# Symbolic Regression and Feature Construction with GP-GOMEA applied to Radiotherapy Dose Reconstruction of Childhood Cancer Survivors

Marco Virgolin
Centrum Wiskunde & Informatica
Amsterdam, The Netherlands
Marco.Virgolin@cwi.nl

Tanja Alderliesten
Academic Medical Center
Amsterdam, The Netherlands
T.Alderliesten@amc.uva.nl

Arjan Bel
Academic Medical Center
Amsterdam, The Netherlands
a.bel@amc.uva.nl

Cees Witteveen
Delft University of Technology
Delft, The Netherlands
C.Witteveen@tudelft.nl

Peter A.N. Bosman
Centrum Wiskunde & Informatica
Amsterdam, The Netherlands
Peter.Bosman@cwi.nl

## ABSTRACT

The recently introduced Gene-pool Optimal Mixing Evolutionary Algorithm for Genetic Programming (GP-GOMEA) has been shown to find much smaller solutions of equally high quality compared to other state-of-the-art GP approaches. This is an interesting aspect as small solutions better enable human interpretation. In this paper, an adaptation of GP-GOMEA to tackle real-world symbolic regression is proposed, in order to find small yet accurate mathematical expressions, and with an application to a problem of clinical interest. For radiotherapy dose reconstruction, a model is sought that captures anatomical patient similarity. This problem is particularly interesting because while features are patient-specific, the variable to regress is a distance, and is defined over patient pairs. We show that on benchmark problems as well as on the application, GP-GOMEA outperforms variants of standard GP. To find even more accurate models, we further consider an evolutionary meta learning approach, where GP-GOMEA is used to construct small, yet effective features for a different machine learning algorithm. Experimental results show how this approach significantly improves the performance of linear regression, support vector machines, and random forest, while providing meaningful and interpretable features.

## CCS CONCEPTS

• **Computing methodologies → Machine learning**; **Genetic programming**; • **Applied computing → Life and medical sciences**;

## KEYWORDS

feature construction, genetic programming, machine learning, GOMEA, radiotherapy, dose reconstruction

**ACM Reference Format:**
Marco Virgolin, Tanja Alderliesten, Arjan Bel, Cees Witteveen, and Peter A.N. Bosman. 2018. Symbolic Regression and Feature Construction with GP-GOMEA applied to Radiotherapy Dose Reconstruction of Childhood Cancer Survivors. In *GECCO '18: Genetic and Evolutionary Computation Conference, July 15–19, 2018, Kyoto, Japan.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3205455.3205604

## 1 INTRODUCTION

Genetic Programming (GP) is a particularly interesting Machine Learning (ML) algorithm when dealing with regression, because it directly evolves mathematical expressions [10, 15]. While GP is in principle capable of generating white-box models, i.e., human-interpretable expressions, the evolved models are often overly complicated and far from being interpretable [11]. This aspect makes the use of GP questionable: why should GP be preferred over faster and similarly accurate ML algorithms like, e.g., support vector machines and random forest [1, 7], if both result in black-box models?

The Gene-pool Optimal Mixing Evolutionary Algorithm for GP (GP-GOMEA) is a recent, model-based algorithm which has been shown to achieve excellent scalability on synthetic Boolean benchmark problems [19], while evolving much smaller solutions than various competing algorithms. GP-GOMEA prevents bloat by construction, and performs variation based on a *linkage model*, i.e., a model that captures genotypic interdependencies. It is interesting to assess whether the fact that GP-GOMEA typically finds smaller solutions extends from Boolean functions to the domain of symbolic regression, and what accuracy can be reached.

As additional motivation, a regression problem in the medical domain is considered, where obtaining small, interpretable models can be of added value for clinicians. The problem is the regression of a notion of distance related to anatomical similarity among pediatric cancer patients, to ultimately enable studies on the late adverse effects of radiotherapy. A peculiarity of this problem is that features are relative to each individual patient, while the distance is measured on patient pairs. To succeed, an ML algorithm needs to learn how to combine individual features to model the distance.

We tackle this problem directly with GP-GOMEA, and compare the results with variants of standard GP and with well-known ML algorithms. Furthermore, to learn even more accurate models,

*GECCO '18, July 15–19, 2018, Kyoto, Japan*
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5618-3/18/07…$15.00
https://doi.org/10.1145/3205455.3205604

we propose an evolutionary meta learning approach where GP-GOMEA is used as a powerful non-linear feature constructor for an external ML algorithm (see [5] for a survey of feature construction by GP). We constrain the evolved features to be particularly small to make them very easy to interpret, and show their effectiveness in increasing the performance of the ML algorithm. Two additional benchmark regression problems are considered for a wider comparison of the algorithms.

## 2 GP-GOMEA

The Gene-pool Optimal Mixing Evolutionary Algorithm for GP (GP-GOMEA), recently introduced in [19], was shown capable of finding much smaller solutions than Standard GP (SGP) and other EAs for well-known synthetic benchmarks and Binary circuit regression, while achieving similar, or superior, scalability.

GP-GOMEA has the same general outline of the GOMEA framework, where, until the termination condition is met (e.g., a number of generations), a linkage model is learned and used by the variation operator Gene-pool Optimal Mixing (GOM) to generate an offspring for every solution in the population. The offspring is by construction at least as fit as the parent, thus a separate selection step is not needed. The model that GOM uses is a collection of *linkage sets*, called the *Family Of Subsets* (FOS). Each linkage set represents genotypic positions with strong interdependency, and specifies which genes are to be mixed during variation. The idea is that mixing interdependent genes *en bloc* prevents the disruption of their joint effect.

To conveniently perform model learning and variation, GP-GOMEA uses a tree representation of solutions different from the one of SGP in that trees have a fixed shape, i.e., they are always complete and full. All nodes which are above a predefined maximum depth $d$ have exactly $r$ child nodes, with $r$ the maximum number of input arguments among function nodes. When computing the output of the tree, if a function node uses only $r' < r$ child nodes as inputs, then the rightmost $r - r'$ child nodes are not executed. Similarly, all child nodes of a terminal node are not executed. The nodes that are not executed are called *introns*. All trees thus have a height $h = d$, and exactly $l = \sum_{i=0}^{h} r^i$ nodes, with some being introns. This means that the syntactic size of a solution is always $l$, but the size that has a semantic impact is *at most l*. By setting a small $d$, bloat is prevented by construction, and GP-GOMEA is forced to perform competent variation using the FOS model to find a good solution of limited size.

