An Evaluation of Landmarking Variants

Johannes Fürnkranz and Johann Petrak

Austrian Research Institute for Artificial Intelligence
Schottengasse 3, A-1010 Wien, Austria
[juffi,johann]@ai.univie.ac.at

Abstract. Landmarking is a novel technique for data characterization in meta-learning. While conventional approaches typically describe a database with its statistical measurements and properties, landmarking proposes to enrich such a description with quick and easy-to-obtain performance measures of simple learning algorithms. In this paper, we will discuss two novel aspects of landmarking. First, we investigate relative landmarking, which tries to exploit the relative order of the landmark measures instead of their absolute value. Second, we propose to use subsampling estimates as a different way for efficiently obtaining landmarks. In general, our results are mostly negative. The most interesting result is a surprisingly simple rule that predicts quite accurately when it is worth to boost decision trees.

1 Introduction

Recently, the idea of *landmarking* as a new technique for data characterization emerged from the METAL project [2, 12]. The basic idea of landmarking is simple and elegant: use the performance of a few fast learning algorithms to characterize the dataset. Typical choices for landmarking algorithms are high bias learners like Naive Bayes classifiers, linear discriminants, or decision stumps.

In this paper, we investigate two variants of the landmarking idea. The first idea—*relative landmarking*—is to replace or enrich the use of absolute values of performance characteristics (typically one uses the accuracy of the landmarking algorithms) with the use of relations between these values. We consider pairwise comparisons, accuracy ratios, orders, and rankings. In the second part of the paper, we propose the use of *subsample landmarkers*, i.e., landmarks that originate from performance estimates of the candidate algorithms on subsets of the data. We evaluate both approaches on pairwise algorithm selection tasks.

2 Data Characterization via Landmarking

As meta-learning is concerned with the learning of mappings from machine learning tasks to machine learning tools, the problem is often formulated as a learning task where the objects are datasets and the class values are candidate algorithms to be used on these datasets. The first problem that has to be solved in such a framework is the definition of a set of descriptors that can be used to describe a dataset in a way that can be used by the meta-learner. This step is called *data characterization*.

Typically, datasets are characterized by measures that can be computed directly from a dataset, ranging from simple things such as the number of (symbolic and numeric) attributes, classes, and instances, to statistical correlation measures and information-theoretic entropy measures. The current version of the METAL Data Characterization Tool DCT [10] computes about 50 base measurements for each datasets, and a variable number of additional measurements for each attribute or each pair of attributes. While the former are quite straight-forward the use, the variable number of the per-attribute measures forces one to use summary measures for the latter. Alternative approaches try to capture the distribution of these values by computing histograms [9] or the use of more expressive learning algorithms [15].

Recently, it was proposed that the use of learning algorithms may provide valuable additional information for this part of the meta-learning problem. *Landmarking* is one such proposal [2, 12], which tries to characterize a dataset by the performance measures of simple learning algorithms. Other proposals focus on using properties of the concepts that are learned with certain algorithms [1] or even the concepts themselves [3].

Landmarking tries to model the practitioner who familiarizes herself with a new problem by first trying a few fast and familiar learning algorithms. Consequently, the basic idea of landmarking is to model the performance of complex and computationally expensive algorithms using performance measures from simple and computationally fast algorithms, the *landmarkers*. Such an approach is based on the assumption that different learning algorithms have different *regions of expertise* in the data space, and that the regions of expertise for complex algorithms can be described by identifying regions of expertise of simpler algorithms. For example, an intuitive rule, such as *"If a decision stump performs well on your data, try growing a decision tree"*, could be learned from a meta-dataset in which several datasets have been annotated with landmarkers, one of them being a decision stump algorithm.

In [12], two important criteria for choosing appropriate landmarkers were proposed:

Efficiency: A good landmarker should be fairly cheap to compute. If expensive computations are required to obtain the landmark, these might be better invested for directly testing the candidate algorithms on the dataset.[1] It makes sense to use one-level decision stumps as landmarkers for predicting whether one should use C50boost, but not the other way around.

Bias Diversity: A good set of landmarkers should consist of landmarkers with different biases. If two landmarkers have very similar performance measures on all data sets, it would probably suffice to include only one of them.

