Multi-Objective Genetic Programming for Dataset Similarity Induction

Jakub Šmíd, Martin Pilát, Klára Pešková
Charles University in Prague
Faculty of Mathematics and Physics
Malostranské nám. 2/25
Prague, Czech Republic
Email: jakub.smid@ktiml.mff.cuni.cz,
martin.pilat@mff.cuni.cz, klara@pisecko.cz

Roman Neruda
Institute of Computer Science
Academy of Sciences of the Czech Republic
Pod Vodárenskou věží 2
Prague, Czech Republic
Email: roman@cs.cas.cz

***Abstract*—Metalearning – the recommendation of a suitable machine learning technique for a given dataset – relies on the concept of similarity between datasets. Traditionally, similarity measures have been constructed manually, and thus could not precisely grasp the complex relationship among the different features of the datasets. Recently, we have used an attribute alignment technique combined with genetic programming to obtain more fine-grained and trainable dataset similarity measure. In this paper, we propose an approach based on multi-objective genetic programming for evolving an attribute similarity function. Multi-objective optimization is used to encourage some of the metric properties, thus contributing to the generalization abilities of the similarity function being evolved. Experiments are performed on the data extracted from the OpenML repository and their results are compared to a baseline algorithm.**

I. Introduction

Metalearning [1], the task of selecting the best machine learning method for a given dataset, tries to compensate for the fact that according to the no free lunch theorem [2], there is not a single machine-learning method that provides the best results for all possible datasets.

The main idea behind metalearning is that machine learning methods are supposed to perform similarly on similar datasets, therefore, the notion of dataset similarity is crucial. In most cases, the similarity is computed using some kind of data about the dataset – *the metadata*. Metalearning techniques use the metadata and previous experience to predict the performance of machine-learning methods on new datasets. In essence, metalearning does not differ much from the traditional machine learning. The main difference is that metalearning works with the metadata of given datasets instead of the actual data, and that the result of metalearning is a recommendation of a machine learning method to use.

The metadata may contain general information about the dataset, like the number of instances and attributes, the number of classes, etc. However, it can also contain information about the individual attributes, its range, standard deviation, correlation to the target class, and many more. To distinguish between these two types of metadata, we call the former *general metadata* and the later *attribute-specific metadata*.

The attribute-specific metadata include more fine-grained information about the dataset. On the other hand, each dataset can have a different number of attributes (and consequently, different amount of attribute-specific metadata), which leads to a more complicated computation of the similarity. This problem has been solved in our previous work [3], [4] where the Hungarian algorithm [5] was used for the metadata alignment. Later [6], we have used *genetic programming* (GP) to evolve similarity measures between datasets. We found out that using a transformation to ensure three out of four metric axioms of the evolved similarity function improves the results. In this work, we use a different approach. Instead of repairing the function evolved by the genetic programming, we use multi-objective optimization to change the search in such a way that functions with metric properties are preferred. We use two objectives, one of them being the error rate of the model. The other expresses how well the evolved similarity matches the properties of metrics. The later objective can be also seen as a regularization term in the traditional optimization. The main goal of this paper is to show that such a multi-objective approach is able to create useful similarity measures.

The structure of this paper is as follows: the next section presents an overview of related work that has inspired us or concerns our research area. Section 3 summarizes our work that we build upon in this paper. Section 4 discusses the important properties of the distance function and how it is possible to evolve them. Then the multi-objective GP algorithm is proposed and its properties are discussed. Section 5 describes the experiments and compares the results to a baseline algorithm. The paper concludes with a discussion and future work.

II. Related work

Most metalearning approaches deal only with the general metadata. Brazdil and Soares [7] use various information theoretic metadata. With this metadata, they use *k-Nearest Neighbors algorithm* (k-NN) to detect the most similar datasets in what they call the *zooming phase*, followed by the *ranking* phase, where the methods used on the datasets from the zooming phase are evaluated and the best of them is recommended. The evaluation is based both on accuracy and training time. On the other hand, Kazík *et al.* [8] used metadata consisting of the number of instances and attributes together

978-1-4799-7560-0/15 $31.00 © 2015 IEEE
DOI 10.1109/SSCI.2015.222

with some information about the task at hand. They always recommend the best method on the closest dataset. The metric between the datasets was a weighted sum of distances between the individual metadata. Weights used were optimized by an evolutionary algorithm.