The FOS is learned every generation before applying GOM. The *Linkage Tree* (LT) FOS is often used in GOMEA because it has been shown to achieve solid performance on different problems [16, 19]. The LT captures hierarchical degrees of interdependency among nodes. This model is learned by measuring the mutual information between all possible pairs of locations in the genotype of the population, and performing hierarchical clustering. The computational effort to learn the LT is $O(\text{population-size} \times l^2)$. This is often a negligible overhead in GP, where computing the output of solutions is typically the performance bottleneck. Further details on the LT can be found in [16, 19]. In this work, GP-GOMEA was always run with the LT FOS.

Lastly, the GOM operator handles both variation and selection. Pseudocode illustrating GOM is shown in Algorithm 1. Given a solution, a copy representing the offspring and a copy for backup, are made. For each linkage set $\mathrm{LT}_i$ in the LT FOS (parsed in a random order to allow different mixing combinations), a random donor from the population is chosen, and the nodes at the position specified by $\mathrm{LT}_i$ are cloned from the donor into the offspring. If this cannot alter the behavior of the offspring, i.e., no node changed value or only intron nodes did, the iteration is concluded. Otherwise, the fitness of the offspring is immediately evaluated. If the fitness of the offspring is not worse than the one of the backup, then the latter becomes a copy of the former. Otherwise, the change is discarded, by reverting the offspring to the backup. This mixing behavior, so-called *optimal mixing*, is guaranteed to always perform the best local step in terms of fitness improvement [17]. After all the linkage sets were considered, the offspring is returned. We remark that the mixing performed by GOM is very different from the classic subtree-swapping crossover (or mutation), as the mixed nodes in GP-GOMEA are not necessarily connected, and it never generates an offspring less fit than the parent.

When all parent solutions underwent GOM, the offspring replace the population of parents, and the generation terminates.

**Algorithm 1** Gene-pool Optimal Mixing

```
function GOM(solution, sol_fit, LT)
    offspring ← solution; off_fit ← sol_fit
    backup ← solution; back_fit ← sol_fit
    RandomlyShuffle(LT)
    for $\mathrm{LT}_i \in \mathrm{LT}$ do
        donor ← RandomlyPickDonorFromPopulation()
        ReplaceNodesAtPositions(offspring, donor, $\mathrm{LT}_i$)
        if MeaningfulChangeExists(offspring, backup) then
            off_fit ← ComputeFitness(offspring)
            if IsEqualOrBetter(off_fit, back_fit) then
                backup ← offspring; back_fit ← off_fit
            else
                offspring ← backup; off_fit ← back_fit
        else
            backup ← offspring
    Return(offspring)
```

### 2.1 Adapting GP-GOMEA for real-world symbolic regression

To tackle real-world symbolic regression problems, we performed the following changes/additions to the core of GP-GOMEA.

**Linear scaling.** Since GP generates solutions by composing the nodes provided in the terminal and function set, a big enough solution size is needed to craft a specific function. Such solutions may become exorbitantly large if the right constants are not available, e.g., when trying to evolve the function $x^{100.0}$ using solely $+, \times, x, 1.0$ as nodes. To alleviate this issue, we adopt linear scaling [8]. Linear scaling is a computationally fast way to scale a solution $f(x)$ as $\alpha f(x) + \beta$ during error evaluation, so that the evolutionary search can focus on the shape of the dynamically scaled function. Given $n$ examples of features-target $(x, z)$ in the dataset, the slope $\alpha$ and the intercept $\beta$ can be efficiently calculated in $O(n)$

as:

$$\alpha = \bar{f}(x) - \beta \bar{z},$$
$$\beta = \frac{\sum_{i=1}^{n}(f(x_i) - \bar{f}(x))(z_i - \bar{z})}{\sum_{i=1}^{n}(z_i - \bar{z})^2},$$

with the overbar $\bar{\cdot}$ representing the mean.

**Interval arithmetic.** We further adopt interval arithmetic to evolve correct expressions without the need of using protected operators, as proposed in [8]. With interval arithmetic, any time a solution is generated (or changed), its validity is assessed by recursively propagating the interval of values that can be assumed by the tree nodes. If at any point, an operation with possible undefined values is encountered (e.g., division with second term having values in the interval $[-1, +1]$, which includes 0), then the solution is discarded and a new, random one is generated (or, in the case of variation, reverted to its valid state).

**Ephemeral random constant sampling.** Common practice in GP-based symbolic regression is to use an Ephemeral Random Constant terminal node (ERC), which value is sampled uniformly from a pre-specified interval when the node is instantiated. This interval is usually set by a rule of thumb, with $[-1, 1]$ being a typical choice for benchmarks. However, the dimensionality of the problem is not taken into account, and sampled constants may not help the search. Therefore, we initialize the interval dynamically at the beginning of the run, based on the values of the problem-specific features. Specifically, let $m$ features and $n$ examples be present in the training set. Let $x_{i,j}$ be the $i$th feature value of the $j$th entry of the training set, then we set the ERC sampling interval to $[\min x_{i,j}, \max x_{i,j}]$, $\forall i, j$.

**Convergence avoidance**. Like in genetic algorithms, the population of GP-GOMEA ultimately converges to the same genotype. This typically happens very quickly in GOMEA in general, because GOM prevents the generation of unfit offspring, and solutions have a fixed size. To avoid (premature) convergence, the worst third of the population is discarded at the end of each generation, and randomly generated anew.

## 3 MACHINE LEARNING ALGORITHMS FOR REGRESSION

The regression problems considered in this paper are defined with a dataset of examples. Each example contains values of the features and of the variable to regress. An ML algorithm uses this data to generate a model, i.e., a combination of the features, that estimates the target variable as closely as possible. Recall that, to assess if the model is capturing the correct feature combination, the original dataset is split into two separate parts. The training set is used to generate the model, and the test set to assess the model performance on unseen examples.

