EGSGP: an Ensemble System Based on Geometric Semantic Genetic Programming

Liah Rosenfeld and Leonardo Vanneschi

NOVA Information Management School (NOVA IMS), Universidade Nova de Lisboa, Campus de Campolide, 1070-312 Lisboa, Portugal

{lrosenfeld,lvanneschi}@novaims.unl.pt

This is the Author Peer Reviewed version of the following conference paper published by Springer:

Rosenfeld, L., Vanneschi, L. (2023). EGSGP: An Ensemble System Based on Geometric Semantic Genetic Programming. In: De Stefano, C., Fontanella, F., Vanneschi, L. (eds) Artificial Life and Evolutionary Computation. WIVACE 2022. Communications in Computer and Information Science, vol 1780. Springer, Cham. https://doi.org/10.1007/978-3-031-31183-3_23

EGSGP: an Ensemble System Based on Geometric Semantic Genetic Programming

Liah Rosenfeld and Leonardo Vanneschi

NOVA Information Management School (NOVA IMS), Universidade Nova de Lisboa,
Campus de Campolide, 1070-312 Lisboa, Portugal
{lrosenfeld,lvanneschi}@novaims.unl.pt

Abstract. This work is inspired by the idea of seeding Genetic Programming (GP) populations with trained models from a pool of different Machine Learning (ML) methods, instead of using randomly generated individuals. If one considers standard GP, tackling this problem is very challenging, because each ML method uses its own representation, typically very different from the others. However, the task becomes easier if we use Geometric Semantic GP (GSGP). In fact, GSGP allows us to abstract from the representation, focusing purely on semantics. Following this idea, we introduce EGSGP, a novel method that can be seen either as a new initialization technique for GSGP, or as an ensemble method, that uses GSGP to combine different Base Learners (BLs). To counteract overfitting, we focused on the study of elitism and Soft Target (ST) regularization, studying several variants of EGSGP. In particular, systems that use or do not use elitism, and that use (with different parameters) or do not use ST were investigated. After an intensive study of the new parameters that characterize EGSGP, those variants were compared with the used BLs and with GSGP on three real-life regression problems. The presented results indicate that EGSGP outperforms the BLs and GSGP on all the studied test problems. While the difference between EGSGP and GSGP is statistically significant on two of the three test problems, EGSGP outperforms all the BLs in a statistically significant way only on one of them.

Keywords: Genetic Programming · Geometric Semantic Genetic Programming · Initialization · Ensemble methods · Meta-learning

1 Introduction

Ensemble Methods (EMs) focus on combining the knowledge obtained from multiple models resulting from different algorithms, with the goal of achieving a final model that is better than any of its singular components [1]. More specifically, EMs form their final models by constructing an ensemble, which is simply a bundle of learners – commonly known as Base Learners (BLs) – whose individual decisions are combined in some way [2]. EMs are therefore anchored in the Aristotelian idea that the whole should be better than the sum of its parts, relying on the "power of combination" to achieve a more accurate final result [1]. Meta-learning is a specific field within EMs that, as its name indicates, focuses on raising the level of abstraction a step further, seeking for the ability of "learning to learn" [3]. More precisely, while a ML algorithm (commonly denoted as

the first-level or base learner) learns to identify patterns in the data in order to make predictions, a meta-learning algorithm (often denoted as the second-level learner) learns how to combine these first-level predictions [1]. Hence, in meta-learning an algorithm learns from previous experience in a data-driven way [4] by taking the output of different base learners as input [5].

This paper proposes Ensembled Geometric Semantic Genetic Programming (EGSGP) as a meta-learning technique, that uses Geometric Semantic Genetic Programming (GSGP) to combine the information obtained from a heterogeneous set of ML algorithms, with the goal of using the power of combination for a better predictive performance. GSGP [6] is an extension of Genetic Programming (GP) [7], that has the goal of evolving individuals while directly searching in the semantic space [8]. In this context, the semantic of an individual refers to its predictions on the training data or, in other words, the vector of its output values [9]. This ability to directly search in the semantic space is possible thanks to the replacement of standard crossover and mutation, typically based on individuals' syntax, with geometric semantic operators (GSOs) [10]. GSOs bestow GSGP with the property of inducing a unimodal error surface [10] (i.e., a fitness landscape characterized by the lack of locally optimal solutions). Besides the introduction of GSOs, several other contributions focused on facilitating GP's search using semantic awareness [8]. For instance, Pawlak and Kraweic introduced semantic geometric initialization [11], opening gates to other GSGP focused initialization methods (e.g., the EDDA initialization [12]), recognizing the importance of an appropriate population initialization and how it may play a crucial role for GSGP.