The problem of metalearning has also been stated from the point of view of collaborative filtering [9]. The main advantage of this approach is that it does not require any metadata about the datasets, however, it can still use such metadata if they are available.

Smith-Miles [10] used PCA to visualize space of optimization problem instances. She then employed search algorithms to find instances that are hard for each solver and provided a map of search space based on the expected footsteps of different optimization algorithms.

Šmíd and Neruda [3], [4] worked on the problem of using the attribute-specific metadata and were able to express it as an assignment problem, which can be solved by the Hungarian algorithm [11] in $O(n^3)$. Their approach requires a distance function between the individual attributes. In case the number of attributes for each of the datasets is different, dummy attributes are introduced to make it even before applying the Hungarian algorithm. Later [6], they used genetic programming and co-evolution to evolve the distance functions between the individual attributes.

In this work, we extend this approach by using multi-objective optimization. Even though the problem at hand is in fact single-objective (we try to minimize the error rate of the algorithm), it can be also expressed as a problem with more objectives. Such an approach is called *multi-objectivization* and it has been shown that it can improve the performance of single-objective optimizers, especially in cases where the optimized function contains plateaus [12]. Pilát and Neruda [13] used multi-objectivization for the hyper-parameter tuning of classifiers, they added two objectives to guide the search – the root mean squared error and the kappa statistic – while they tried to optimize the hyper-parameters for the best accuracy of the model. In the field of machine learning, multi-objective optimization can also be used for regularization [14]. In such case the regularizing term is added as another objective rather than summing it with the optimized criterion.

In order to be able to solve multi-objective problems, we need a multi-objective algorithm. There has been a large number of such algorithms proposed in the past. In this work we will use the NSGA-II algorithm [15]. Although this algorithm is rather old, it is still among the best optimizers for two-objective problems (which is what we use in this paper). The main difference between a single-objective and a multi-objective evolutionary algorithm is in the selection operator. The selection of NSGA-II is based on Pareto dominance. A solution is said to dominate another solution if the values of all the objectives are better for the former one. The solutions in the population are thus divided into non-dominated fronts. The first front contains solutions, which are not dominated by any other solution in the population. The second front contains individuals dominated only by solutions from the first front and so on. Individuals from fronts with lower number are preferred in the evolution. To distinguish among individuals in the same front a secondary sorting criterion is used. In the case of NSGA-II this criterion expresses how crowded the region around each individual is and individuals from less crowded regions are preferred.

III. Ranking Through Alignment

Given a new data-mining task (e.g. a classification task), ranking orders machine-learning methods according to their predicted performance on the given task. In this section, we describe our approach to ranking using attribute alignment.

The usual assumption in metalearning is that algorithms perform similarly on similar datasets. To exploit the idea, dataset similarity measure is needed along with an algorithm that can utilize known results on the set of similar datasets. Often, the similarity measure is almost exclusively based on general metadata extracted from datasets.

In [4] we proposed a method that goes one step further – it extracts attribute-specific metadata from the datasets instead of dataset metadata. Given the attribute similarity measure, it is possible to find an assignment of attributes of one dataset to the attributes of the other dataset, such that the attributes with similar metadata are matched together. Thus, the attributes can be compared with each other which provides more precise comparison of datasets than when dealing only with general metadata.

More formally, if the two datasets we compare have the same number of attributes, the distance between the attributes is defined as follows:

Definition 1: Given an attribute similarity measure d_{att}, two datasets a and b with the same number of attributes and a bijection f between the attributes of a, b we define the *attribute distance* induced by the bijection between the datasets as:

$$d_f(a,b) = \sum_{k=1}^{n} (d_{att}(a_k, f(a_k))). \tag{1}$$

We would like to match the attributes as best as possible, so optimally we are looking for f':

$$f' = \mathrm{argmin}_f(d_f). \tag{2}$$

In case that the number of attributes is different, we transform this problem to the previous one by adding dummy attributes into the dataset with less attributes and by supplying the distance between a regular attribute and a dummy one.