We consider two types of ML algorithms: evolutionary and non-evolutionary ones. The evolutionary ML algorithms considered are GP-GOMEA, Standard tree-based GP (SGP), SGP in a multi-objective formulation (SGP$_{\text{mo}}$), with size as secondary objective and implemented as NSGA-II [4], and a version of SGP forced to evolve small solutions by using the same maximum tree height of GP-GOMEA (SGP$_{\text{bounded}}$).

Note that SGP$_{\text{mo}}$ returns a Pareto set of solutions that do not dominate each other. To select a final solution from the Pareto set, the training set is further split before beginning the evolution, into so-called training-training set and training-test set (also known as validation set). The training-training set is used by SGP$_{\text{mo}}$ to find the Pareto set. In the end, the final solution is the one with minimum error on the training-test set. In other words, the training-test set is used to pick the solution that generalizes better.

Three non-evolutionary ML algorithms are considered: Linear Regression (LR), Support Vector Machines (SVM) for regression [2, 3], and Random Forest (RF) [1]. These algorithms are easy to use and are often effective on high-dimensional datasets, even with default parameters. Furthermore, they are relatively much faster to run than the evolutionary ML algorithms. LR is a deterministic method and models only linear combination of the features to regress the target variable. In this work we consider the most common form of LR, based on the least square error. SVM is also deterministic, but differently from LR uses the kernel trick method to express non-linear feature combinations. Here, the standard radial basis function kernel is used. RF is a stochastic, bagging ensemble algorithm that models non-linear feature combinations by means of an ensemble of regression trees. While LR-generated models are typically interpretable (mostly depending on the number of features), SVM, and RF-generated ones are much harder to interpret.

In the following, both evolutionary and non-evolutionary ML algorithms are used to directly perform regression. For the meta learning approach (explained below), GP-GOMEA is used as a feature constructor for the non-evolutionary ML algorithms, i.e., LM, SVM, and RF.

## 4 EVOLUTIONARY META LEARNING

Together with applying GP-GOMEA directly to symbolic regression, we consider an evolutionary meta learning approach where GP-GOMEA performs feature construction for a non-evolutionary ML algorithm. It works as follows. GP-GOMEA is run for $G$ generations, during which the population of solutions competes to become *one* new feature for the ML algorithm. Differently from the direct symbolic regression, here the fitness of a solution is calculated by running the ML algorithm on a training set where the feature represented by the solution is included. In an attempt to obtain a robust fitness for the solution, the ML algorithm is trained and tested $T$ times, each time on a different random split of the feature-enriched training set, and the maximum prediction error is considered. This process is shown in Algorithm 2. Note that, differently from the direct symbolic regression case, linear scaling is not applicable here, as it requires error residuals of the predicting solution, while now a feature is being evolved by considering the prediction error of an external ML algorithm.

Because evaluating the fitness is expensive, line 8 of Algorithm 1 is particularly important in order to prevent useless evaluations. No evaluation is performed if the mixing does not result in a syntactic change, nor in a *semantic* change. The latter check is an addition for the meta learning, done after a syntactic change is observed. A syntactic change may still lead to no semantic change, e.g., the swap of operands for + (commutativity). To check for semantic changes, the output of the changed solution is calculated (as done in symbolic regression in order to compute the error), and is compared with the output of the backup.

After $G$ generations, GP-GOMEA terminates and the best solution found becomes the new feature of the dataset. This whole process is iterated $K$ times: each time further enriching the training set with another evolved feature. Note that also the original test set is enriched. The final performance is given by training the ML algorithm on the enriched training set, and testing it on the enriched test set. We remark that, at any iteration, only the original features are used as input variables for the solutions, i.e., no previously evolved features are used to evolve new features. This way, evolved features can only be simple combinations of the original ones.

**Algorithm 2** Machine Learning run on Enriched Dataset

```
function RunMachineLearningAlgorithm(training, solution)
    max_error ← 0
    enriched_training ← EnrichDataset(training, solution)
    for $T$ times do
        training-training, training-test ← Split(enriched_training)
        $M$ ← TrainMLAlgorithm(training-training)
        error ← ComputePredictionError($M$, training-test)
        if error > max_error then
            max_error ← error
    Return(max_error)
```

## 5 MATCHING SIMILAR PATIENTS

Finding a model capable of pairing similar patients is important to improve the design of radiation treatments. For children who underwent radiotherapy before the advent of 3D treatment planning, coarse and limited data is known on what radiation dose they received. Now, childhood cancer survivors experience *late* adverse effects that may be explained by studying which radiation dose was absorbed by what organ subvolumes during treatment [6, 18]. Gaining this information is extremely valuable to design better treatments and reduce adverse effects. To acquire accurate dose estimations, *dose reconstruction* methods are nowadays performed, where a 3D anatomy (e.g., a virtual human model or a CT scan) is used to represent the body of the historical patient, and on which the historical radiation treatment is simulated [12]. Here, the first step of a CT scan-based dose reconstruction method is considered: the selection of a representative CT scan given the features of a historically treated patient. This extends our previous work [20].

We use anonymous data of 35 children from participating hospitals, to learn a distance notion modeling anatomical similarity among pediatric cancer patients. For validation purposes recent patients are used, however only features that are normally available in historical records are considered. These features are shown in Table 1. For example, measurements are taken from 2D digitally reconstructed radiographs, which are generated from the 3D CT scans. Because most of the patients were treated for Wilms' tumor (i.e., a tumor of the kidney), the abdomen is taken as region of interest on the CT scans, specifically from the top of the 10th thoracic vertebra to the bottom of the 1st sacral vertebra.

We consider two notions of anatomical distance, previously introduced in [20]: a distance based on the effort needed to register one CT scan to another, and a distance based on the overlap of 3D abdomen contours.