The proposed technique (EGSGP) is contextualized in this research track. In fact, it can be seen as an ensemble initialization method that utilizes the outputs obtained from a heterogeneous set of ML algorithms, the BLs, as the initial population for GSGP. This is possible because, as recognized in [6], GSGP can abstract from representation, i.e. from the genotype of the individuals, focusing solely on their semantics, a property that distinguishes GSGP from traditional GP. So, even though the different BLs have very diverse representations, the evolution can be based only on their output values, calculated on the training observations. One can perceive EGSGP as either a novel initialization method for GSGP, where the initial population is no longer composed of random individuals, but rather of the predictions (i.e., the semantics) of the BLs, or as a meta-estimator that utilizes GSGP as a trainable combiner (i.e., the entity responsible for combining the knowledge obtained from the different models [13]). Additionally, an analysis of the effects of elitism and of the incorporation of a regularization method called Soft Target (ST) [14] in the models, with the goal of limiting overfitting and dealing with potential diversity issues, was performed.

This document is organized as follows: in Section 2, we present GSGP; Section 3 introduces ST regularization; Section 4 discusses the proposed EGSGP technique; Section 5 presents our experimental study, including a discussion of the test cases, of the used experimental settings and of the obtained results; finally, Section 6 concludes the work and proposes ideas for future research.

2 Geometric Semantic Genetic Programming

Let $X = \{\overrightarrow{x_1}, \overrightarrow{x_2}, ..., \overrightarrow{x_n}\}$ be the set of input data (training instances, observations or fitness cases) of a supervised learning problem, and $\overrightarrow{t} = [t_1, t_2, ..., t_n]$ the vector of the respective expected output or target values (in other words, for each $i = 1, 2, ..., n$, t_i is the expected output corresponding to input $\overrightarrow{x_i}$). A GP individual (or program) P can be seen as a function that, for each input vector $\overrightarrow{x_i}$ returns the scalar value $P(\overrightarrow{x_i})$. Following [6], we call *semantics* of P the vector $\overrightarrow{s_P} = [P(\overrightarrow{x_1}), P(\overrightarrow{x_2}), ..., P(\overrightarrow{x_n})]$. This vector can be represented as a point in a n-dimensional space, the *semantic space*. As explained above, GSGP is a variant of GP, where the traditional crossover and mutation are replaced by new operators, the Geometric Semantic Operators (GSOs). The objective of GSOs is to define modifications on the syntax of GP individuals, that have a precise effect on their semantics. One of the reasons for the success of GSOs is because they induce an unimodal error surface (on training data) for any supervised learning problem, where fitness is calculated using an error measure between outputs and targets. In other words, using GSOs the error surface on training data is guaranteed to not have any locally optimal solution, except for the global optima. The definitions of the GSOs are, as given in [6], respectively:

Geometric Semantic Crossover (GSC). Given two parent functions $T_1, T_2 : R^n \rightarrow R$, the geometric semantic crossover returns the real function $T_{XO} = (T_1 \cdot T_R) + ((1 - T_R) \cdot T_2)$, where T_R is a random real function whose output values range in the interval $[0, 1]$.

Geometric Semantic Mutation (GSM). Given a parent function $T : R^n \rightarrow R$, the geometric semantic mutation with mutation step ms returns the real function $T_M = T + ms \cdot (T_{R1} - T_{R2})$, where T_{R1} and T_{R2} are random real functions.

Even though this is not in the original definition of GSM, later contributions [10] have clearly shown that limiting the codomain of T_{R1} and T_{R2} in a predefined interval (for instace $[0, 1]$, as it is done for T_R in GSC) helps improving the generalization ability of GSGP. As in several previous works [10], here we constrain the outputs of T_R, T_{R1} and T_{R2} by wrapping them in a logistic function. Only the definitions of the GSOs for symbolic regression problems are given here, since they are the only ones used in this work. For the definition of GSOs for other domains, the reader is referred to [6].