Our goal is to find a good d_{att}. It is arguable whether one should come up with the distance function between attributes that covers all cases including the distance between a categorical and numerical attribute. Metafeatures that can be extracted from categorical and numerical attributes naturally vary, thus making it difficult to specify a distance. To solve this problem we have split the distance into two parts: the distance between numerical attributes and the distance between categorical attributes. As the total distance we have taken the sum of the sub-distances:

$$\begin{aligned} TotalDistance &= Distance_{categorical} + \\ &+ Distance_{numerical} \end{aligned} \tag{3}$$

Given d_{att}, the so called Hungarian algorithm [5] running in $O(n^3)$ can be used to find f' for the optimal assignments

Algorithm 1 AttributeDistance(d_{att},att_a,att_b)

Require: d_{att} ▷ attribute similarity measure
Require: att_a ▷ attributes of dataset a
Require: att_b ▷ attributes of dataset b
1: Add dummy attributes into the list with less attributes
2: $M_{i,j} \leftarrow d_{att}(att_a[i], att_b[j])$ ▷ distance matrix
3: $f' \leftarrow \arg\min_f(d_f)$. ▷ Hungarian algorithm
4: **return** $d_{f'}$

Algorithm 2 DatasetSimilarity(a,b,d_{cat},d_{num})

Require: a, b ▷ datasets
Require: d_{cat} ▷ categorical attribute similarity
Require: d_{num} ▷ numerical attribute similarity
1: $a_c \leftarrow$ catAttr(a); $b_c \leftarrow$ catAttr(b) ▷ categorical attrib.
2: $a_n \leftarrow$ numAttr(a); $b_n \leftarrow$ numAttr(a) ▷ numerical attrib.
3: $dist_{cat} \leftarrow$ AttributeDistance(d_{cat}, a_c, b_c)
4: $dist_{num} \leftarrow$ AttributeDistance(d_{num}, a_n, b_n)
5: **return** $dist_{cat} + dist_{num}$

of categorical and numerical attributes, allowing to compute $TotalDistance$. This calculation of the distance function will be used by k-NN algorithm to calculate ranking. A boosting algorithm could be used to further improve the results. However, if we want to asses the quality of distance measure, it would be counter-productive as we could be accidentally assessing the quality of the boosting algorithm.

To asses the quality of ranking for one dataset the *Spearman's rank correlation coefficient* will be used:

$$r_s = 1 - \frac{6\sum_{i=1}^{n}(R'_i - R_i)^2}{n^3 - n}, \tag{4}$$

where n is the number of models, R_i is the actual rank and R'_i is the predicted rank. The Spearman's rank correlation coefficient returns values in the range $\langle -1, 1\rangle$, 1 is perfect match, -1 is perfect mismatch and 0 indicates results as good as random guessing. To asses the quality for all datasets, we take average of Spearman's rank correlation coefficient over all datasets.

The whole ranking quality assessment is summarized in the pseudocode of the following three algorithms: 1, 2, and 3. The algorithm 3 takes algorithm 2 as a dataset similarity measure, algorithm 2 calls algorithm 1 to calculate $TotalDistance$. The only input missing (beside datasets) to provide exact calculation are the attribute similarity measures.

IV. Attribute Similarity Measure Evolution

In this section we propose a multi-objective genetic programming (GP) based algorithm to evolve categorical and numerical distance measure.

As we stated before, the goal of this work is to find a good attribute distance measures that will be used as an input in algorithm 2. As the desired attribute distance is not known, the supervised learning cannot be used for this task. Once we have the measure function, we still need many evaluations of this function in order to calculate ranking and assess its quality. In algorithm 3, every dataset is enumerated.

Algorithm 3 QualityAssessment(DB,sim)

Require: DB ▷ sets of all datasets
Require: sim ▷ dataset similarity measure
1: coef $\leftarrow 0$
2: **for all** data $\in DB$ **do**
3: rest $\leftarrow DB \setminus \{data\}$
4: $r \leftarrow$ k-NN(data,rest,sim) ▷ predict ranking
5: sp $\leftarrow$ Spearman(r, DB) ▷ compare with actual results
6: coef $\leftarrow$ coef + sp
7: **end for**
8: **return** coef / $|DB|$

The k-NN then calculates the distance for each dataset to the currently enumerated dataset. During the distance calculation, the Hungarian algorithm is used and the distance matrix is calculated. The size of matrix for a dataset pair is the square of the number of attributes of the dataset with more attributes. For each matrix entry, the similarity measure is invoked. In this sense, we are dealing with reinforcement learning, where the space searched is the space of functions mapping Cartesian product of attribute metafeatures space to real numbers: $f : \{metafeatures\} \times \{metafeatures\} \rightarrow \mathbb{R}$. The genetic programming is a suitable method for this kind of task as it can search the space based on the ranking quality assessment and it does not need a model of the environment like some other reinforcement learning algorithms.