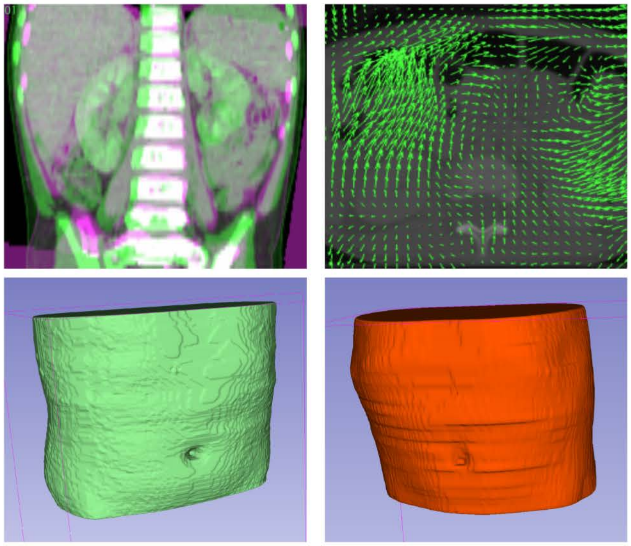

**Figure 1: Top-left: two rigidly aligned CT scans in coronal view, one in green and one in purple (aligned voxels are gray values). Top-right: deformation vector field defining a registration, in axial view. Bottom: Examples of 3D adbomen countours used for the computation of the overlap-based distance.**

### 5.1 Distance based on CT scan registration

This distance is based on the effort needed to perform a deformable image registration of CT scans. In particular, the software elastix [9] was used to register the CT scans with each other (after a first manual alignment based on bony anatomy) using deformable b-spline registration. The registration parameters are the following (not listed ones are set to default): adaptive stochastic gradient descent optimizer; Mattes' mutual information metric; multi-resolution b-spline transformation; 28mm control points grid spacing. A measure of deformation effort is derived from the registration outcomes, by looking at the 3D displacement of b-spline control points. Specifically, the deformation effort is defined as:

$$\sum_{p\in P} \frac{\sum_{s\in S_p} ||\vec{v}_p - \vec{v}_s||}{|S_p|},$$

with $P$ the 3D set of b-spline control points, $S_p$ the set containing $p$ and the control points adjacent to $p$, and $\vec{v}_p$ the 3D vector corresponding to point $p$ that specifies the 3D offset the transformation applies to the control point (similarly, $\vec{v}_s$ is the 3D vector specifying the 3D offset of point $s$). Image shrinking/stretching effects contribute to the sum, while translations do not.

Since deformable registrations are asymmetrical, the deformation efforts are computed by registering the images in both directions, and the mean is taken. Figure 1 shows an example of rigidly aligned CT scans and deformation vector field.

**Table 1: Pediatric patient features used in this work.**

| Numerical features | Mean | St. dev. | Min | Max | ID |
|---|---|---|---|---|---|
| Age (years) | 3.90 | 1.05 | 2.21 | 5.56 | $x_0$ |
| Anterior-posterior diameter measured at isocenter (cm) | 13.36 | 1.31 | 11.30 | 16.00 | $x_1$ |
| Distance iliac crest-spinal cord (cm) | 5.58 | 0.56 | 4.34 | 6.75 | $x_2$ |
| Hearth diameter (cm) | 8.40 | 0.79 | 6.80 | 9.85 | $x_3$ |
| Height (cm) | 104.41 | 9.74 | 89.00 | 123.00 | $x_4$ |
| Left-right diameter measured at 2nd lumbar vertebra (cm) | 19.51 | 1.56 | 16.30 | 23.50 | $x_5$ |
| Length right diaphragm (cm) | 8.38 | 0.71 | 7.10 | 9.76 | $x_6$ |
| Length spinal cord from 12th thoracic to 4th lumbar vertebra (cm) | 9.33 | 0.91 | 7.00 | 10.90 | $x_7$ |
| Weight (kg) | 16.85 | 3.85 | 10.00 | 28.00 | $x_8$ |

| Categorical features | Values | ID |
|---|---|---|
| Gender | 2 categories: 19 females, 16 males | $x_9$ |
| Diagnosis | 6 categories: 21 Wilms' tumor, 14 other | $x_{10}$ |
| Partial nephrectomy | 3 categories: 2 left, 1 right, 32 none | $x_{11}$ |
| Radical nephrectomy | 3 categories: 10 left, 10 right, 15 none | $x_{12}$ |
| Tumor site | 10 categories: 10 left kidney, 11 right kidney, 14 other | $x_{13}$ |

### 5.2 Distance based on abdomen overlap

Another notion of distance is based on the overlap of the abdomens that were manually segmented from the CT scans. Two examples of abdominal volumes are shown in Figure 1. This distance measure is computed after alignment of the given volumes $V1$, $V2$ on the center of mass, using the Dice Similarity Coefficient (DSC), which is defined as:

$$\text{DSC}(V_1, V_2) = \frac{|V_1 \bigcap V_2|}{|V_1| + |V_2|}.$$

The distance is then simply $100(1 - DSC)$.

### 5.3 Datasets of patient similarity

For both distances, a dataset was generated where each row represents a pairing of patients. The predictor variables contained in a row are the features of the paired patients, listed one next to the other (i.e., given patients $x, y$, one feature is $x_{\text{age}}$ and another is $y_{\text{age}}$). Consecutive integer numbers are used for categorical features, e.g., female = 0, male = 1 for gender, and left = -1, none = 0, right = 1 for radical nephrectomy. The task of combining these features is left to the ML algorithm (feature relevance and selection is outside the scope of this work). As to the target variable, in the dataset $D_{\text{Deform}}$ the CT deformation-based distance is used, while in the dataset $D_{\text{Overlap}}$ the target variable is the abdominal overlap-based distance. The two datasets are thus each composed of $14 \times 2 = 28$ features, 1 target variable, and $\binom{35}{2} = 595$ examples.

## 6 EXPERIMENTAL SETUP

We ran all the ML algorithms directly on the datasets, and the non-evolutionary ML algorithms embedded in the meta-learning approach, with varying number of iterations $K$ to construct new features. Together with the two regression problems on patient anatomical distance, two further well-known real-wold benchmark datasets are considered[1], namely Boston housing (13 features and 506 examples) and Servo (19 features and 167 examples).