Reference [10] contains a detailed and intuitive explanation of the reason why the error surface is unimodal when GSOs are used. However, even without dwelling with the formal details, that can be found in that paper, one can have an intuition of why GSO induce a unimodal error surface, looking at the GSM definition. GSM corresponds to box mutation in the semantic space, which means that if we mutate an individual x, it is possible to create an individual with any semantic included in a box centered in the semantic of x. As such, and recalling that also the target vector is a point in the semantic space, with a GSM step

applied to any x it is always possible to get closer than x to the global optimum. One important notation is that, even though having a unimodal error surface significantly simplifies the learning process, it does not "trivialize" the problem. If fact, it is important to understand that we are facing supervised machine learning problems here, and not pure optimization problems. As such, what is interesting for us is the model accuracy on *unseen* data, while the unimodality of the error surface can be proven only on training data.

Furthermore, using GSOs the semantics of the offspring is completely defined by the semantics of the parents: the semantics of an offspring produced by GSC will lie on the segment joining the semantics of the parents (geometric crossover), while GSM defines a mutation such that the semantics of the offspring lies within the ball of radius ms centered in the semantics of the parent (geometric mutation, or "ball mutation"). As discussed in [6, 10], GSOs always generate larger offspring than the parents, and this entails a rapid growth of the size of the individuals in the population. To counteract this problem, in [10] an implementation of GSOs is presented that makes GSGP not only usable in practice, but also significantly faster than standard GP. This is possible by means of a particular representation of GP individuals, that allows us to not store their genotypes during the evolution. This point, i.e. not having to store the genotype of the evolving individuals, is key for our work, since the initial individuals are models coming from the training of very diverse BLs. So, the implementation presented in [10] is the one we used here.

3 Soft Target Regularization

Considering that EGSGP constructs an initial population for GSGP using pre-trained BL models, a potential tendency for overfitting, due to the lack of initial diversity, could imaginably become a problem. For this reason, regularization may be an essential step for EGSGP. The choice of implementing Soft Target (ST) (first proposed by Aghajanyan in [14]) and not another regularization method was mainly anchored in the fact that ST was already successfully employed with GP [15]. ST works by maintaining information from the models' outputs in the early phases of the training process, as their early mistakes can be useful and provide information regarding what Aghajanyan called the "co-label effects" [15]. More specifically, ST redefines the traditional loss function as a quantification of the distance between the calculated outputs and the soft target. The soft target is a weighted average between the expected outputs and some information concerning the past models' outputs. At each generation t, this information consists in a moving average of the past models' outputs $\hat{y}^t$, defined as:

$$\begin{cases} \hat{y}^0 = f^0(X) \\ \hat{y}^t = \beta \cdot \hat{y}^{t-1} + (1-\beta) \cdot f^t(X), \quad \text{if } t > 0 \end{cases} \tag{1}$$

where X is the set of input vectors, $X = \{\overrightarrow{x_1}, \overrightarrow{x_2}, ..., \overrightarrow{x_n}\}$. Consequently, we update the value of the soft target $T^t = \{t_1^t, t_2^t, ..., t_n^t\}$ through a weighted average of $\hat{y}^t$ and the true expected output $y = \{y_1, y_2, ..., y_n\}$, as follows:

$$\begin{cases} t^0 = y \\ t^t = \gamma \cdot \hat{y}^t + (1 - \gamma) \cdot y, \quad \text{if } t > 0 \end{cases} \tag{2}$$

In the previous two equations, β and γ are parameters of the algorithm. The former (β) allows us to tune the importance of the past models' outputs compared to the current one, while the latter (γ) allows us to tune the importance of the moving average of the past models, compared to the true expected output. Finally, for each iteration t, the loss function $\mathcal{L}$ is defined in the following way:

$$\mathcal{L}(f^t) = \left(\frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_i(f^t(x_i), t_i^t) \right)^m \tag{3}$$

where, in the case of our work, we used $\mathcal{L}_i(a, b) = (a - b)^2$ and $m = 1/2$, thus turning $\mathcal{L}$ into a Root Mean Square Error (RMSE), as it is customary for GP [15]. ST regularization requires two parameters in addition to β and γ: the burning period (bp) and the frequency of update (fu), corresponding to the number of initial generations in which the algorithm is executed without any ST regularization and the frequency of update of the soft target, respectively. Thus, setting the ST parameters allows us to tune how much importance is given to the early stages of the learning. The next section provides further details regarding how ST regularization was used by the EGSGP algorithm.