Based on these criteria, typical choices of landmarkers were simple, computationally efficient algorithms with a high bias. For example, [2] chose one-level decision stumps (with the best, the worst, and a random node), the Naïve Bayes classifier, two versions of nearest neighbour algorithms (one of them operating only on a subset of the available features), a linear discriminant classifier, and the default predictor.

[1] This, of course, holds for all data characterisation techniques. For example, one statistical test originally proposed for DCT had a complexity of $O(n^3)$. This turned out to be too expensive and was removed from consideration. [12] suggest to limit the complexity of landmarkers to $O(n \log(n))$.

Table 1. A simple example where conventional landmarking might fail.

	Landmarker 1	Landmarker 2	Recommendation
Dataset 1	0.9	0.8	use Algorithm A
Dataset 2	0.8	0.9	use Algorithm B
Dataset 3	0.2	0.1	use Algorithm A
Dataset 4	0.1	0.2	use Algorithm B

Note that such algorithms often implicitly measure important properties of a dataset: for example, the accuracy of a linear discriminator may be interpreted as a measure of the linear separability of a dataset; the accuracy of a Naive Bayes classifier may be interpreted as a measure of the independence of the attributes; and the accuracy of a nearest-neighbor classifier may be interpreted as a measure of how clustered the classes of the problem are [2].

3 Relative Landmarking

Crucial to the validity of landmarking is how to identify the region of expertise of an algorithm. The conventional approach is to simply include estimates for the predictive accuracy of the landmarkers on a dataset. The research reported in this paper was motivated by the belief that this approach has its limitations, which we will discuss in the following.

A conventional, propositional meta-learning algorithm is only able to look at each attribute in isolation. Therefore it is only able to capture whether the performance of the landmarker is high or low (compared to the performance of the same landmarker on the other data sets), but not whether the algorithm performs well on this particular dataset. In other words, the differences in absolute size of their values depend not so much on the performance differences between the landmarkers on the *same* problem but on the performance differences between the landmarkers on *different* problems. Hence, the obtained landmarks are likely to differ much more between the two datasets than among each other.

This suggests that it might not be ideal to characterize an algorithm's region of expertise by the absolute value of its predictive accuracy, but by its relation to the performance of other, competing algorithms. This is illustrated in Table 1. It shows a hypothetical meta-database with landmarkers, in which the task is to predict whether algorithm A or algorithm B should be used on a given dataset. The four datasets are characterized by the performance of two landmarkers on each of these datasets. The first two datasets represent domains in which both landmarkers achieve fairly good results, while on datasets 3 and 4 both algorithms are in the lower performance regions. Neither Landmarker 1 nor Landmarker 2 is well-suited for predicting which algorithm—A or B—should be used for the prediction task or, in other words, for identifying the regions of expertise of algorithms A and B. Nevertheless, a closer inspection of table 1 reveals that, in fact, the regions of expertise of algorithms A and B are very well correlated with the regions of expertise of the landmarkers: Algorithm A should be used

if Landmarker 1 performs better than Landmarker 2, and algorithm B should be used otherwise.

Hence, the technique we propose—*relative landmarking*—provides the meta-learner not with the absolute performance measures of the landmarkers, but with relations that capture the landmarkers' performance relative to each other. In the simplest case, such relations could be inequalities between each pair of landmarkers, but more complicated relations, like inequalities involving significant tests or rankings of the landmarkers are possible.

In particular, we will investigate the following five landmarking representations:

Absolute (LM): This strategy is the conventional landmarking technique, where the accuracy estimates for the landmark algorithms are directly used.

Ranks (RK): Like with conventional landmarking, there is one attribute corresponding to each landmarker. However, this attribute does not encode the accuracy estimate obtained for the landmarker, but its performance rank among its competitors (i.e., a number from ≥ 1, where 1 means it was best, 2 means it was 2nd-best, etc.). In the case of ties, all tied values receive the same value (the best rank, i.e., the lowest number). The ranks are used as continuous features and not as symbolic values.