In order to asses fitness of an individual, both numerical and categorical distances are needed (see algorithm 2). Both distances were evolved as one individual.

When measuring a distance in general, a metric is typically used. In our case, it may also be beneficial for the generalization abilities of the evolved function, if our attribute similarity measure had some of the properties of a metrics, which are:

1) $d(x, y) \geq 0$ (non-negativity)
2) $d(x, y) = 0 \Leftrightarrow x = y$ (coincidence axiom)
3) $d(x, y) = d(y, x)$ (symmetry)
4) $d(x, z) \leq d(x, y) + d(y, z)$ (triangle inequality)

To increase the evolution pressure on searching mainly the metric subspace of the search space, second criterion was added to the fitness. This criterion measured the evolved function similarity to metric and consisted of four parts, one per each metric axioms:

$$fitness_{distance} = \frac{a_1 + a_2 + a_3 + a_4}{4}, \tag{5}$$

where a_i is the ratio of the cases in data that maintains the property of the axiom i. It can be seen from the formula that every part of the distance fitness function has the same weight.

For the multi-objective fitness, we have used the same GP settings as we have used in [6], these are repeated later in this section in order to make the paper self-contained.

Multi-objective genetic programming with the tree representation is employed to search the space of similarity measures. The fitness is composed of two objectives, the first one measures its ability to predict dataset similarity, the latter is used to enforce metric properties. NSGA-II will be utilized to drive the evolution towards the optimal Pareto front. We have

used type consistent genetic programming with real numbers as the only type accepted and returned by all functions. If an n-ary function was not defined on the whole $\mathbb{R}^n$, we have proposed its extended definition on the whole $\mathbb{R}^n$. Boolean terminals were converted to 0/1 real terminals.

The non-terminal set of the genetic programming includes basic arithmetic functions – addition, subtraction, multiplication, and division – and also a few more advanced functions – square root and logarithm. Some of these functions had to be extended in such a way that they are defined for all real numbers. The following generalization was chosen:

$$\mathrm{divide}(a_1, a_2) = \begin{cases} \frac{a_1}{a_2}; & a_2 \neq 0 \\ 0; & \text{otherwise} \end{cases}$$

The square root and base 2 logarithm functions were generalized as follows:

$$\mathrm{sqrt}(a_1) = \begin{cases} \sqrt{a_1}; & a_1 \geq 0 \\ \sqrt{|a_1|}; & \text{otherwise} \end{cases}$$

$$\log_2(a_1) = \begin{cases} \log_2 a_1; & a_1 > 0 \\ 0; & a_1 = 0 \\ \log_2 |a_1|; & \text{otherwise} \end{cases}$$

Apart from the arithmetical functions, we have also used comparisons of two types – less then, and less then or equal. For the sake of type consistency they were defined as follows:

$$\mathrm{less}(a_1, a_2, a_3, a_4) = \begin{cases} a_3; & a_1 < a_2 \\ a_4; & \text{otherwise} \end{cases}$$

$$\mathrm{lessorequal}(a_1, a_2, a_3, a_4) = \begin{cases} a_3; & a_1 \leq a_2 \\ a_4; & \text{otherwise} \end{cases}$$

We wanted GP to be able to evolve both the numerical and categorical similarity function. To achieve this, we used different terminal set for each type. The terminals contain several statistic measures [16], and were divided into sets as follows:

Measures common for both terminal sets (in some cases categorical attributes were cast to integers for the computation of some attributes, e.g. for the computation of the correlation coefficient):

- Whether the attribute contains missing values,
- Whether the attribute is continuous,
- Whether the target is discrete or continuous (this could change the meaning of some additional metadata),
- Entropy of values of the attribute,
- Pearson Correlation coefficient,
- Spearman's rank Correlation coefficient.