4 Ensembled Geometric Semantic Genetic Programming

Simply put, EGSGP consists in a novel initialization for GSGP. In EGSGP, the population is no longer seeded with random individuals. Instead, the initial individuals are generated using the semantics of the BL models, that had been previously trained. Six different methods were constructed in order to assess the performance of our proposed technique. These methods are characterized by the possible combinations of elitism and ST regularization, that we identify as two important elements to counteract overfitting. As previously discussed, the ST regularization depends on different parameters, that allow us to tune the importance given to the previous models during the learning. Thus, when ST regularization is used, two different sets of ST parameter combinations are employed. For simplicity, we call the two variants ST_{pa} and ST_{pb}.

ST_{pa} consists of ST regularization with a parameter set that gives high importance to past models' outputs (and therefore to the original BLs), starting the regularization at an early stage of the learning process. The parameters that characterize ST_{pa} version are: $\beta = 0.95, \gamma = 0.95, bp = 0, fu = 2$.

ST_{pb}, instead, applies ST regularization with a parameter set that is more balanced, giving importance both to the original BLs and to the changes that

happen to them throughout the learning process. The parameters that characterize ST_{pb} are: $\beta = 0.5, \gamma = 0.5, bp = 5, fu = 5$.

Furthermore, EGSGP with no regularization was also studied. In this fashion, the combination of the whether Elitism or ST regularization (and which ST regularization parameters) were or not employed results in six EGSGP models: EL_NO_ST, that uses elitism and no ST; NO_EL_NO_ST, that uses no elitism and no ST; EL_STA, that uses elitism and ST_{pa}; NO_EL_STA, that uses no elitism and ST_{pa}; EL_STB, that uses elitism and ST_{pb}; and NO_EL_STB, that uses no elitism and no ST.

5 Experimental Study

Base Learners. In this work, EGSGP seeds the initial population of GSGP with the semantics of 15 different BL pre-trained models, that underwent no parameter optimization, since their outputs will later be further trained by GSGP. Hence, the default setting of the Python libraries' functions were kept. The used BLs were: Decision Trees, Random Forests, Extra Tree Regressor, AdaBoost, Gradient Boosting, Lasso Regression, Ridge Regression, Support Vector Machines, K-Nearest Neighbours, Elastic Net, Multilayer Perceptron, ADR Regressor, Kernel Ridge and two variants of XgBoost.

Test Cases. Three real-life regression problems were used to assess the performance of EGSGP. The first one, the Concrete dataset, has the objective to predict the strength of high perfomance concrete given its components. The remaining two datasets are problems from the biomedical area; the Plasma Protein Binding (PPB) levels dataset aims at predicting the value of a pharmacokinetic parameter (the PPB), based on a set of molecular descriptors of potential new drugs, and the Unified Parkinson's Disease Rating Scale (UPDRS) dataset seeks to predict the UPDRS assessment, in order to aid the automatization of the tracking of the development of Parkinson's disease symptoms. These three datasets have often been used as benchmarks for GP. The Concrete dataset is discussed in [16], and it is characterized by 8 features and 1030 observations. The PPB dataset is described in [17], and it is characterized by 626 features and 131 observations. The Parkinson's dataset was introduced in [18], and it is characterized by 18 features and 5875 observations.

Experimental Settings. We performed the Monte-Carlo cross-validation with 30 independent runs, using a different random seed for each one of them. At each run, 30% of the data was used for testing. The remaining portion of the data (70%) was split in two parts: training and validation, with 70% and 30% of that remaining portion respectively. For each dataset, the six different EGSGP models were tested. The GSGP parameters, like the probability of mutation (PMut), the probability of crossover (PXover), the mutation step (ms), and others, were determined via a grid-search, where the parameter combination with the best median validation error was chosen. The parameter values

tested in the grid search were: crossover and mutation rates equal to 0.25, 0.5, 0.75 and 1; mutation step equal to 0.001, 0.01, 0.1 and 1; tournament size equal to 2 and 10; population size equal to 25, 50, 100, 200, 250; maximum number of generations equal to 25, 50, 100, 200, 250 (plus the additional values of 650 and 1250, that were tested only for GSGP). We gave GSGP the possibility of evolving for a larger number of generations than EGSGP (in fact, for GSGP also the values 650 and 1250 were tested), because the initial population of EGSGP is composed of pre-trained models. Hence, we considered that allowing GSGP to train for more generations is important for having a fair comparison. More specifically, the value 1250 was chosen in view of the fact that the highest number of maximum iterations for the training stage of the BLs was 1000. Therefore, EGSGP can be seen as having an additional 1000 iterations when compared to GSGP, unless we allow GSGP to run for 1000 more generations.