Order (OR): This encoding tries to capture the order of the landmarkers: there is one attribute for each possible rank, and its value is the landmarker that obtained that rank. E.g., the attribute that corresponds to the best/worst landmarker will have the name of the best/worst landmarker on this dataset as its value. In the case of ties (multiple algorithms achieve the same rank) these are broken randomly, so that in each case, exactly one algorithm is assigned to exactly one of the order attributes.

Pairwise (RL): This strategy enables the learner at the meta-level to make use of pairwise comparisons between the accuracies of the landmarkers. For each pairs of landmarkers, these relations returned $+1$ if the first value was larger, -1 if the second value was larger, and the missing value `?` otherwise.

Ratios (RT): Here, we encode the pairwise accuracies ratios for all pairs of landmarkers. This representation may be seen as a generalization of the previous one, where the symbolic, binary attribute with the values $-1, +1$ is represented as a continuous range. We use ratios instead of differences because we wanted to have differences relative to the order of magnitude of the error, instead of absolute values.[2]

Naturally, these different types of encoding relational knowledge may be used in combination with each other. In the following, we describe an experimental comparison between these five groups of landmarking estimates and all their combinations.

In our experiments, we used the following landmarkers, which closely follow the suggestions made in [2, 12]:

[2] The fact that the resulting value range is $(0, \infty)$ is assymetric around 1, the value of equal performance, should not make a difference to algorithms like C4.5 or Ripper, which treat continuous attributes by dynamically discretizing them, and which we will use as meta-algorithms. It might, however, make a difference if numerical algorithms are used (e.g., linear regression).

– Average Node: the average accuracy of all one node decision stumps.[3] This may be interpreted as the expected value of the accuracy of a random node, which is used in the original proposal.
– Best Node: a decision stump using the best attribute according to information gain ratio
– Worst Node: a decision stump using the worst attribute according to information gain ratio
– Linear Discriminant: a linear discriminant function
– Naive Bayes: a Naive Bayes classifier

The main difference to the landmarkers used in [2, 12] is that we use training accuracy of the landmarkers instead of a 10-fold cross-validation. The reason for this is that on the one hand landmarkers should be cheap: the time spent in performing a 10-fold cross-validation of five landmarkers would probably be better invested in computing an evaluation of one of the candidate algorithms. For this reason, we also eliminated the 1NN and Elite-1NN landmarkers, whose accuracy must be estimated with some form of cross-validation (training accuracy is always 100%). All of the remaining five learners have a high bias, so that the training accuracy should not differ much from the cross-validation estimate of the test accuracy.

The datasets used were all sets used within the METAL project that have 1000 or more examples (48 in total). We used Ripper [5] and C4.5 [13] as the meta-learning algorithms. Evaluation on the meta-datasets was via a leave-one-out cross-validation. As the meta-level task, we selected the simplest setting, namely binary algorithm selection, i.e., to predict which of a pair of algorithms will be more accurate. The training set for this task is constructed by estimating the predictive accuracy of the base algorithms via a 10-fold cross-validation, and picking the better algorithm as the class label. In case both algorithms were equal on a training set, we simply ignored this case for the training phase, so that the individual datasets had slightly fewer examples. The reasoning behind this is that on datasets with equal performance, it does not matter which algorithm we predict.[4]

As the base level algorithms we used Ripper [5], Ltree [7], C50tree, C50rules, and C50boost (commercial successors of C4.5 [13]). We constructed one meta data set for each of the 10 pairs of these 5 algorithms, and represented it in $2^5 - 1 = 31$ different ways, one for each combination of one or more of the five landmarking techniques. On each of the resulting 310 meta-datasets, we evaluated C4.5 and Ripper using a leave-one-out cross-validation.

The results are summarized in Table 2. We report the average error rates from the leave-one-out cross-validation over the 10 datasets and an adjusted error rate that normalizes the size of each meta-dataset to 48 by assuming that the predictions which were left out in the training set were correctly classified (both base-level algorithms

[3] For numerical attributes we also build only one level, i.e., only one binary split, which maximizes information gain ratio.

[4] This procedure could probably be improved by only considering algorithm performances different if their difference is above a certain level (determined, e.g., by a significance test). This would presumably make the prediction task easier, but the evaluation harder because of the loss in training examples.

