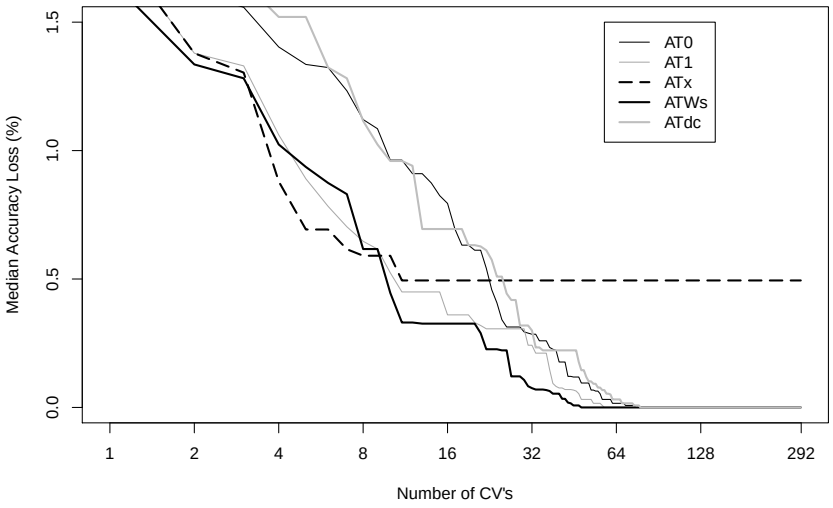

Figure 1. Median loss as a function of the number of CV tests.

SVM kernels were used with their own parameter ranges and all non-ensemble learners were used as base-learners for the ensemble learners mentioned above. The 76 datasets used in this study were all from UCI [1]. A complete overview of the data used in this study, including links to all algorithms and datasets can be found on `http://expdb.cs.kuleuven.be/ref/blv11`.

The data measures used to characterize the datasets for variant ATdc were *number of examples*, *proportion of nominal attributes*, *proportion of missing values*, *class entropy* and *mean mutual information*.

Regarding ATWs_k we set $k = 20$ (this value was set by exploring only the alternatives 30, 20 and 10).

The main aim of the test was to prove the research hypothesis formulated earlier: relative landmarks provide useful information for predicting the most promising algorithms on new datasets. Therefore, we also include two baseline methods:

TopN has been described before (e.g. [3]). It also builds a ranking of candidate algorithms as described in step 1 (although other measures different from mean rank could be used), and only runs CV tests on the first N algorithms. The overall winner is returned.

Rand simply selects N algorithms at random from the given set, evaluates them using CV and returns the one with the best performance. It is repeated 10 times with different random seeds and the results are averaged.

Since our AT methods are iterative, we will restart TopN and Rand N times, with N equal to the number of iterations (or CV tests).

To evaluate the performance of all approaches, we calculate the *loss* of the currently best algorithm, defined as $M(a_{best}, d_{new}) - M(a*, d_{new})$, where a_{best} represents the currently best algorithm, $a*$ the best possible algorithm and $M(.)$ represents the performance measure (success rate).

4.2 Results

By aggregating the results over D datasets, we can track the *median loss* of the recommended algorithm as a function of the number of CV tests carried out. The results are shown in Figure 1. Note that the number of CV tests is plotted on a logarithmic scale.

First, we see that ATWs and AT1 perform much better than AT0, which indicates that it is indeed useful to include dataset similarity. If we consider a particular level of loss (say 0.5%) we note that these variants require much fewer CV tests than AT0. The results also indicate that the information associated with relative landmarks obtained on the new dataset is indeed valuable. The median loss curves decline quite rapidly and are always below the AT0 curve. We also see that after only 10 CV tests (representing about 3% of all possible tests), the median loss is less than 0.5%. If we continue to 60 tests (about 20% of all possible tests) the median loss is near 0.

Also note that ATWs, which uses all relative landmarks involving a_{best} and d_{new}, does not perform much better than AT1, which only uses the most recent CV test. However in Figure 2 we show the performance of variant ATWs_k is much better than ATWs.

Method ATx, the most restrictive approach, only considers prior datasets on which *all* relative landmarks including a_{best} obtained similar results. As shown in Figure 1, this approach manages to reduce the loss quite rapidly, and competes well with the other variants in the initial region. However, after achieving a minimum loss in the order of 0.5%, there are no more datasets that fulfill this restriction, and consequently no new competitor can be chosen, causing it to stop. The other three methods, ATWs, ATdc and AT1, do not suffer from this shortcoming. We can see that the variant that uses data characteristics (ATdc) is generally worse than the other variants. Before the 23^{rd} CV test all variants AT1, ATx and ATWs need about half the CV tests needed by ATdc.

