Semantic Learning Machine: A Feedforward Neural Network Construction Algorithm Inspired by Geometric Semantic Genetic Programming

Ivo Gonçalves[1,2(✉)], Sara Silva[1,2,3], and Carlos M. Fonseca[1]

[1] CISUC, Department of Informatics Engineering,
University of Coimbra, 3030-290 Coimbra, Portugal
icpg@dei.uc.pt
[2] BioISI - Biosystems & Integrative Sciences Institute, Faculty of Sciences,
University of Lisbon, 1749-016 Campo Grande, Lisbon, Portugal
sara@fc.ul.pt
[3] NOVA IMS, Universidade Nova de Lisboa, 1070-312 Lisbon, Portugal
cmfonsec@dei.uc.pt

Abstract. Geometric Semantic Genetic Programming (GSGP) is a recently proposed form of Genetic Programming in which the fitness landscape seen by its variation operators is unimodal with a linear slope by construction and, consequently, easy to search. This is valid across all supervised learning problems. In this paper we propose a feedforward Neural Network construction algorithm derived from GSGP. This algorithm shares the same fitness landscape as GSGP, which allows an efficient search to be performed on the space of feedforward Neural Networks, without the need to use backpropagation. Experiments are conducted on real-life multidimensional symbolic regression datasets and results show that the proposed algorithm is able to surpass GSGP, with statistical significance, in terms of learning the training data. In terms of generalization, results are similar to GSGP.

1 Introduction

Moraglio et al. [6] recently proposed a new Genetic Programming formulation called Geometric Semantic Genetic Programming (GSGP). GSGP derives its name from the fact that it is formulated under a geometric framework [5] and from the fact that it operates directly in the space of the underlying semantics of the individuals. In this context, semantics is defined as the outputs of an individual over a set of data instances. The most interesting property of GSGP is that the fitness landscape seen by its variation operators is always unimodal with a linear slope (cone landscape) by construction. This implies that there are no local optima, and consequently, that this type of landscape is easy to search. When applied to multidimensional real-life datasets, GSGP has shown competitive results in learning and generalization [3,7]. In this paper, we adapt

F. Pereira et al. (Eds.) EPIA 2015, LNAI 9273, pp. 280–285, 2015.
DOI: 10.1007/978-3-319-23485-4_28

the geometric semantic mutation to the realm of feedforward Neural Networks by proposing the Semantic Learning Machine (SLM). Section 2 defines the SLM. Section 3 describes the experimental setup. Section 4 presents and discusses the results of the SLM and GSGP, and Section 5 concludes.

2 Semantic Learning Machine

Given that the geometric semantic operators are defined over the semantic space (outputs), they can be extended for different representations. The Semantic Learning Machine (SLM) proposed in this section is based on a derivation of the GSGP mutation operator for real-value semantics. This implies that the SLM shares the same semantic landscape proprieties as GSGP. Particularly, the fitness landscape induced by its operator is always unimodal with a linear slope (cone landscape) by construction, and consequently easy to search. This is valid across all supervised learning problems.

2.1 A Geometric Semantic Mutation Operator for Feedforward Neural Networks

The GSGP mutation for real-value semantics [6] is defined as follows:

Definition 1. (GSGP Mutation). *Given a parent function* $T : \mathbb{R}^n \rightarrow \mathbb{R}$*, 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.*

This mutation essentially performs a linear combination of two individuals: the parent and a randomly generated tree (which results from subtracting the two subtrees T_{R1} and T_{R2}). The degree of semantic change is controlled by the mutation step.