Table 2. Error rates (*Error*, the error estimate resulting from the loo x-val), adjusted error rates (*Adjusted*, the error rated normalized to all 48 examples, where in the case of tied best algorithms all predictions were counted correct), and ratio to the default error rate (*DefRatio*) for all combinations of the five landmarking techniques. The results are averages over leave-one-out cross-validations (geometric averages in the case of the ratios) on 10 meta-datasets with up to 48 examples, one meta-dataset for each pairwise algorithm selection problem for the base algorithms C50tree, C50rules, C50boost, Ltree, Ripper. Results are given for both Ripper and C4.5 at the meta-level.

	Ripper			C4.5		
	Error	*Adjusted*	*DefRatio*	*Error*	*Adjusted*	*DefRatio*
Default	30.63	28.12	1.000	30.63	28.12	1.000
LM	33.97	31.25	1.106	33.74	31.04	1.075
OR	33.16	30.42	0.990	29.34	26.88	0.818
RK	38.00	35.00	1.040	37.35	34.38	1.037
RL	29.22	27.08	0.849	31.78	29.37	0.970
RT	34.32	31.67	1.055	34.79	32.08	1.059
LM OR	30.95	28.54	0.988	29.99	27.71	0.847
LM RK	30.86	28.54	0.894	33.90	31.25	0.968
LM RL	30.25	27.92	0.951	30.31	27.92	0.939
OR RK	37.02	34.17	1.027	33.98	31.25	0.950
RL OR	32.68	30.21	0.973	32.46	30.00	0.993
RL RK	33.43	31.04	0.928	33.75	31.25	0.948
OR RT	34.58	31.87	1.080	36.30	33.33	1.099
RK RT	32.35	29.79	0.972	34.12	31.46	0.978
LM RT	31.94	29.37	1.012	34.00	31.46	1.034
RL RT	32.98	30.42	1.037	37.95	35.00	1.172
LM OR RK	33.36	30.83	0.952	30.06	27.71	0.882
LM RL OR	33.61	31.04	1.073	33.85	31.25	1.030
LM RL RK	31.95	29.58	0.914	31.18	28.75	0.906
RL OR RK	33.86	31.46	0.937	34.87	32.29	0.985
LM OR RT	32.60	30.00	1.020	33.85	31.25	1.016
LM RK RT	31.90	29.37	0.953	35.69	32.92	1.019
LM RL RT	32.59	30.00	1.027	37.33	34.37	1.188
OR RK RT	33.54	30.83	1.009	37.10	34.17	1.039
RL OR RT	34.15	31.46	1.073	36.56	33.75	1.135
RL RK RT	32.56	30.00	0.975	35.64	32.92	1.011
LM RL OR RK	29.76	27.50	0.862	32.96	30.42	0.947
LM OR RK RT	32.50	30.00	0.976	37.03	34.17	1.058
LM RL OR RT	32.37	29.79	1.018	37.04	34.17	1.149
LM RL RK RT	32.12	29.58	0.965	35.23	32.50	1.016
RL OR RK RT	33.70	31.04	1.017	36.05	33.33	1.014
LM RL OR RK RT	32.11	29.58	0.964	36.54	33.75	1.038

Table 3. Default error rates and adjusted error rates (averaged over all combinations of the five landmarking techniques) for all combinations of base algorithms for both Ripper and C4.5 at the meta-level.

		Ripper		C4.5	
	Default	*Adj.Err.*	*DefRatio*	*Adj.Err.*	*DefRatio*
RIPPER LTREE	35.42	32.80	0.908	31.05	0.860
C50TREE LTREE	41.67	50.07	1.188	56.18	1.332
C50BOOST LTREE	27.08	35.55	1.298	35.48	1.295
C50RULES LTREE	35.42	40.12	1.127	45.03	1.253
C50TREE RIPPER	27.08	34.07	1.248	32.86	1.195
C50BOOST RIPPER	18.75	20.77	1.100	25.54	1.343
C50RULES RIPPER	29.17	36.96	1.262	37.43	1.266
C50BOOST C50TREE	20.83	9.27	0.383	8.40	0.325
C50RULES C50TREE	25.00	25.20	0.982	27.82	1.093
C50BOOST C50RULES	20.83	18.21	0.849	17.00	0.797