For both the direct regression and the evolutionary meta learning, all experiments consisted of 30 independent runs, each with a random split of the dataset. A 70-30 split is used to partition the examples into training set and test set.

### 6.1 Direct regression

For the direct regression, non-evolutionary ML algorithms are run with default parameters. As to the evolutionary algorithms, a time limit of 1 hour is set. The parameter settings used for GP-GOMEA and SGP are reported in Table 2. Both methods use interval arithmetics and linear scaling. In the variation phase of SGP, whenever crossover, mutation, and reproduction do not happen, a new solution is generated, either with the full or grow method (50-50 chance) [10]. The node selected for subtree variation in crossover and mutation is chosen with the uniform depth node selection method [13], which better prevents bloat. In SGP$_{\text{mo}}$, the initial training test is split into training-training and training-test with a 70-30 split. SGP$_{\text{bounded}}$ uses the same maximum tree height of GP-GOMEA, i.e., 5. Lastly, both GP-GOMEA and SGP use the caching of node outputs (i.e., partial evaluations) to speed up solution evaluation time [13, 19].

### 6.2 Evolutionary meta learning

For the meta learning approach, the parameter settings are the following. GP-GOMEA is run for 10 generations, with a population size of 120 solutions and maximum tree height 2. From 1 to 10 meta learning iterations are done. During the evaluation of a solution (Algorithm 2), the number $T$ of repetitions of the ML algorithm on different train set splittings is set to 10 (and 100 trees are used for RF), and 70-30 split of the feature-enriched training set is used to generate the training-training and training-test set. To evaluate the final performance, the ML algorithm is run on the test set, enriched with the new features, and the test error is considered. For RF, which is stochastic, 30 repetitions are done for the test, each using 2000 trees, and the average error is returned.

[1] Boston housing and Servo are available on the UCI Machine Learning Repository website: http://archive.ics.uci.edu/ml.

**Table 2: Parameter settings for GP-GOMEA and SGP**

| Parameter | Value |
|---|---|
| Time limit | 1 hour |
| Population size | 2000 |
| Terminal set | features + ERC |
| Function set | $\{+, -, \times, \div, \cdot^2, \exp, \log\}$ |
| Tree initialization method | half-n-half |
| Minimum tree height at initialization | 2 |
| GP-GOMEA fixed maximum tree height | 5 |
| SGP maximum tree height at initialization | 5 |
| SGP maximum tree height during evolution | 17 (5 for SGP$_{\text{bounded}}$) |
| SGP selection method | tournament of size 7 |
| SGP crossover/mutation/reproduction rate | 0.9/0.1/0.1 |
| SGP elitism | 1 best of generation |
| SGP new solution sampling | if cross., mut., and repr. do not happen |
| GP-GOMEA new solution sampling | replace worst 3rd pop. |

The code for GP-GOMEA and for the variants of SGP was implemented in C++. The meta learning was made possible by interfacing GP-GOMEA with existing implementations of the non-evolutionary ML algorithms for R [2, 14]. The ML algorithms have been used with default parameter settings, as parameter tuning is outside the scope of this work. Experiments were run on a machine with 2 Intel® Xeon® CPU E5-2699 v4 @ 2.20GHz and 630 GB of RAM.

## 7 RESULTS

The results are presented separately for the direct regression, and the evolutionary meta learning approach. The coefficient of determination $R^2 = 1 - \sum(\hat{z} - z)^2/\text{var}(z)$, with $z$ the variable to regress and $\hat{z}$ the model prediction, is used to measure the performance.

### 7.1 Direct regression

The $R^2$ obtained by the direct application of all ML algorithms on the four datasets is reported in Table 3. It can be seen that, overall, RF is the significantly best performing algorithm on both train and test for 3 out of 4 datasets, according to the unpaired two-samples Wilcoxon test (p-value < 0.05). SVM performs second best, however both RF and SVM are black-boxes. The performance of LR is poor in all cases. As to the evolutionary ML algorithms, GP-GOMEA reaches lower $R^2$ compared to SGP and SGP$_{\text{mo}}$ on the training, however the latter algorithms tend to overfit, as some solutions reach extremely large errors on the test set. In SGP$_{\text{mo}}$, the use of size as second objective and an intermediate validation step lowers the chance of extremely bad performance on unseen data, yet does not improve median performance. SGP$_{\text{bounded}}$, which uses the same maximum tree height of GP-GOMEA, performs significantly worse than the latter on both training and testing. This last result shows that the model-based variation performed by GOM with the LT is more competent than the blind variation operators of SGP.

While the maximum solution size (i.e., tree nodes) reachable by GP-GOMEA is 63 (given by maximum tree height $h = 5$ and maximum function arity 2), the solutions found are typically half this size (by counting active nodes), making them effectively inspectable. Figure 3 shows models found by GP-GOMEA for the regression of patient distance with median test performance. It can be seen that the model for $D_{\text{Deform}}$ is a linearly-scaled logarithm of the sum between the squared difference of left-right diameters, and a ratio involving four types of features. This last term is not immediately readable, and could be overfitting the data. Pre-processing the data with, e.g., feature selection, may improve the interpretability and performance of the evolved models. Like for GP-GOMEA, the models found by SGP$_{\text{bounded}}$, and to some extent the ones found by LR, can also be inspected; however they perform worse.

Additional experiments (not reported here) showed that increasing the maximum height for GP-GOMEA trees and enriching the function set (e.g., with the *sin* and *cos* functions) leads to improved performance, however solutions become much harder to interpret.

### 7.2 Evolutionary meta learning

As to the evolutionary meta learning, Figure 2 shows boxplots representing the change in $R^2$ on the test set that is obtained by adding up to 10 extra features to LM, SVM, and RF. The bottom and top of a box are the lower and upper quartile ($LQ, UQ$), respectively. The band near the middle of a box is the median. The lower and upper whiskers are computed as $\max(\min(R^2), LQ + 1.5(UQ - LR))$ and $\min(\max(R^2), UQ + 1.5(UQ - LR))$, respectively. Circles are outliers. Note that when no evolved features are added, the performance is the same as reported in Table 3.