Finally, an assessment of the models was carried out by executing a comparison between EGSGP, the best BL for each test case, and traditional GSGP. The validation dataset was not solely used for parameter optimization. Additionaly, BLs that were deemed as "overfitters" based the on validation dataset were not included in the initial population. For the Concrete and Parkinson's case studies, we considered that a BL overfits if its median validation error was more than four times its median training error (given that that amount was far from the "normally expected" seen to unseen error worsening in our particular case studies). For the PPB dataset, given its noticeable tendency to overfit, BLs with a median training error below 0.15 and with a median validation error above two times the median validation error of all the BLs, were excluded. Finally, in the Concrete test case, no BLs were excluded from the initial EGSGP population. For the PPB dataset, the BL models that were excluded were the ones generated by Extra Tree Regressor, XgBoost, Gradient Boosting, Lasso, Ridge, K-Nearest Neighbours, Elastic Net and Multilayer Perceptron. For the Parkinson's test case, only one model was excluded: the one generated by AdaBoost.

Experimental Results. The obtained experimental results are reported in Tables 1, 2 and 3 (for the Concrete, PPB and Parkinson's case studies, respectively). The first column of these tables reports the studied methods (either one of the EGSGP variants, or traditional GSGP, or the best BL). The remaining columns present the median RMSE of the models for both training and unseen (i.e., testing) data. Furthermore, to assess the statistical significance of the results, a set of tests were performed. Their outcomes are presented in Tables 4, 5 and 6 (for the Concrete, PPB and Parkinson, respectively). Since the results of the Kolmogorov-Smirnov tests revealed that the data are not normally distributed, the non-parametric Mann-Whitney U-statistic was chosen for our analyses. As we can see from the tables, EGSGP is able to outperform traditional GSGP in all the studied test problems, both on training and unseen (i.e., testing) data. The difference between the two methods is statistically significant in all case studies, with the exception of PPB. However, when we observe the same tables and compare EGSGP with the best BL, we conclude that the fre-

Table 1. Results of the median training and unseen error obtained on the Concrete case study over 30 independent runs for the EGSGP models, GSGP and the best BL. The best training and unseen error are highlighted in bold.

Method	Median Training Error	Median Unseen Error
EL_NO_ST	3.290	5.462
NO_EL NO_ST	**3.317**	5.495
EL_STA	3.434	5.468
NO_EL_STA	3.421	5.461
EL_STB	3.332	**5.442**
NO_EL_STB	3.339	5.491
Best BL	3.455	5.477
GSGP	13.325	13.627

Table 2. Results of the median training and unseen error obtained on the PPB case study over 30 independent runs for the EGSGP models, GSGP and the best BL. The best training and unseen error are highlighted in bold.

Method	Median Training Error	Median Unseen Error
EL_NO_ST	0.188	33.378
NO_EL NO_ST	1.535	33.280
EL_STA	**0.186**	33.366
NO_EL_STA	0.650	**32.915**
EL_STB	0.260	33.042
NO_EL_STB	1.536	33.282
Best BL	0.189	33.378
GSGP	36.890	39.749

Table 3. Results of the median training and unseen error obtained on the Parkinson's case study over 30 independent runs for the EGSGP models, GSGP and the best BL. The best training and unseen error are highlighted in bold.

Method	Median Training Error	Median Unseen Error
EL_NO_ST	2.607	3.221
NO_EL NO_ST	**2.587**	**3.218**
EL_STA	2.594	3.227
NO_EL_STA	2.623	3.284
EL_STB	2.607	3.223
NO_EL_STB	2.588	3.220
Best BL	2.641	3.309
GSGP	7.699	7.721

Table 4. p-values of the Mann-Whitney test for the Concrete dataset.

Method	Method vs GSCP	Method vs BL
EL_NO_ST	$< 10^{-4}$	0.17774
NO_EL NO_ST	$< 10^{-4}$	0.33137
EL_STA	$< 10^{-4}$	0.35859
NO_EL_STA	$< 10^{-4}$	0.32071
EL_STB	$< 10^{-4}$	0.18554
NO_EL_STB	$< 10^{-4}$	0.34216

quently observed improvement on both training and unseen error is only enough to yield statically significant results on the Parkinson's case study.