Numerical terminal set:

- Minimum of the attribute,
- Maximum of the attribute,
- Mean of the attribute,
- Standard deviation of the attribute,
- Variance of the attribute,
- Skewness of the attribute,
- Kurtosis of the attribute,
- Result of the Kolmogorov-Smirnoff test for uniform distribution,
- Covariance between attribute and target (continuous target only).

Categorical terminals:

- Result of χ^2 test for uniform distribution,
- Ratio of pairs of values of the attribute which lead to different distribution of the target as indicated by the following statistical test:
  - Kolmogorov-Smirnoff test (continuous target only),
  - Mann-Whitney U-test (continuous target only),
  - χ^2-test (categorical target only).

Each terminal was generated with a Boolean value, which determines whether the terminal used the value from the first dataset or the second dataset (that are being compared by the evolved similarity measure).

V. Experiments

To assess the performance of the above described technique, we conducted a series of experiments on large amount of data. In this section, we first describe the data we used, then we discuss the settings of the GP algorithm and finally we provide the results of the experiments followed by evaluation of the results and discussion.

Both datasets and performance results of machine learning algorithms on those datasets are needed to evaluate a GP individual. As the source of data, the OpenML [17] machine learning repository was used. In total, 418 datasets were retrieved from the repository and used for metadata extraction. There were 282,885 raw performance records in the form {dataset, algorithm, parameters, performance results} that were downloaded from the repository. However, some additional filtering was applied to these raw records. Firstly, datasets with more than 50 attributes were omitted as those increased the time to evaluate one individual significantly. These changed the number of datasets slightly, but decreased the time needed to conduct the experiments by orders of magnitude. Secondly, only the result with the highest performance (and best parameter settings) was used for each pair {algorithm, dataset}. To identify these best results the Predictive Accuracy – the percentage of instances that were classified correctly by the algorithm – was used as a performance indicator. The reason for this was that we wanted to asses the best potential of algorithms. Finally, the results of ensemble algorithms like bagging, boosting, stacking and rotation forests [18] were omitted. These ensembles are encompassing different number of other machine algorithms and can take advantage of such combined power. Such composite behaviour resulted in heavily

outperforming every other algorithm in the database thus changing the results significantly. Furthermore, their performance relies heavily on the parameters that specify what algorithms should be used in the ensemble and as such, every parameter setting should be better treated as a separate algorithm. More careful examination is thus necessary before including ensemble algorithms into our experiments. To sum up, 23962 pre filtered rows of performance records of machine learning algorithms were used as data source. These rows contained records of 116 algorithms over 360 datasets. The data were randomly split into the training and testing sets with ratio 2:1 in favour of the training set.

The NSGA-II GP algorithm, as described in the previous section, was used to perform experiments. For each dataset do: estimate the ranking of algorithms by using k-NN. For the needs of k-NN, compute distance between every remaining dataset. The distance computation is n^3, n is the number of attributes. This gives us a rough estimation of the number of attribute distance function evaluation needed to asses the fitness of one individual:

$$n_d(n_d)^2(n_{att})^3, \tag{6}$$

where n_d is the number of datasets and n_{att} is the number of attributes in the dataset. Given amount of data we have, using the above equation we get an estimation of $125,000,000,000$ of evaluation of the evolved function in order to assess its quality. Indeed, on the single node of our computation cluster it took tens of minutes to compute a single fitness value. Given that our computation cluster consisted of 40 nodes and we wanted to have experiments no longer than few days, we drew some conclusions regarding the proposed experiment settings:

- small to medium population.
- none or very small parameter tuning.
- higher evolution pressure to empower quicker convergence.

The size of the population was set to 200. This was a compromise between having bigger and probably more diverse population and being able to compute the fitness of all individuals in a reasonable time. The NSGA-II algorithm uses tournament selection as a selection mechanism. The probability of better individual winning the tournament was set to 0.7. Also, the NSGA-II uses elitism, so the best individual are guaranteed to be copied to the next generation. Tree mutation and crossover were used as genetic operators. The probabilities of mutation and crossover were chosen based on our previous experiments and were set to 0.2 and 0.7 respectively. The termination criterion was set to 80 generations. This number was chosen based on our preliminary experiments. We have noticed that around this generation, the bloating occurs. Also, the improvement in fitness was not accompanied with the performance improvement on the testing set and the individuals tend to overfit. With this GP settings, it took our 40 nodes cluster two days to conduct a single experiment.