AT0 was also our best baseline method. To avoid overloading Figure 1, we show this separately in Figure 2. Indeed, AT0 is clearly better than the random choice method $Rand$. Comparing AT0 to

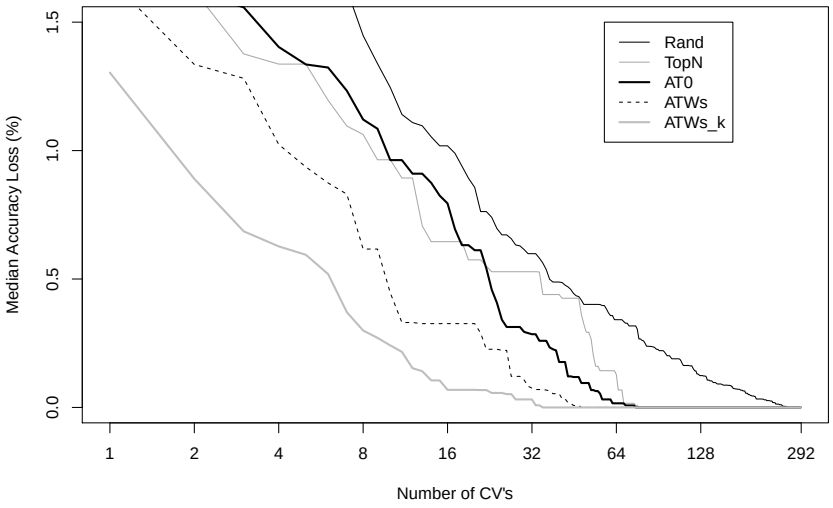

Figure 2. Median loss of AT0, ATWs, ATWs_k and the two baseline methods.

TopN, we cannot say that one is clearly better than the other overall, as the curves cross. However, it is clear that TopN looses out if we allow more CV tests, and that it is not competitive with the more advanced methods such as AT1, ATWs and ATWs_k. Is clear in Figure 2 that the best method variant is ATWs_k (using $k = 20$). We can see clearly in the curves that ATWs_k typically needs about half the CV tests needed by ATWs. The comparison with all the other variants is even more impressive.

The curves for mean loss (instead of median loss) follow similar trends, but the values are 1-2% worse due to outliers (see Fig. 3 relative to method ATWs_k). Besides, this figure shows also the curves associated with quartiles of 25% and 75% for ATWs_k. As the number of CV tests increases, the distance between the two curves decreases and approaches the median curve. Similar behavior has been observed for ATWs and AT1 but we skip the curves in this text.

Algorithm trace. It is interesting to trace the iterations carried out for one particular dataset. Table 2 shows the details for method AT1, where $abalone$ represents the new dataset. Column 1 shows the number of the iteration (thus the number of CV tests). Column 2 shows the most promising competitor a_k chosen in each step. Column 3 shows the index of a_k in our initial ranking CA, and column 4 the index of a_{best}, the *new best* algorithm after the CV test has been performed. As such, if the values in column 3 and 4 are the same, then the most promising competitor has won the duel. For instance, in step 2, $SMO.C.1.0.Polynomial.E.3$, i.e. SVM with complexity constant set to 1.0 and a 3rd degree polynomial kernel, (index 96) has been identified as the best competitor to be used (column 2), and after the CV test, it has won against $Bagging.I.75..100.PART$, i.e. Bagging with a high number of iterations (between 75 and 100) and PART as a base-learner. As such, it wins this round and becomes the new a_{best}. Columns 5 and 6 show the *actual* rank of the competitor and the winner on the $abalone$ dataset. Column 7 shows the loss compared to the optimal algorithm and the final column shows the number of datasets whose similarity measure is 1.

We observe that after only 12 CV tests, the method has identified an algorithm with a very small loss of 0.2%: $Bagging.I.25..50.MultilayerPerceptron$, i.e. Bagging with relatively few iterations but with a MultiLayerPerceptron base-learner.

Incidentally, this dataset appears to represent a quite atypical problem: the truly best algorithm, $SMO.C.1.0.RBF.G.20$, i.e. SVM with an RBF kernel with kernel width (gamma) set to 20, is ranked globally as algorithm 246 (of all 292). AT1 identifies it after 177 CV tests.

5 Related Work in Other Scientific Areas

In this section we briefly cover some work in other scientific areas which is related to the problem tackled here and could provide further insight into how to improve the method.

One particular area is *experiment design* [6] and in particular *active learning*. As discussed before, the method described here follows the main trends that have been outlined in this literature. However, there is relatively little work on active learning for ranking tasks. One notable exception is [15], who use the notion of *Expected Loss Optimization (ELO)*. Another work in this area is [4], whose aim was to identify the most interesting substances for drug screening using a minimum number of tests. In the experiments described, the authors have focused on the top-10 substances. Several different strategies were considered and evaluated. Our problem here is not ranking, but rather simply finding the best item (algorithm), so this work is only partially relevant.