For each one of the studied test problems, Figure 1 reports the error evolution on the test set, against generations, for the best EGSGP model, the best BL (represented as an horizontal line in the plots) and GSGP, while Figure 2 reports an overall comparative performance assessment of the different EGSGP models.

Table 5. p-values of the Mann-Whitney test for the PPB dataset.

Method	Method vs GSCP	Method vs BL
EL_NO_ST	0.08343	0.47642
NO_EL NO_ST	0.08573	0.47937
EL_STA	0.08806	0.48526
NO_EL_STA	0.05599	0.38656
EL_STB	0.07472	0.4676
NO_EL_STB	0.08573	0.47937

Table 6. p-values of the Mann-Whitney test for the Parkinson's dataset.

Method	Method vs GSCP	Method vs BL
EL_NO_ST	$< 10^{-4}$	0.00031
NO_EL NO_ST	$< 10^{-4}$	0.0002
EL_STA	$< 10^{-4}$	0.00026
NO_EL_STA	$< 10^{-4}$	0.10310
EL_STB	$< 10^{-4}$	0.00036
NO_EL_STB	$< 10^{-4}$	0.00019

For the Concrete dataset, we can conclude that the EGSGP models are able to improve the fitness of the best BL without, however, yielding significant statistical results. This is true with the exception of methods NO_EL_NO_ST and NO_EL_STB. Note that both of these methods do not use elitism. Interestingly, adding a ST regularization that gives great importance to previous models' outputs (method NO_EL_STA) clearly reduces overfitting. Also, EGSGP was able to improve the fitness of the best BL, regardless of the hyper-parameter model combination, in both the PPB and Parkinson's case studies. Finally, we point out that in the Parkinson's case study, EGSGP significantly outperformed the best BL (with the exception of method NO_EL_STA).

6 Conclusions and Future Work

This work exploited the idea of seeding a Genetic Programming (GP) population with models generated by other Machine Learning (ML) methods, instead of using randomly generated individuals. While tackling this problem with standard GP would have been extremely challenging, because each ML method uses its own representation, typically very different from the others, the task becomes straightforward if we use Geometric Semantic GP (GSGP). In fact, GSGP allows us to abstract from the representation of the evolving individuals, focusing purely on their semantics (i.e., the vector of their output values on the training observations). Following this idea, in this paper we have presented EGSGP, a novel method, that can be seen either as a new intialization technique for GSGP or as an ensemble method, that uses GSGP to combine different Base Learners (BLs). Given that the initial population of EGSGP is composed by pre-trained models, it is easy to expect that a further optimization of those models, given by the GSGP evolution, would lead the system to overfit training data. For this reason, we identified elitism and Soft Target (ST) regularization as elements that, if appropriately tuned, could help us limit overfitting and so several variants of EGSGP were studied. In particular, variants that use or do not use elitism, and variants that use (with different parameters) or do not use ST were studied. Those variants were compared with the best BL and with GSGP on three real-life regression problems (Concrete, PPB and Parkinson's datasets). An analysis