In almost all cases, it can be seen that iteratively adding evolved features slightly, yet steadily improves performance, up to a point where no further improvements are observed. Also, it is remarkable to notice that mean performance never becomes worse, although variation may increase. The most dramatic performance increase is obtained for the evolutionary meta learning with LR (eLR), which $R^2$ on the test set can increase up to four times ($D_{\text{Overlap}}$). For the evolutionary meta learning with SVM (eSVM), performance increase is more moderate than for eLR, but is present on all problems. The evolutionary meta learning with RF (eRF) is better than RF on $D_{\text{Overlap}}$ and Servo, while is similar on $D_{\text{Deform}}$ and Boston. RF seems alone capable of effectively combining the original features, although it is very hard to interpret how RF combines them. Table 4 summarizes statistical superiority on the test set of the ML algorithms, as directly applied to the datasets, and with the addition of 10 features in the meta learning. The best algorithm on $D_{\text{Deform}}$ and $D_{\text{Overlap}}$ is eSVM, with a mean $R^2$ of 0.81 (st. dev. 0.02) and of 0.94 (st. dev. 0.01), respectively. Boston is best solved by RF and eRF. On Servo, surprisingly, eLR performs as good as eRF (mean test $R^2$ 0.84, st. dev. 0.09 for the first, mean test $R^2$ 0.87, st. dev. 0.07 for the second), both methods being significantly superior to the others.

The first (median) features evolved by GP-GOMEA are shown in Table 5 for the patient distance regression problems. Despite their simplicity, these features are typically responsible for a large performance improvement (particularly for eLR). It can be seen that the features found by GP-GOMEA are pseudo-distances, consisting of non-linear interaction between the anterior-posterior and left-right abdominal diameters, and the weight of two patients. As an example, the feature found for eLR on $D_{\text{Deform}}$ is $(x_5 - y_5)\frac{x_8}{y_8}$, i.e., the difference of left-right diameters times the ratio of the weights. This simple feature allows LR to improve its $R^2$ by almost a factor of 4, while being extremely easy to interpret.

**Table 3: Mean $R^2$ (and standard deviation) obtained by direct application of the ML algorithms on the datasets. Statistically significant superior performances are reported in bold. For the test performance of SGP and SGP$_{\text{mo}}$, the median is reported (indicated by *), because of the large variations found. The size of LR models is estimated as the number of nodes for an equivalent GP tree with all coefficient values different from 0. No size is reported for SVM and RF.**

| Training | GP-GOMEA | SGP | SGP$_{\text{mo}}$ | SGP$_{\text{bounded}}$ | LR | SVM | RF |
|---|---|---|---|---|---|---|---|
| $D_{\text{Deform}}$ | 0.67 0.05 | 0.79 0.08 | 0.69 0.14 | 0.60 0.06 | 0.24 0.09 | 0.89 0.01 | **0.95 0.00** |
| $D_{\text{Overlap}}$ | 0.80 0.07 | 0.91 0.03 | 0.83 0.11 | 0.71 0.04 | 0.27 0.10 | 0.96 0.00 | **0.97 0.00** |
| Boston | 0.80 0.05 | 0.93 0.02 | 0.89 0.07 | 0.78 0.04 | 0.75 0.04 | 0.88 0.02 | **0.97 0.00** |
| Servo | 0.94 0.04 | **0.99 0.01** | 0.94 0.07 | 0.89 0.06 | 0.78 0.06 | 0.90 0.03 | 0.93 0.02 |
| Testing | GP-GOMEA | SGP | SGP$_{\text{mo}}$ | SGP$_{\text{bounded}}$ | LR | SVM | RF |
| $D_{\text{Deform}}$ | 0.63 0.05 | 0.58* Inf | 0.55* 928.43 | 0.57 0.07 | 0.16 0.07 | 0.76 0.03 | **0.80 0.03** |
| $D_{\text{Overlap}}$ | 0.77 0.08 | 0.79* Inf | 0.68* 0.41 | 0.69 0.05 | 0.19 0.09 | **0.90 0.02** | 0.89 0.01 |
| Boston | 0.76 0.05 | 0.74* $3e^{25}$ | 0.75* 52.49 | 0.73 0.06 | 0.71 0.03 | 0.82 0.04 | **0.87 0.03** |
| Servo | **0.79 0.12** | 0.09* Inf | 0.74* 2126.12 | 0.75 0.11 | 0.70 0.07 | **0.78 0.07** | **0.81 0.07** |
| Solution size | GP-GOMEA | SGP | SGP$_{\text{mo}}$ | SGP$_{\text{bounded}}$ | LR | SVM | RF |
| $D_{\text{Deform}}$ | 27.70 6.85 | 670.83 154.39 | 419.37 107.49 | **21.60 7.41** | 113.00 0.00 | | |
| $D_{\text{Overlap}}$ | 34.97 13.52 | 845.40 226.44 | 536.57 149.45 | **31.57 7.78** | 113.00 0.00 | | |
| Boston | 24.27 12.28 | 684.17 132.05 | 546.20 120.85 | 24.43 8.11 | 53.00 0.00 | | |
| Servo | 38.23 11.36 | 764.07 132.63 | 464.13 87.34 | 36.00 7.35 | 77.00 0.00 | | |