performed indistinguishably on these cases, so either prediction would be correct). The third measure shows the geometric mean of the achieved error rate over the default error on each problem (i.e., the error rate for predicting the majority class). Values > 1 show a degradation of performance, while values < 1 show an improvement. Performance ratios were used because they give a higher weights to the (absolute) performance differences of datasets with a low default error (compared to equal differences on datasets with higher default errors. This is necessary byecause for problem with different default errors, different absolute improvements are possible (e.g., for a default error of 0.5, the room for improvement is 0.5, while it is only 0.01 for a problem with default error 0.01). Given that the default errors over all problems vary considerably (see first column of Table 3, we decided to normalize for this effect and report relative improvements. Also note that the values are, of course, not symmetric around 1: the performance degradation that corresponds to an improvement rate of x is $1/x$ (i.e., 0.75 corresponds to 1.5, not 1.25).

At first sight, the obtained performance seems to be not very impressive: the average accuracies are almost always above the average default accuracies, while the relative improvement rates appear to be normally distributed around the default accuracy. A closer examination reveals a few systematic differences. For example, ratios (RT) seem to be a particularly bad feature type. The settings that include ratios in their feature sets (i.e., where RT appears somewhere in the first column) are almost always above 1.0, while those that do not include ratios are almost always below 1.0, i.e., lead to an improvement over default accuracy. Apparently, the unrestricted numeric range of this feature allows for too much overfitting.

The table without the ratio feature would contain only seven values above 1.0 (including the two for the absolute landmarks, LM), while 23 values would be below 1.0. Thus, one could be lead to the conclusion that relative landmarking (with the exception of ratios) does in most cases improve over absolute landmarking. However, a closer

analysis of the performances on the individual algorithm combinations reveals that the situation is more controversial.

Table 3 shows the average results over all feature sets for all pairwise algorithm comparisons. Despite the seemingly acceptable overall performance of the relative landmarkers, this analysis shows that the improvement is mostly due to a very significant improvement in one case: the question whether boosting decision trees improves performance can be answered surprisingly well. A closer examination of the results on this meta data set (not shown in detail, but cf. Table 4) shows that the use of regular landmarking has an error rate of 35.42% on this problem (for both C4.5 and Ripper), which is a increase over the default accuracy by about 70%. However, the results for all relative landmarking techniques are in the range of 4.17% to 16.67%, i.e., they all decrease the perfomance considerably. The best results are obtained whenever ranked features are used as (part of) the feature set. In these cases, the meta-learners invariably learn the following rule (with an error rate of 4.17%, i.e., it makes the wrong choice in only 2 of 48 datasets).

`IF` Best Node is the best landmarker `THEN` C50tree `ELSE` C50boost

The condition, namely that the Best Node landmarker outperforms all other landmarkers, presumably encodes whether the dataset contains many irrelevant features. On the one hand, their presence may have a detrimental effect on the performance of both linear discriminant and Naive Bayes, as well as decision tree algorithms, in particular at lower levels of the tree where discriminatory attributes have to be selected based small on samples. Boosting may reduce the variance that is introduced by such effects. On the other hand, irrelevant attributes will have no effect on the Best Node landmarker if the dataset contains at least one strongly relevant feature. With this interpretation, the rule encodes the fact that boosting will help considerably if there are many irrelevant features, while it may hurt if there are strongly relevant ones (in this case, the re-weighting in the boosting iterations may effect the algorithms' performance by eventually forcing it to select less relevant attributes). While this interpretation is quite intuitive and seems to make sense, we think its operational formulation and its discovery within the landmarking framework is interesting and surprising.

4 Subsampling Landmarkers

We now introduce another technique, the use of subsampling estimates as landmarkers. The basic idea is simple: instead of resorting to computationally simple algorithms, one could also select computationally complex algorithms, but evaluate their performance only on a subset of the available data. In fact, we could include evaluations of the same algorithms whose performance we want to predict, on smaller subsets of the data. Clearly, evaluating the algorithms on subsets of the available data is cheaper than a full evaluation of the algorithm on the entire dataset, so that the meta-learning approach may still save costs compared to the latter approach. In addition, we believe that this approach has another interesting property:

Similarity of Regions of Expertise: It is not unreasonable to expect that the region of expertise of an algorithm on the collection of "full" data sets corresponds fairly well to the region of expertise on the collection of subsamples of these databases.