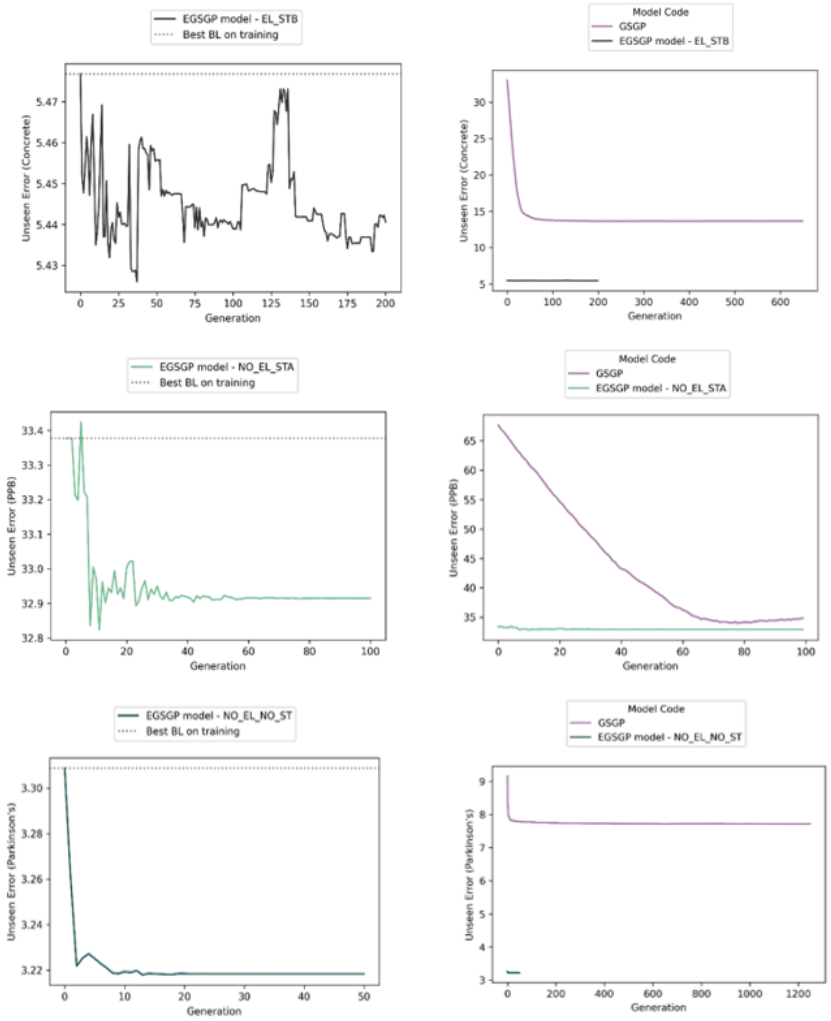

Fig. 1. Unseen error comparison between the best EGSGP method and the best BL (on the left) and the best ESGP method and GSGP (on the right) in the Concrete case study (first row), the PPB case study (second row) and the Parkinson's case study (third row).

of the presented results indicates that there is no EGSGP hyper-parameter combination that outperforms the others in all case studies. For the Concrete and PPB case studies, ST yielded the best results (with Elitism and without elitism, respectively) while the best model for the Parkinson's case study was obtained without ST and without elitism. Furthermore, the best models were, in two of the three case studies, the ones with no elitism. In general, having elitism or not affects the models' results more when the ST regularization process is started

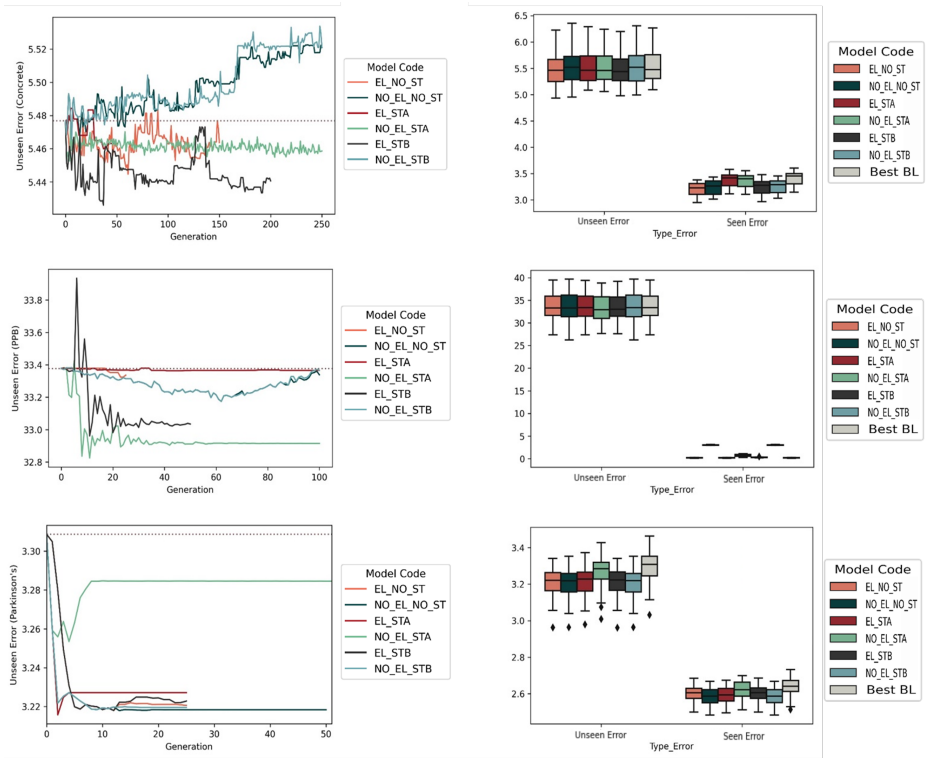

Fig. 2. Error comparison between the different EGSGP models and the best BL for the Concrete case study (first row), the PPB case study (middle row) and the Parkinson's case study (last row). The leftmost column presents the unseen error comparison, while the rightmost column presents the distribution of the median training and unseen error for each case study over the 30 seeds.