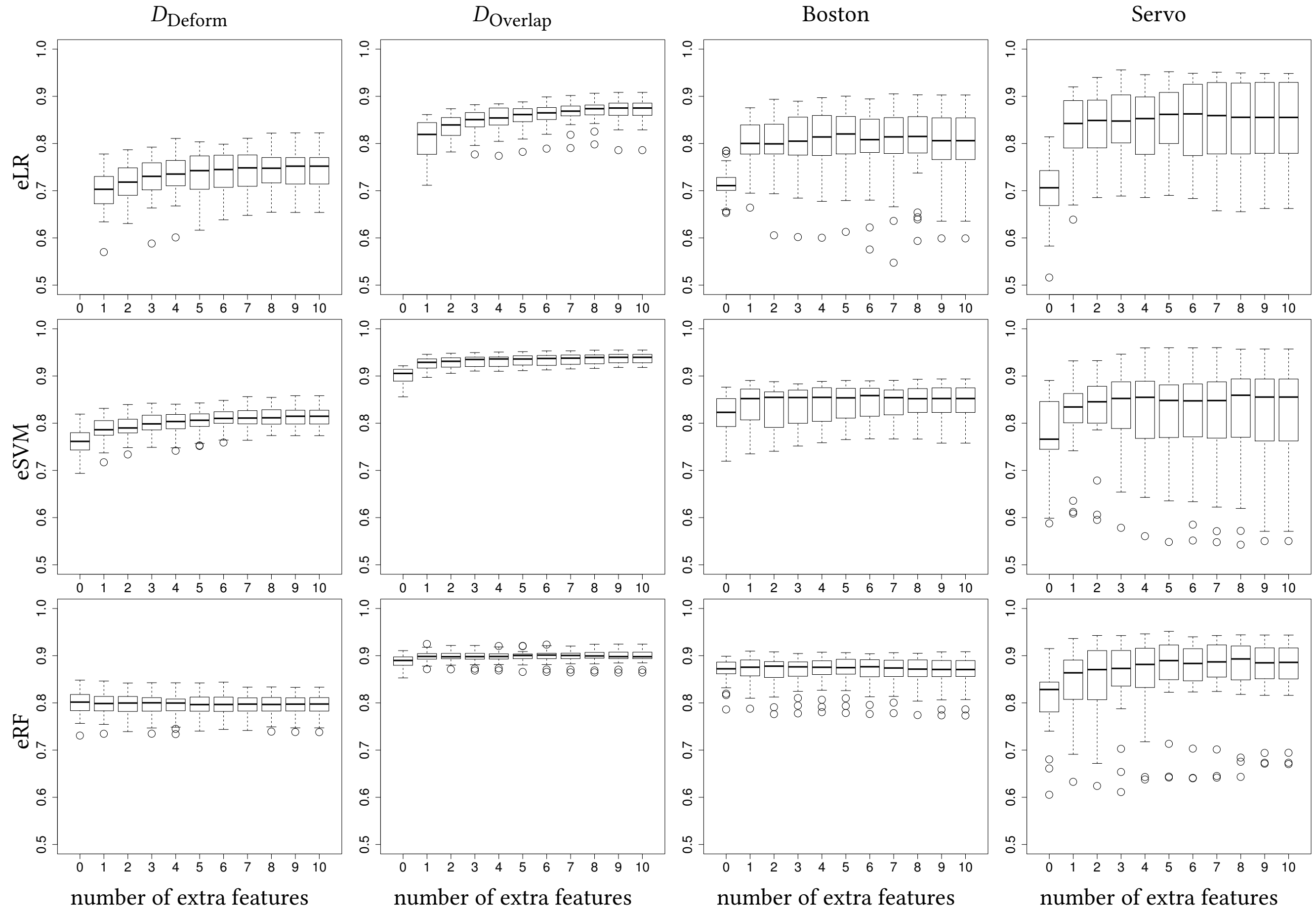

**Figure 2: Boxplots representing test $R^2$ with different number of evolved features. The boxplots of LR (no extra features) are too low to be displayed for $D_{\text{Deform}}$ and $D_{\text{Overlap}}$ (mean of 0.16 and 0.19, respectively).**

**Table 4: Statistical significance results on the four datasets. The ≻ (≺) symbol represents significant superiority (inferiority) of the row element against the column element.**

| | $S_{\text{Deform}}$ | | | | | | $S_{\text{Overlap}}$ | | | | | | Boston | | | | | | Servo | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | LR | eLR | SVM | eSVM | RF | eRF | LR | eLR | SVM | eSVM | RF | eRF | LR | eLR | SVM | eSVM | RF | eRF | LR | eLR | SVM | eSVM | RF | eRF |
| LR | | ≺ | ≺ | ≺ | ≺ | ≺ | | ≺ | ≺ | ≺ | ≺ | ≺ | | ≺ | ≺ | ≺ | ≺ | ≺ | | ≺ | ≺ | ≺ | ≺ | ≺ |
| eLR | ≻ | | | ≺ | ≺ | ≺ | ≻ | | ≺ | ≺ | ≺ | ≺ | ≻ | | | ≺ | ≺ | ≺ | ≻ | | ≻ | | ≻ | |
| SVM | ≻ | | | ≺ | ≺ | ≺ | ≻ | ≻ | | ≺ | ≻ | | ≻ | | | ≺ | ≺ | ≺ | ≻ | ≺ | | ≺ | | ≺ |
| eSVM | ≻ | ≻ | ≻ | | ≻ | ≻ | ≻ | ≻ | ≻ | | ≻ | ≻ | ≻ | ≻ | ≻ | | ≺ | ≺ | ≻ | | ≻ | | | ≺ |
| RF | ≻ | ≻ | ≻ | ≺ | | | ≻ | ≻ | ≺ | ≺ | | ≺ | ≻ | ≻ | ≻ | ≻ | | | ≻ | ≺ | | | | ≺ |
| eRF | ≻ | ≻ | ≻ | ≺ | | | ≻ | ≻ | | ≺ | ≻ | | ≻ | ≻ | ≻ | ≻ | | | ≻ | | ≻ | ≻ | ≻ | |

$D_{\text{Deform}}$

$$-0.608323 + 0.792302 \log\left((x_5 - y_5)^2 + y_2 \frac{y_1(1+x_1)-x_8}{x_6(y_2+y_8)}\right)$$

$D_{\text{Overlap}}$

$$11.515279 + 0.874599\left(\left(\log(y_0) - \log(x_0) + y_7 \frac{y_8 + y_1}{x_4}\right)^2 + \log(y_5)^2 - \frac{(x_5)^2}{y_5 + y_1}\right)$$

**Figure 3: Models with median test performance on $D_{\text{Deform}}$ and $D_{\text{Overlap}}$ learned by GP-GOMEA. Features $x_i, y_i$ refer to the first and second patient in the pair, respectively (see Table 1 for the meaning of $x_i$).**