An equivalent geometric semantic mutation operator can be derived for feedforward Neural Networks (NN). The only three small restrictions for this NN mutation operator are: the NN must have at least one hidden layer; the output layer must have only one neuron; and the output neuron must have a linear activation function. Each application of the operator adds a new neuron to the last hidden layer. The weight from the new neuron to the output neuron is defined by the learning step (SLM parameter). This learning step is the equivalent of the mutation step in the GSGP mutation. It defines the amount of semantic change for each application of the operator. The weights from the last hidden layer to the previous layer are randomly generated. This is the equivalent of generating the two random subtrees in the GSGP mutation. In this work these weights are generated with uniform probability between -1.0 and 1.0. If more than one hidden layer is used, all other weights remain constant once initialized. In this work all experiments are conducted with a single hidden layer. The activation function for the neurons in the last hidden layer can be freely chosen. However, it has been recently shown, in the context of GSGP, that applying a structural bound to the randomly generated tree (which results from subtracting

the two subtrees T_{R1} and T_{R2}) results in significant improvements in terms of generalization ability [3]. In fact, if a unbounded mutation (equivalent to using a linear activation function) is used, there is a tendency for GSGP to greatly overfit the training data [3]. For this reason, it is recommended that the activation function for the neurons in the last hidden layer to be a function with a relatively small codomain. In this work a modified logistic function (transforming the logistic function output to range in the interval $[-1, 1]$) is used for this purpose. In terms of generalization ability, it is also essential to use a small learning/mutation step [3]. If more than one hidden layer is used, the activation functions for the remaining neurons may be freely chosen.

2.2 Algorithm

The SLM algorithm is essentially a geometric semantic hill climber for feedforward neural networks. The idea is to perform a semantic sampling with a given size (SLM parameter) by applying the mutation operator defined in the previous subsection. As is common in hill climbers, only one solution (in this case a neural network) is kept along the run. At each iteration, the mentioned semantic sampling is performed to produce N neighbors. At the end of the iteration, the best individual from the previous best and the newly generated individuals is kept. The process is repeated until a given number of iterations (SLM parameter) has been reached. As mentioned in the previous subsection, the mutation operator always adds a new neuron to the last hidden layer, so the number of neurons in the last hidden layer is at most the same as the number of iterations. This number of neurons can be smaller than the number of iterations if in some iterations it was not possible to generate an individual superior to the current best.

3 Experimental Setup

The experimental setup is based on the setup of Vanneschi et al. [7] and Gonçalves et al. [3], since these works recently provided results for GSGP. Experiments are run for 500 iterations/generations because that is where the statistical comparisons were made in the mentioned works. 30 runs are conducted. Population/sample size is 100. Training and testing set division is 70% - 30%. Fitness is computed as the root mean squared error. The initial tree initialization is performed with the ramped half-and-half method, with a maximum depth of 6. Besides GSGP, the Semantic Stochastic Hill Climber (SSHC) [6] is also used as baseline for comparison. The variation operators used are the variants defined for real-value semantics [6]: SGXM crossover for GSGP, and SGMR mutation for GSGP and SSHC. For GSGP a probability of 0.5 is used for both operators. The function set contains the four binary arithmetic operators: +, -, *, and / (protected). No constants are used in the terminal set. Parent selection in GSGP is based on tournaments of size 4. Also for GSGP, survivor selection is elitist as the best individual always survives to the next generation. All claims

of statistical significance are based on Mann-Whitney U tests, with Bonferroni correction, and considering a significance level of $\alpha = 0.05$. For each dataset 30 different random partitions are used. Each method uses the same 30 partitions. Experiments are conducted on three multidimensional symbolic regression real-life datasets. These datasets are the Bioavailability (hereafter Bio), the Plasma Protein Binding (hereafter PPB), and the Toxicity (hereafter LD50). The first two were also used by Vanneschi et al. [7] and Gonçalves et al. [3]. These datasets have, respectively: 359 instances and 241 features; 131 instances and 626 features; and 234 instances and 626 features. For a detailed description of these datasets the reader is referred to Archetti et al. [1]. These datasets have also been used in other Genetic Programming studies, e.g., [2,4].