This is expectation is based on the results in [11], where the extent to which the order of fast subsampling estimates of classifier performance correspond to their true performance ranking was investigated.

We would also expect that it is useful to combine subsample estimates with relative landmarking. Consider, e.g., a straight-forward benchmark classifier that simply picks the algorithm that has the better performance on the subsample of the specified size. This classifier could be encoded in the form of a simple rule with one condition (namely the relation of the subsample landmarks). This means that a meta learner the uses a relative encoding of subsample landmarks should be able to fall back on this default classfier, and, ideally, be able to improve upon it by learning a more complex theory. Similarly, the ratio of subsample landmarkers obtained for different training set sizes could yield information about the slope of the learning curve at this point, which might be a useful feature for learning.

We evaluated the performance of subsample landmarkers in the following way: for each algorithm pair we included the performance estimates of these two algorithms on subsamples of fixed sizes with 100 and 200 examples (recall the minimum example size in our 48 datasets is 1000). The performance estimate is computed by evaluating the theories learned on this sample on 10% of the total data. Hence, four landmarkers were generated for each dataset. We evaluated them in isolation or in addition to the original landmark values described in the previous section, and subjected both landmark ensembles to the same series of experiments with relative encodings as described in the previous section.

The averages over all pairwise classification tasks or over all combinations of feature representations (not shown) are considerably worse than their counterparts in Tables 2 and 3, i.e., they predominantly contain entries that are larger than 1. The main reason for this is that the powerful rule for recognizing boosting potential is no longer found in the case of using only subsample landmarks and much harder to find in the case of combining subsample and regular landmarks because the subsample landmarks may interfer with the ordering. This can be seen from Table 4 that compares the performance ratios of the individual feature representations (i.e., using only one of the features) for regular landmarking, subsample landmarking, and both. Due to space limitations we can only show the results for Ripper as a base learner here. The results for C4.5 can be found in [6].

The results are very diverse and inconclusive, but again mostly negative. There are a few cases (in particular with Ltree (first four columns)) where the performance seems to be a bit better, but this can be fairly well explained when one looks at the last two lines: these show the results of simply picking the algorithm that has the better performance on the subsample of the specified size, as discussed above. For some feature sets (in particular those that could directly encode it, e.g., RL), the results are fairly well correlated with this dummy classifier, but the improvement we hoped for did not occur. One reason might be that the additional information (the addition of the regular land-

Table 4. Results for individual feature set types for all pairwise combinations of the base learners for regular landmarks, subsampling landmarks and both.

C4.5		r-l	ct-l	cb-l	cr-l	ct-r	cb-r	cr-r	cb-ct	cr-ct	cb-cr
Ripper		r-l	ct-l	cb-l	cr-l	ct-r	cb-r	cr-r	cb-ct	cr-ct	cb-cr
LM	regl.	0.88	1.10	1.15	1.24	1.00	1.11	1.14	1.70	0.83	1.10
	subs.	1.18	1.15	1.00	1.12	1.00	1.22	1.36	1.10	0.75	1.20
	both	0.65	1.15	1.00	0.94	1.31	1.11	1.14	1.80	0.92	1.10
RK	regl.	1.41	1.70	1.23	1.00	1.54	1.33	1.36	0.20	1.50	0.60
	subs.	0.94	0.75	1.23	0.77	1.00	1.00	1.50	1.10	1.25	1.30
	both	1.18	1.10	1.23	1.41	1.00	0.89	1.07	0.70	1.58	0.80
OR	regl.	1.00	1.25	0.85	1.00	1.54	1.22	1.21	0.50	1.50	0.50
	subs.	1.41	1.05	0.77	0.82	1.23	1.22	0.79	1.00	1.33	1.50
	both	1.18	0.95	1.46	1.41	1.15	0.78	1.36	0.80	1.25	1.20
RL	regl.	1.00	1.40	0.92	1.00	1.08	1.22	1.07	0.40	0.67	0.40
	subs.	0.94	0.80	0.92	0.82	1.08	1.00	1.36	1.20	1.00	1.50
	both	1.18	0.85	0.92	1.41	1.23	1.78	1.43	0.40	1.17	0.80
RT	regl.	0.82	1.20	1.69	1.18	1.39	1.00	1.29	0.50	1.08	0.90
	subs.	0.77	0.80	0.92	1.24	1.08	1.22	1.29	1.20	1.33	1.60
	both	0.94	0.90	1.31	1.59	1.31	1.67	1.00	0.90	1.17	1.00
SS	100	1.11	0.65	1.23	0.88	1.62	2.00	1.43	1.40	1.25	1.40
	200	1.24	0.70	0.92	0.71	1.69	1.56	1.64	1.50	1.08	1.30