To be able to compare the results of our experiments a baseline algorithm as proposed in [19] was used. This baseline algorithm predicts the results based on average rank over all datasets in the metaknowledge base. The average Spearman's rank correlation coefficient of the baseline algorithm was 0.549

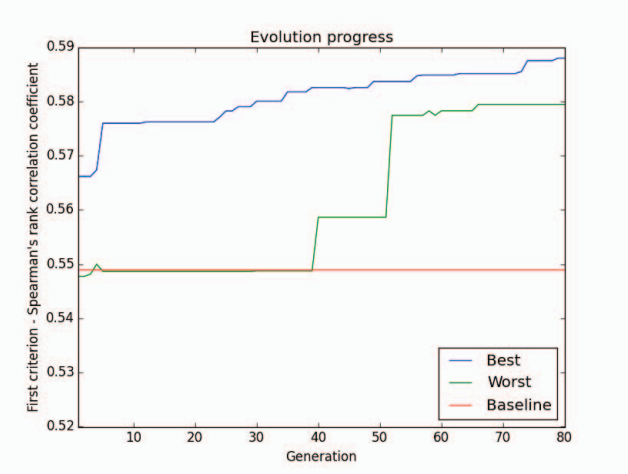

Fig. 1. Evolution progress of the best – seventh – experiment. The picture shows results of the first Pareto front – best and worst values of the first criterion.

on the training and 0.555 on the testing dataset. Note that the baseline algorithm does not use any distance function, thus one can only compare average Spearman's rank correlation coefficient of baseline and his own algorithm.

In total, seven runs were executed. The evolution progress of the best run from the perspective of the first criterion is depicted in the Fig. 1. The individual with the best first criterion surpassed the baseline on the training set at the very beginning of the evolution. The individual from the Pareto front with the worst first criterion struggled around baseline level until generation 39. From the generation 40 all individuals from the first Pareto front surpassed the baseline algorithm on the training set.

The output of each run was the first Pareto front of the multi-objective optimization. To be able to compare with the baseline algorithm, a single representative had to be chosen from each run. We decided to use the individual with the best first criterion on the training set. Spearman's correlation coefficients of such individuals on testing set are shown in the table I.

All values are better than the result of the baseline algorithm on the testing set. Based on these results we propose the null hypothesis that our experiments resulted in statistically significant improvement of the Spearman's correlation coefficients on the testing set. As no information was available about the probability distributions of the experiments performance, non-parametric statistical test had to be used to validate our results. Two tailed Mann-Whitney U Test was used in our case. The test passed the significance level of 0.05 with the p-value of 0.011. Based on this fact we accepted the null hypothesis and concluded that our results are indeed better than the results of the baseline algorithm.

Further examining the results showed another interesting property. The first and second criterion values of the first Pareto front was highly correlated on the testing set. This was most visible on the results of the second experiment. The values of both criterions of the second run are shown in the Figs. 2 and

TABLE I. RESULTS OF THE BEST INDIVIDUALS.

run	1	2	3	4	5	6	7
Spearman – training	0.577	0.580	0.582	0.587	0.583	0.585	0.588
Metric – training	0.996	0.996	0.996	0.979	0.995	0.912	0.999
Spearman – testing	0.558	0.555	0.560	0.560	0.564	0.569	0.567

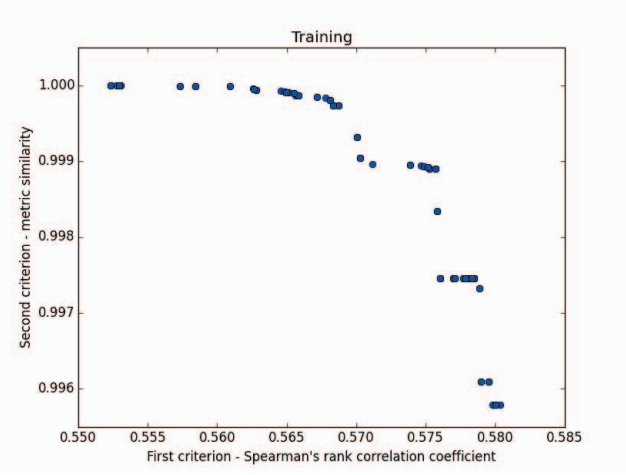

Fig. 2. Results of the first Pareto front of the second run on the training set.