4 Experimental Study

Figure 1 presents the training and testing error evolution plots for SLM, GSGP and SSHC. These evolution plots are constructed by taking the median over 30 runs of the training and testing error of the best individuals in the training data. The mutation/learning step used was 1 for the the Bio and PPB datasets (as in Vanneschi et al. [7] and Gonçalves et al. [3]), and 10 for the LD50 as it was found, in preliminary testing, to be a suitable value (other values tested were: 0.1, 1, and 100). A consideration for the different initial values (at iteration/generation 0) is in order. The SLM presents much higher errors than GSGP/SSHC after the random initialization. This is explained by the fact that the weights for the SLM are generated with uniform probability between -1.0 and 1.0, and consequently, the amount of data fitting is clearly bounded. On the other hand, GSGP and SSHC have no explicit bound on the random trees and therefore can provide a superior initial explanation of the data. It is interesting to note that, despite this initial disadvantage, the SLM compensates with a much higher learning rate. This higher learning efficiency is confirmed by the statistically significant superiority found in terms of training error across all datasets, against GSGP (p-values: Bio 2.872×10^{-11}, PPB 2.872×10^{-11} and LD50 7.733×10^{-10}), and against SSHC (p-values: Bio 2.872×10^{-11}, PPB 2.872×10^{-11} and LD50 3.261×10^{-5}).

This learning superiority is particularly interesting when considering that the SLM and the SSHC use the exact same geometric semantic mutation operator. This raise the question: how can two methods with the same variation operator, the same induced semantic landscape, and the same parametrizations achieve such different outcomes? The answer lies in the different semantic distributions that result from the random initializations. Different representations have different natural ways of being randomly initialized. This translates into different semantic distributions and, consequently, to different offspring distributions. From the results it is clear that the distribution induced by the random initialization of a list of weights (used in SLM), is more well-behaved than the initialization of a random tree (used in SSHC). In the original GSGP proposal, Moraglio et al. [6] provided a discussion on whether syntax (representation) matters in terms of search. They argued that, in abstract, the offspring distributions may be affected by the different syntax initializations. In our work, we

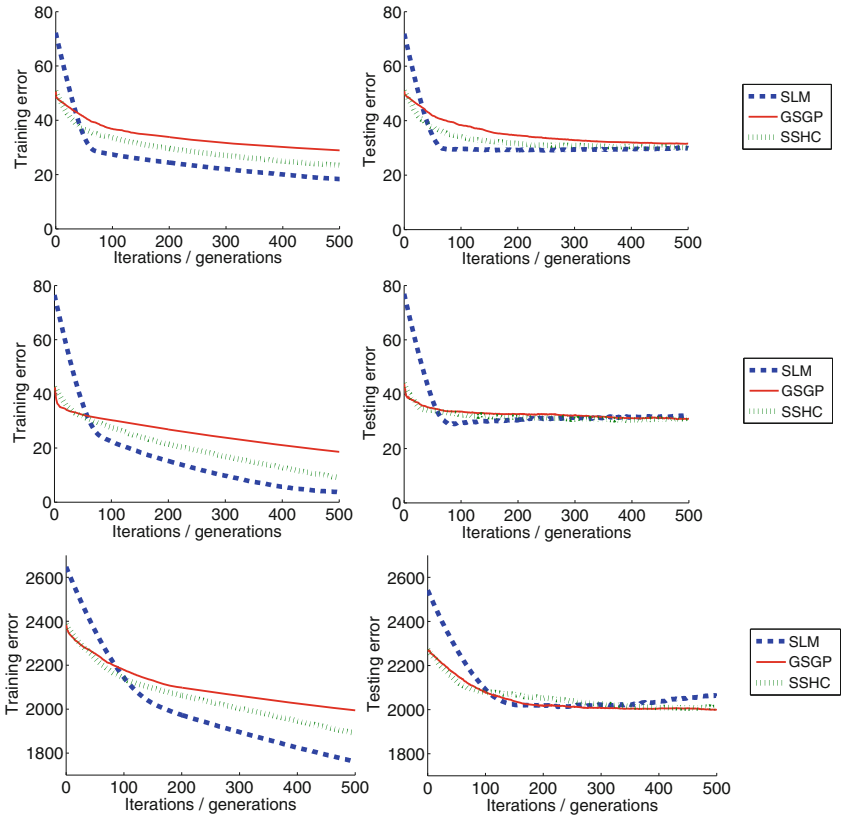

Fig. 1. Bio (top), PPB (center) and LD50 (bottom) training and testing error evolution plots

can empirically see how different representations induce different offspring distributions and consequently reach considerably different outcomes. A possible research venue lies in analyzing the semantic distributions induced by different tree initialization methods, and to possibly propose new tree initializations that are more well-behaved.