markers when moving from line 2 to line 3) does not add valuable information. Maybe additional features from DCT would allow to learn more informative rules.

The last two lines also show that the quality of the subsample landmarkers is very bad. For example, one would expect that the performance of the benchmark classifier best on 200 examples should be above that based on 100 examples but this is not the case. This hints that our subsampling strategy was far too simple and the variance in the estimates is far too high to obtain useful results. This can also be seen from the fact that often, in particular with C4.5 and ordered feature sets, all features are ignored and the algorithm consistently falls back to default prediction (ratio 1.00).

5 Conclusion

In this paper we have introduced two novel twists to the landmarking meta-learning technique, the use of subsample estimates for the target algorithms as landmarkers and the use of relations between the landmarkers instead of using their absolute values.

To evaluate these techniques, we have performed a total of $3 \times \times \binom{5}{2} \times (2^5 - 1) \times 2 = 1860$ leave-one-out cross-validations: each possible combination of 5 landmarking techniques (1 absolute, 4 relative) was evaluated for each pairwise combination of 5 base algorithms for 2 meta-learning algorithms and 3 different sets of base landmarkers (conventional landmarkers, subsampling landmarkers, and both). In summary, our results seem to indicate that the average performance of landmarking—relative or absolute—is quite bad.

We are not yet clear about the generality of these negative results. It still needs to be explored how our techniques work together with the statistical data characterisations provided by DCT [10] (in particular in connection with feature subset selection [14, 8]), and whether they generalize to the more general scenario of algorithm ranking instead of algorithm selection, in particular if the ranking is based on both accuracy and run-time [4]. Our experimental set-up differed in several aspects from the scenarios studied in previous works [12, 2], where the reported results were more encouraging: first, although we followed the setup of one of the experiments reported in [12] for the base algorithms C50boost, Ripper, and Ltree, their results were based on a meta dataset with 216 examples, which was generated from artificial data, and used considerably more expensive landmarkers (e.g., cross-validated C50tree). Clearly, having larger meta datasets will allow the meta learner to more reliably identify important data characteristics and might improve the results. In experiments with real-world data, previous works typically compared representatives from different families of learning algorithms (e.g., decision trees vs. rules vs. neural networks). Among different groups, the performance differences are usually larger than in our scenario, where the performance of all five base learners was very similar. Other scenarios included the use of significant tests for deciding on whether an algorithm should be preferred (which we shunned because of the loss in training data) as well as trying to predict whether an algorithm is better than average.

We are convinced that our results could be improved by increasing the effort that can be devoted to computing the landmarks. For example, more expensive landmarking algorithms could be used (as in some experiments in [12]). More importantly, we would expect that subsampling landmarkers could eventually result in better performance if bigger subsamples are used or the variance in the estimates is reduced by averaging over multiple subsamples of the same size. In fact, we plan to do a more thorough investigation of subsampling landmarkers of different sizes. The large number of experiments performed for this paper prevented us from backtracking right away. Besides, we are also convinced that these negative results are worth to be reported.

However, when moving to more expensive landmarkers, one must always keep in mind that there is a trade-off between accuracy and time invested for computing the data characterisations. Eventually, there will be a point where it makes more sense to directly evaluate the base algorithms instead of investing a lot of effort into the computation of data characterisation measures. This issue has not been addressed in research on meta-learning so far.