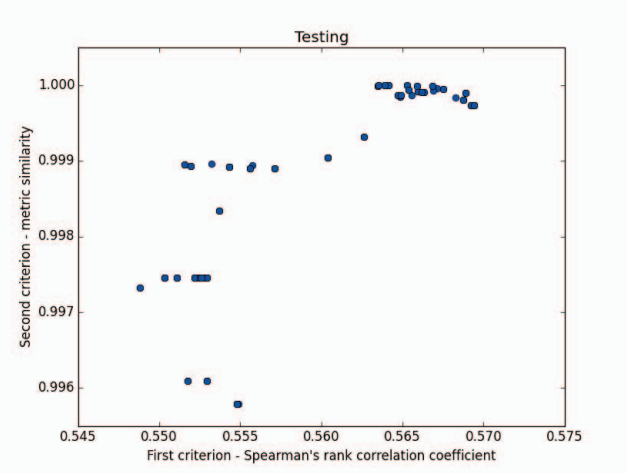

Fig. 3. Results of the first Pareto front of the second run on the testing set. Note the high correlation between both criteria - the higher values of the accuracy criterion are associated with higher values of the metric similarity criterion.

3. On the testing set, the values of the first criterion and the second criterion are highly correlated. The Spearman's rank correlation coefficient between the first and second criterion of the second run on the testing set is 0.734. This supports our hypothesis that metric properties are important for the generalization abilities of the evolved similarity measures and that the addition of the second criterion evaluating the function similarity to a metric was meaningful. Both objectives were also correlated during the other experiment runs, although not that significantly.

One could easily employ ensemble algorithms to boost the results of trained algorithms, however, the goal of the experiments was to prove the applicability of the method.

To summarize, we have performed seven GP runs. The results of the experiments were significantly better on the testing set compared to the baseline algorithm. The high correlation of the metric similarity and prediction accuracy has been observed, thus supporting the hypothesis that the metric properties are important for the generalization abilities of the induced dataset similarity measure.

VI. CONCLUSION

Predicting algorithm accuracy on the new dataset is an important and challenging problem. The essential part of this predictive metalearning algorithm is a solid function to compare the similarity of datasets. We have proposed multi-objective genetic programming algorithm to evolve such function. The features encouraged by evolution are the ability to predict accuracy and the metric properties. Number of experiments were conducted on the real data from the OpenML machine learning repository. We have verified that the experiments provided significantly better results than the baseline algorithm. The experiment results also suggest, that it is important for the similarity function to have metrics properties. Currently, the same weight is assigned to all metric axioms. However, there has been a long debate in the community, whether all these properties are equally important for the similarity function in metalearning. For example, there are arguments against the triangle inequality [20], a popular example is as follows: "a man is similar to a centaur, the centaur is similar to a horse, but the man is completely dissimilar to the horse". This example clearly violates the triangle inequality and speaks against using it. Moreover, it seems the coincidence axiom is also not very important. One can for example imagine, that some of the metadata are not important for the similarity, thus returning zero even in cases where these values are different would still make sense. Other arguments against using the coincidence axiom can be found in [21].

In our future work, we would like to use strongly typed genetic programming [22], [23] to improve the results of the algorithm. Further parameter tuning of the GP algorithm can be performed in order to increase the performance of our genetic algorithm. Furthermore, with the PCA algorithm modified for a different number of attributes a visualization technique similar to the one described in [10] can be used to visualize our results.

Acknowledgment

J. Šmíd and K. Pešková have been supported by the Charles University Grant Agency project no. 610214, M. Pilát and R. Neruda have been supported by the National Science Foundation of the Czech Republic project no. P103-15-19877S.

REFERENCES

[1] R. Vilalta and Y. Drissi, "A perspective view and survey of meta-learning," *Artificial Intelligence Review*, vol. 18, pp. 77–95, 2002.

[2] D. H. Wolpert, "The supervised learning no-free-lunch theorems," in *Proceedings of the 6th Online World Conference on Soft Computing in Industrial Applications*, 2001, pp. 25–42.