Another relevant area is the so called *multi-armed bandit problem (MAB)* studied in statistics and machine learning [9, 16]. This problem is often formulated in a setting that involves a set of traditional slot machines. When a particular lever is pulled, a reward is provided from a distribution associated with that specific lever. The bandit problem is formally equivalent to a one-state Markov decision process. The aim is to minimize *regret* after T rounds, which is defined as a difference between the reward sum associated with an optimal strategy and the sum of collected rewards. Indeed, pulling

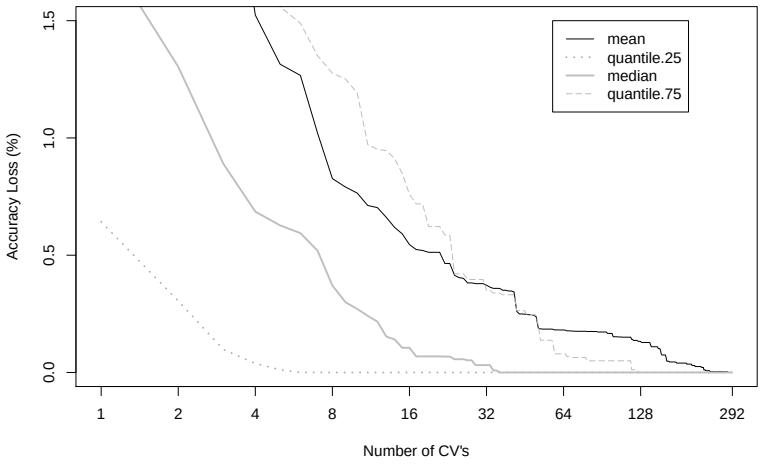

Figure 3. Loss of ATWs_k (k=20) as a function of the number of CV tests.

Table 2. Trace of the steps taken by AT1 in the search for the supposedly best algorithm for the $abalone$ dataset

CV test	Algorithm used (current best competitor, a_k)	CA a_k	CA new a_{best}	abalone a_k	abalone new a_{best}	Loss (%)	D size
1	Bagging.I.75..100.PART	1	1	75	75	1.9	75
2	SMO.C.1.0.Polynomial.E.3	96	96	56	56	1.6	29
3	AdaBoostM1.I.10.MultilayerPerceptron	92	92	47	47	1.5	34
4	Bagging.I.50..75.RandomForest	15	92	66	47	1.5	27
...	...	...	...	...	...	...	...
10	LMT	6	6	32	32	1.1	45
11	LogitBoost.I.10.DecisionStump	81	6	70	32	1.1	51
12	Bagging.I.25..50.MultilayerPerceptron	12	12	2	2	0.2	37
13	LogitBoost.I.160.DecisionStump	54	12	91	2	0.2	42
...	...	...	...	...	...	...	...
177	SMO.C.1.0.RBF.G.20	246	246	1	1	0	9

a lever can be compared to carrying out a CV test on a given algorithm. However, there is one fundamental difference between MAB and our setting: whereas in MAB the aim is to maximize the sum of collected rewards, our aim it to maximize *one* reward, i.e. the reward associated with identifying the best algorithm. So again, this work is only partially relevant.

To the best of our knowledge, no other work in this area has addressed the issue of how to select a suitable algorithm from a large set of candidates.

6 Significance and Impact

In this paper we have addressed the problem of selecting the best classification algorithm for a specific task. We have introduced a new method, called *active testing*, that exploits information concerning past evaluation results (metadata), to recommend the best algorithm using a limited number of tests on the new dataset.

Starting from an initial ranking of algorithms on previous datasets, the method runs additional CV evaluations to test several competing algorithms on the new dataset. However, the aim is to reduce the number of tests to a minimum. This is done by carefully selecting which tests should be carried out, using the information of both past and present algorithm evaluations represented in the form of relative landmarks.

In our view this method incorporates several innovative features. First, it is an iterative process that uses the information in each CV test to find the most promising next test based on a history of prior 'algorithm duels'. In a tournament-style fashion, it starts with a current best (parameterized) algorithm, selects the most promising rival algorithm in each round, evaluates it on the given problem, and eliminates the algorithm that performs worse. Second, it continually focuses on the most similar prior datasets: those where the algorithm duels had a similar outcome to those on the new dataset.

Four variants of this basic approach, differing in their definition of dataset similarity, were investigated in a very extensive experiment setup involving 292 algorithm-parameter combinations on 76 datasets. Our experimental results show that particularly versions ATWs and AT1 provide good recommendations using a small num-

ber of CV tests. When plotting the median loss as a function of the number of CV tests (Fig. 1), it shows that both outperform all other variants and baseline methods. They also outperform AT0, indicating that dataset similarity is an important aspect.