**Table 5: First feature learned by GP-GOMEA in the evolutionary meta learning approach which lead to median test performance on $D_{\text{Deform}}$ and $D_{\text{Overlap}}$. Feature $x_i$ ($y_i$) represents the $i$th feature of the first (second) patient.**

| | eLR | eSVM | eRF |
|---|---|---|---|
| $D_{\text{Deform}}$ | $\frac{x_5-y_5}{y_8/x_8}$ | $(x_5 - y_5)(x_5 - y_1)$ | $(x_5/y_5)(y_0 - x_5)$ |
| $D_{\text{Overlap}}$ | $\frac{x_1-y_5}{x_5/y_1}$ | $(y_1 - x_5)(y_5 - x_1)$ | $(y_8 - x_8)(y_1 - x_1)$ |

## 8 CONCLUSION

The GP version of the Gene-pool Optimal Mixing Evolutionary Algorithm (GP-GOMEA) generally evolves smaller solutions than other GP algorithms without compromising on accuracy. We tested this property on the domain of symbolic regression, in order to obtain accurate and readable mathematical expressions. Our experimental results confirm that GP-GOMEA finds similarly or more accurate models than three variants of standard GP, and is less prone to overfitting. On a clinical problem, where a model capturing anatomical dissimilarity needs to be found, we showed that the models found by GP-GOMEA are interpretable to some degree.

We furthermore explored the possibility to use GP-GOMEA in a meta learning approach, making the EA evolve very small but salient features to use by a machine learning algorithm for regression. Almost always, this approach resulted in statistically superior prediction performance, and in no case performance deteriorated. At the same time, the features found were extremely simple, and easy to interpret. In conclusion, GP-GOMEA proves to be an excellent EA for symbolic regression, and a powerful feature constructor.

## 9 ACKNOWLEDGMENTS

The authors acknowledge the Kinderen Kankervrij foundation for financial support (project #187), and the Maurits and Anna de Kock foundation for financing a high performance computing system.

## REFERENCES

[1] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[2] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2 (2011), 27:1–27:27. Issue 3. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[3] Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning* 20, 3 (1995), 273–297.

[4] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (2002), 182–197.

[5] Alex A. Freitas. 2010. *A Review of Evolutionary Algorithms for Data Mining.* Springer US, Boston, MA, 371–400.

[6] Mathilde C. Geenen, Maud M .and Cardous-Ubbink, Leontien C.M. Kremer, Cor van den Bos, Helena J.H. van der Pal, Richard C Heinen, Monique W.M. Jaspers, Caro C.E. Koning, Foppe Oldenburger, Nelia E. Langeveld, et al. 2007. Medical assessment of adverse health outcomes in long-term survivors of childhood cancer. *Jama* 297, 24 (2007), 2705–2715.

[7] Marti A. Hearst, Susan T. Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. 1998. Support vector machines. *IEEE Intelligent Systems and their Applications* 13, 4 (1998), 18–28.

[8] Maarten Keijzer. 2003. Improving symbolic regression with interval arithmetic and linear scaling. *Genetic Programming* (2003), 275–299.

[9] Stefan Klein, Marius Staring, Keelin Murphy, Max A. Viergever, and Josien P.W. Pluim. 2010. Elastix: a toolbox for intensity-based medical image registration. *IEEE Transactions on Medical Imaging* 29, 1 (2010), 196–205.

[10] John R. Koza. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* MIT Press, Cambridge, MA, USA.

[11] Sean Luke and Liviu Panait. 2006. A comparison of bloat control methods for genetic programming. *Evolutionary Computation* 14, 3 (2006), 309–344.

[12] Angela Ng, Kristy K. Brock, Michael B. Sharpe, Joanne L. Moseley, Tim Craig, and David C. Hodgson. 2012. Individualized 3D reconstruction of normal tissue dose for patients with long-term follow-up: a step toward understanding dose risk for late toxicity. *International Journal of Radiation Oncology·Biology·Physics* 84, 4 (2012), e557–e563.

[13] Tomasz P. Pawlak, Bartosz Wieloch, and Krzysztof Krawiec. 2015. Semantic backpropagation for designing search operators in genetic programming. *IEEE Transactions on Evolutionary Computation* 19, 3 (2015), 326–340.

[14] R Development Core Team. 2008. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. http://www.R-project.org ISBN 3-900051-07-0.

[15] Michael Schmidt and Hod Lipson. 2009. Distilling free-form natural laws from experimental data. *Science* 324, 5923 (2009), 81–85.

[16] Dirk Thierens. 2010. The linkage tree genetic algorithm. In *International Conference on Parallel Problem Solving from Nature*. Springer, 264–273.

[17] Dirk Thierens and Peter A.N. Bosman. 2011. Optimal mixing evolutionary algorithms. In *Proceedings of the 13th annual Conference on Genetic and Evolutionary Computation.* ACM, 617–624.

[18] Irma W.E.M. van Dijk, Foppe Oldenburger, Mathilde C. Cardous-Ubbink, Maud M. Geenen, Richard C. Heinen, Jan de Kraker, Flora E. van Leeuwen, Helena J.H. van der Pal, Huib N. Caron, Caro C.E. Koning, and Leontien C.M. Kremer. 2010. Evaluation of late adverse events in long-term Wilms' tumor survivors. *International Journal of Radiation Oncology·Biology·Physics* 78, 2 (2010), 370 – 378.

[19] Marco Virgolin, Tanja Alderliesten, Cees Witteveen, and Peter A.N. Bosman. 2017. Scalable genetic programming by gene-pool optimal mixing and input-space entropy-based building-block learning. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '17).* ACM, New York, NY, USA, 1041–1048.

[20] Marco Virgolin, Irma W. E. M. van Dijk, Jan Wiersma, Cécile M. Ronckers, Cees Witteveen, Arjan Bel, Tanja Alderliesten, and Peter A. N. Bosman. 2018. On the feasibility of automatically selecting similar patients in highly individualized radiotherapy dose reconstruction for historic data of pediatric cancer survivors. *Medical Physics* (2018). https://doi.org/10.1002/mp.12802