at an earlier stage of the evolution, giving more importance to the original BLs. With respect to its competitors, EGSGP was able to not only outperform GSGP, but also the best BL. This improvement yielded statistically significant results in one of the three case studies. Nonetheless, we consider the performance of EGSGP to be overall positive, given the fact that it was able to improve the best BL in all case studies, even when not enough to be deemed statistically significant. These results reflect GSGP's potential to become a powerful meta-learning combiner that is able to integrate the knowledge obtained from a set of different ML models, in a way that outputs results that are better than the ones individually obtained. Future work should focus on the reconstruction of the genotype of the final model returned by EGSGP. The task is ambitious, because it implies the definition of a middle layer of representation, common to all the ML methods used as BLs.

References

1. Zhi-Hua Zhou. *Ensemble methods: foundations and algorithms*. Chapman and Hall/CRC, 2019.
2. Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.
3. Sebastian Thrun and Lorien Pratt. Learning to learn: Introduction and overview. In *Learning to learn*, pages 3–17. Springer, 1998.
4. Joaquin Vanschoren. Meta-learning. In *Automated Machine Learning*, pages 35–61. Springer, Cham, 2019.
5. Lior Rokach. *Pattern classification using ensemble methods*, volume 75. World Scientific, 2010.
6. Alberto Moraglio, Krzysztof Krawiec, and Colin G Johnson. Geometric semantic genetic programming. In *International Conference on Parallel Problem Solving from Nature*, pages 21–31. Springer, 2012.
7. John R Koza. *Genetic programming: on the programming of computers by means of natural selection*, volume 1. MIT press, 1992.
8. Illya Bakurov, Leonardo Vanneschi, Mauro Castelli, and Francesco Fontanella. Edda-v2–an improvement of the evolutionary demes despeciation algorithm. In *International Conference on Parallel Problem Solving from Nature*, pages 185–196. Springer, 2018.
9. Leonardo Vanneschi, Mauro Castelli, and Sara Silva. A survey of semantic methods in genetic programming. *Genetic Programming and Evolvable Machines*, 15(2):195–214, 2014.
10. Leonardo Vanneschi. An introduction to geometric semantic genetic programming. In *NEO 2015*, pages 3–42. Springer, 2017.
11. Tomasz P Pawlak and Krzysztof Krawiec. Semantic geometric initialization. In *European Conference on Genetic Programming*, pages 261–277. Springer, 2016.
12. Illya Bakurov. *An initialization technique for geometric semantic geneticprogramming based on demes evolution and despeciation: machine learning for rare diseases: a case study*. PhD thesis, NOVA IMS, 2018.
13. David W. Fan, Philip K. Chan, and Salvatore J. Stolfo. A comparative evaluation of combiner and stacked generalization. In *Proceedings of AAAI-96 workshop on Integrating Multiple Learned Models*, pages 40–46, 1996.
14. Armen Aghajanyan. Soft target regularization: An effective technique to reduce over-fitting in neural networks. In *2017 3rd IEEE International Conference on Cybernetics (CYBCONF)*, pages 1–5. IEEE, 2017.
15. Leonardo Vanneschi and Mauro Castelli. Soft target and functional complexity reduction: A hybrid regularization method for genetic programming. *Expert Systems with Applications*, 177:114929, 2021.
16. Mauro Castelli, Leonardo Vanneschi, and Sara Silva. Prediction of high performance concrete strength using genetic programming with geometric semantic genetic operators. *Expert Systems with Applications*, 40(17):6856–6862, 2013.
17. Leonardo Vanneschi. Improving genetic programming for the prediction of pharmacokinetic parameters. *Memetic Computing*, 6(4):255–262, 2014.
18. Mauro Castelli, Leonardo Vanneschi, and Sara Silva. Prediction of the unified parkinson's disease rating scale assessment using a genetic programming system with geometric semantic genetic operators. *Expert Systems with Applications*, 41(10):4608–4616, 2014.