We also see that after only 10 CV tests (representing about 3% of all possible tests), the median loss is less than 0.5%. If we continue to 60 tests (about 20% of all possible tests) the median loss is near 0. Similar trends can be observed when considering mean loss.

The results support the hypothesis that we have formulated at the outset of our work, that relative landmarks are indeed informative and can be used to suggest the best contender. If this is procedure is used iteratively, it can be used to accurately recommend a classification algorithm after a very limited number of CV tests.

Still, we believe that the results could be improved further. Classical information-theoretic measures and/or sampling landmarks could be incorporated into the process of identifying the most similar datasets. This could lead to further improvements and forms part of our future plans.

ACKNOWLEDGEMENTS

This work is funded (or part-funded) by the ERDF European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness) and by National Funds through the FCT Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project FCOMP - 01-0124-FEDER-022701.

REFERENCES

[1] C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998.

[2] B.Pfahringer, H.Bensussan, and C. Giraud-Carrier, 'Meta-learning by landmarking various learning algorithms', in *Proceedings of the 17th Int. Conf. on Machine Learning (ICML-2000*, Stanford,CA, (2000).

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

[4] K. De Grave, J. Ramon, and L. De Raedt, 'Active learning for primary drug screening', in *Proceedings of Discovery Science*. Springer, (2008).

[5] J. Demsar, 'Statistical comparisons of classifiers over multiple data sets', *The Journal of Machine Learning Research*, **7**, 1–30, (2006).

[6] V. Fedorov, *Theory of Optimal Experiments*, Academic Press, New York, 1972.

[7] Y. Freund, H. Seung, E. Shamir, and N. Tishby, 'Selective sampling using the query by committee algorithm', *Machine Learning*, **28**, 133–168, (1997).

[8] Johannes Fürnkranz and Johann Petrak, 'An evaluation of landmarking variants', in *Proceedings of the ECML/PKDD Workshop on Integrating Aspects of Data Mining, Decision Support and Meta-Learning (IDDM-2001)*, pp. 57–68. Springer, (2001).

[9] J. Gittins, 'Multi-armed bandit allocation indices', in *Wiley Interscience Series in Systems and Optimization*, John Wiley & Sons, Ltd., (1989).

[10] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten, 'The WEKA data mining software: an update', *SIGKDD Explor. Newsl.*, **11**(1), 10–18, (2009).

[11] H.Blockeel, 'Experiment databases: A novel methodology for experimental research', in *Leacture Notes on Computer Science 3933*. Springer, (2006).

[12] J.Fürnkranz and J.Petrak, 'An evaluation of landmarking variants', in *Working Notes of ECML/PKDD 2000 Workshop on Integration Aspects of Data Mining, Decision Support and Meta-Learning*, eds., C.Carrier, N.Lavrac, and S.Moyle, (2001).

[13] R. Leite and P. Brazdil, 'Predicting relative performance of classifiers from samples', in *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pp. 497–503, New York, NY, USA, (2005). ACM Press.

[14] Rui Leite and Pavel Brazdil, 'Active testing strategy to predict the best classification algorithm via sampling and metalearning', in *Proceedings of the 19th European Conference on Artificial Intelligence - ECAI 2010*, (2010).

[15] B. Long, O. Chapelle, Y. Zhang, Y. Chang, Z. Zheng, and B. Tseng, 'Active learning for rankings through expected loss optimization', in *Proceedings of the SIGIR'10*. ACM, (2010).

[16] A. Mahajan and D. Teneketzis, 'Multi-armed bandit problems', in *Foundations and Applications of Sensor Management*, eds., D. A. Castanon, D. Cochran, and K. Kastella, Springer-Verlag, (2007).

[17] D. Michie, D.J.Spiegelhalter, and C.C.Taylor, *Machine Learning, Neural and Statistical Classification*, Ellis Horwood, 1994.

[18] Tom M. Mitchell, *Machine Learning*, McGraw-Hill, New York, 1997.

[19] John R. Rice, 'The algorithm selection problem', volume 15 of *Advances in Computers*, 65 – 118, Elsevier, (1976).

[20] Kate A. Smith-Miles, 'Cross-disciplinary perspectives on meta-learning for algorithm selection', *ACM Comput. Surv.*, **41**(1), 1–25, (2008).

[21] Carlos Soares, Johann Petrak, and Pavel Brazdil, 'Sampling-based relative landmarks: Systematically test-driving algorithms before choosing', in *Proceedings of the 10th Portuguese Conference on Artificial Intelligence (EPIA 2001)*, pp. 88–94. Springer, (2001).

[22] J. Vanschoren and H. Blockeel, 'A community-based platform for machine learning experimentation', in *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2009*, volume LNCS 5782, pp. 750–754. Springer, (2009).

[23] Ricardo Vilalta and Youssef Drissi, 'A perspective view and survey of meta-learning', *Artif. Intell. Rev.*, **18**(2), 77–95, (2002).