In terms of generalization, results show that all methods achieve similar results. The only statistically significant difference shows that the SLM is superior to GSGP in the Bio dataset (p-value: 1.948×10^{-4}). However, it seems that in this case, GSGP is still evolving and that in a few more generations may reach a generalization similar to the SLM. On a final note, the evolution plots also show that SSHC consistently learns the training data faster and better than GSGP. This should be expected as the semantic space has no local optima and consequently the search can be focused around the best individual in the population. These differences are confirmed as statistically significant (p-values: Bio 2.872×10^{-11}, PPB 2.872×10^{-11} and LD50 1.732×10^{-4}). There are no statistically significant differences in terms of generalization.

5 Conclusions

This work presented a novel feedforward Neural Network (NN) construction algorithm, derived from Geometric Semantic Genetic Programming (GSGP). The proposed algorithm shares the same fitness landscape as GSGP, which enables an efficient search for any supervised learning problem. Results in regression datasets show that the proposed NN construction algorithm is able to surpass GSGP, with statistical significance, in terms of learning the training data. Generalization results are similar to those of GSGP. Future work involves extending the experimental analysis to other regression datasets and to provide results for classification tasks. Comparisons with other NN algorithms and other commonly used supervised learning algorithms (e.g. Support Vector Machines) are also in order.

Acknowledgments. This work was partially supported by national funds through FCT under contract UID/Multi/04046/2013 and projects PTDC/EEI-CTP/2975/2012 (MaSSGP), PTDC/DTP-FTO/1747/2012 (InteleGen) and EXPL/EMS-SIS/1954/2013 (CancerSys). The first author work is supported by FCT, Portugal, under the grant SFRH/BD/79964/2011.

References

1. Archetti, F., Lanzeni, S., Messina, E., Vanneschi, L.: Genetic programming for computational pharmacokinetics in drug discovery and development. Genetic Programming and Evolvable Machines **8**(4), 413–432 (2007)
2. Gonçalves, I., Silva, S.: Balancing learning and overfitting in genetic programming with interleaved sampling of training data. In: Krawiec, K., Moraglio, A., Hu, T., Etaner-Uyar, A.Ş., Hu, B. (eds.) EuroGP 2013. LNCS, vol. 7831, pp. 73–84. Springer, Heidelberg (2013)
3. Gonçalves, I., Silva, S., Fonseca, C.M.: On the generalization ability of geometric semantic genetic programming. In: Machado, P., Heywood, M.I., McDermott, J., Castelli, M., García-Sánchez, P., Burelli, P., Risi, S., Sim, K. (eds.) Genetic Programming. LNCS, vol. 9025, pp. 41–52. Springer, Heidelberg (2015)
4. Gonçalves, I., Silva, S., Melo, J.B., Carreiras, J.M.B.: Random sampling technique for overfitting control in genetic programming. In: Moraglio, A., Silva, S., Krawiec, K., Machado, P., Cotta, C. (eds.) EuroGP 2012. LNCS, vol. 7244, pp. 218–229. Springer, Heidelberg (2012)
5. Moraglio, A.: Towards a Geometric Unification of Evolutionary Algorithms. Ph.D. thesis, Department of Computer Science, University of Essex, UK, November 2007
6. Moraglio, A., Krawiec, K., Johnson, C.G.: Geometric semantic genetic programming. In: Coello, C.A.C., Cutello, V., Deb, K., Forrest, S., Nicosia, G., Pavone, M. (eds.) PPSN 2012, Part I. LNCS, vol. 7491, pp. 21–31. Springer, Heidelberg (2012)
7. Vanneschi, L., Castelli, M., Manzoni, L., Silva, S.: A new implementation of geometric semantic GP and its application to problems in pharmacokinetics. In: Krawiec, K., Moraglio, A., Hu, T., Etaner-Uyar, A.Ş., Hu, B. (eds.) EuroGP 2013. LNCS, vol. 7831, pp. 205–216. Springer, Heidelberg (2013)