Notwithstanding the overall negative results, we have nevertheless seen that relative landmarkers can significantly improve performance on individual meta-learning tasks. In particular, they allowed the formulation of a simple and surprising rule for deciding whether to use c50's boosting option or not, which we think tests for the presence or absence of irrelevant values. We would see this finding as our main positive result, and plan on future work to investigate its validity in more depth.

Acknowledgments

This research was financed by the ESPRIT long-term research project METAL (project nr. 26357). The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry of Education, Science and Culture.

References

1. Hilan Bensusan. God doesn't always shave with occam's razor - learning when and how to prune. In C. Nédellec and C. Rouveirol, editors, *Proceedings of the 10th European Conference on Machine Learning (ECML-98)*, pages 119–124, 1998.
2. Hilan Bensusan and Christophe Giraud-Carrier. Casa Batló is in Passeig de Gràcia or landmarking the expertise space. In J. Keller and C. Giraud-Carrier, editors, *Proceedings of the ECML-00 Workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination*, pages 29–46, Barcelona, Spain, 2000.
3. Hilan Bensusan, Christophe Giraud-Carrier, and Claire Kennedy. A higher-order approach to meta-learning. In *Proceedings of the ECML'2000 workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination*, pages 109–117. ECML'2000, June 2000.
4. Pavel B. Brazdil and Carlos Soares. A comparison of ranking methods for classification algorithm selection. In *Proceedings of the 11th European Conference on Machine Learning (ECML-2000)*, pages 63–74, Barcelona, Spain, 2000. Springer-Verlag.
5. William W. Cohen. Fast effective rule induction. In A. Prieditis and S. Russell, editors, *Proceedings of the 12th International Conference on Machine Learning (ML-95)*, pages 115–123, Lake Tahoe, CA, 1995. Morgan Kaufmann.
6. Johannes Fürnkranz and Johann Petrak. An evaluation of landmarking variants. Technical Report OEFAI-TR-2001-13, Austrian Research Institute for Artificial Intelligence, Wien, Austria, 2001.
7. João Gama and Pavel B. Brazdil. Linear tree. *Intelligent Data Analysis*, 3(1):1–22, 1999.
8. Alexandros Kalousis and Melanie Hilario. Feature selection for meta-learning. In *Proceedings of the 5th Pacific Asia Conference on Knowledge Discovery and Data Mining (PAKDD-01)*. Springer-Verlag, 2001.
9. Alexandros Kalousis and T. Theoharis. Noemon: Design, implementation and performance results of an intelligent assistant for classifier selection. *Intelligent Data Analysis*, 3(5):319–337, November 1999.
10. Guido Lindner and Rudi Studer. AST: Support for algorithm selection with a CBR approach. In C. Giraud-Carrier and B. Pfahringer, editors, *Proceedings of the ICML-99 Workshop on Recent Advances in Meta-Learning and Future Work*, Bled, Slovenia, 1999.
11. Johann Petrak. Fast subsampling performance estimates for classification algorithm selection. In J. Keller and C. Giraud-Carrier, editors, *Proceedings of the ECML-00 Workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination*, pages 3–14, Barcelona, Spain, 2000.
12. Bernhard Pfahringer, Hilan Bensusan, and Christophe Giraud-Carrier. Meta-learning by landmarking various learning algorithms. In *Proceedings of the 17th International Conference on Machine Learning (ICML-2000)*, Stanford, CA, 2000.
13. John Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.
14. Ljupčo Todorovski, Pavel Brazdil, and Carlos Soares. Report on the experiments with feature selection in meta-level learning. In P. Brazdil and A. Jorge, editors, *Proceedings of the PKDD-00 Workshop on Data Mining, Decision Support, Meta-Learning and ILP: Forum for Practical Problem Presentation and Prospective Solutions*, pages 27–39, Lyon, France, 2000.
15. Ljupčo Todorovski and Sašo Džeroski. Experiments in meta-level learning with ILP. In *Proceedings of the 3rd European Conference on Principles of Data Mining and Knowledge Discovery (PKDD-99)*, pages 98–106. Springer-Verlag, 1999.