[3] J. Šmíd and R. Neruda, "Using genetic programming to estimate performance of computational intelligence models," in *Adaptive and Natural Computing Algorithms*, ser. Lecture Notes in Computer Science, M. Tomassini, A. Antonioni, F. Daolio, and P. Buesser, Eds. Springer Berlin Heidelberg, 2013, vol. 7824, pp. 169–178. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-37213-1_18

[4] J. Smid and R. Neruda, "Comparing datasets by attribute alignment," in *Computational Intelligence and Data Mining (CIDM), 2014 IEEE Symposium on*, Dec 2014, pp. 56–62.

[5] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, pp. 83–97, 1955.

[6] J. Šmíd, M. Pilát, K. Pešková, and R. Neruda, "Co-evolutionary genetic programming for dataset similarity induction," in *Evolutionary Computation (CEC), 2015 IEEE Congress on*, 2015, in print.

[7] P. B. Brazdil and C. Soares, "Zoomed ranking: Selection of classification algorithms based on relevant performance information," in *Proceedings of Principles of Data Mining and Knowledge Discovery, 4th European Conference (PKDD 2000)*. Springer, 2000, pp. 126–135.

[8] O. Kazík, K. Pešková, M. Pilát, and R. Neruda, "Meta learning in multi-agent systems for data mining," in *International Conference on Intelligent Agent Technology (IAT 2011)*. IEEE Computer Society, 2011, pp. 433–434.

[9] M. Misir and M. Sebag, "Algorithm Selection as a Collaborative Filtering Problem," Research Report, Dec. 2013. [Online]. Available: https://hal.inria.fr/hal-00922840

[10] K. Smith-Miles, D. Baatar, B. Wreford, and R. Lewis, "Towards objective measures of algorithm performance across instance space," *Comput. Oper. Res.*, vol. 45, pp. 12–24, May 2014. [Online]. Available: http://dx.doi.org/10.1016/j.cor.2013.11.015

[11] J. Edmonds and R. M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems," *Journal of the ACM*, vol. 19, no. 2, pp. 248–264, Apr. 1972. [Online]. Available: http://doi.acm.org/10.1145/321694.321699

[12] D. Brockhoff, T. Friedrich, N. Hebbinghaus, C. Klein, F. Neumann, and E. Zitzler, "On the effects of adding objectives to plateau functions," *Trans. Evol. Comp*, vol. 13, no. 3, pp. 591–603, Jun. 2009. [Online]. Available: http://dx.doi.org/10.1109/TEVC.2008.2009064

[13] M. Pilát and R. Neruda, "Multi-objectivization and surrogate modelling for neural network hyper-parameters tuning," in *Emerging Intelligent Computing Technology and Applications*. Springer Berlin Heidelberg, 2013, pp. 61–66.

[14] Y. Jin, *Multi-objective machine learning*. Springer, 2006, vol. 16.

[15] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 2, pp. 182–197, 2002.

[16] D. J. Sheskin, *Handbook of Parametric and Nonparametric Statistical Procedures*, 4th ed. Chapman & Hall/CRC, 2007.

[17] J. Vanschoren, J. N. van Rijn, B. Bischl, and L. Torgo, "Openml: Networked science in machine learning," *SIGKDD Explorations*, vol. 15, no. 2, pp. 49–60, 2013. [Online]. Available: http://doi.acm.org/10.1145/2641190.2641198

[18] J. Rodriguez, L. Kuncheva, and C. Alonso, "Rotation forest: A new classifier ensemble method," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 10, pp. 1619–1630, Oct 2006.

[19] P. Brazdil, C. G. Giraud-Carrier, C. Soares, and R. Vilalta, *Metalearning — Applications to Data Mining.*, ser. Cognitive Technologies. Springer, 2009. [Online]. Available: http://dblp.uni-trier.de/db/series/cogtech/index.html#0022052

[20] E. G. Ashby and N. A. Perrin, "Toward a unified theory of similarity and recognition," *Psychological Review*, vol. 95, pp. 124–150, 1988.

[21] C. L. Krumhansl, "Concerning the applicability of geometric models to similarity data: The interrelationship between similarity and spatial density," 1978.

[22] D. J. Montana, "Strongly typed genetic programming," *Evolutionary Computation*, vol. 3, pp. 199–230, 1994.

[23] T. Křen and R. Neruda, "Generating lambda term individuals in typed genetic programming using forgetful A*," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*, July 2014, pp. 1847–1854.