Energy and Buildings 102 (2015) 67–74

Contents lists available at ScienceDirect

Energy and Buildings

journal homepage: www.elsevier.com/locate/enbuild

Prediction of energy performance of residential buildings: A genetic programming approach

Mauro Castelli[a,*], Leonardo Trujillo[b], Leonardo Vanneschi[a], Aleš Popovič[c,a]

[a] NOVA IMS, Universidade Nova de Lisboa, 1070-312 Lisboa, Portugal
[b] Instituto Tecnológico de Tijuana, Tijuana 22500, Mexico
[c] University of Ljubljana, Faculty of Economics, Kardeljeva ploščad 17, 1000 Ljubljana, Slovenia

ARTICLE INFO

Article history:
Received 16 December 2014
Received in revised form 7 April 2015
Accepted 1 May 2015
Available online 19 May 2015

Keywords:
Energy consumption
Heating load
Cooling load
Genetic programming
Machine learning

ABSTRACT

Energy consumption has long been emphasized as an important policy issue in today's economies. In particular, the energy efficiency of residential buildings is considered a top priority of a country's energy policy. The paper proposes a genetic programming-based framework for estimating the energy performance of residential buildings. The objective is to build a model able to predict the heating load and the cooling load of residential buildings. An accurate prediction of these parameters facilitates a better control of energy consumption and, moreover, it helps choosing the energy supplier that better fits the energy needs, which is considered an important issue in the deregulated energy market. The proposed framework blends a recently developed version of genetic programming with a local search method and linear scaling. The resulting system enables us to build a model that produces an accurate estimation of both considered parameters. Extensive simulations on 768 diverse residential buildings confirm the suitability of the proposed method in predicting heating load and cooling load. In particular, the proposed method is more accurate than the existing state-of-the-art techniques.

1. Introduction

Energy consumption of buildings has received increasing interest in today's economies. As buildings represent substantial consumers of energy worldwide, with this trend increasing over the past few decades due to rising living standards, this issue has drawn considerable attention from various stakeholders (e.g. inhabitants, policy makers, industry). As reported in [1], from 1994 to 2004, building energy consumption in Europe and North America has increased at a rate of 1.5% and 1.9% per year, respectively. An even bigger increase has characterized the energy market in China, where building energy consumption has increased at more than 10% per year for the past 20 years [2]. The high level of building energy consumption and the steady increase in building energy demand require the design of energy efficient buildings and an improvement of their energy performance. Another important aspect is the effect of this continuous increase of energy consumption on the environment. According to a report of the United Nations Environment Program (UNEP) [3] buildings use about 40% of global energy, 25% of global water and 40% of global resources. In many countries, there is a real danger that, as a consequence of global warming and climate change (including growing reliance on air-conditioning) energy demand and CO_2 emissions will increase even further [4]. In an effort to reduce the impact of building energy consumption on the environment, the European Union has recently adopted a directive (European Directive 2002/91/EC [5]) requiring European countries to conform to appropriate minimum requirements regarding energy efficiency.

With regard to residential buildings, the largest part of the energy consumption is due to the use of so-called *heating, ventilation and air-conditioning* (HVAC) systems. As reported in [6], one way to alleviate the ever increasing demand for additional energy supply is to have more energy-efficient building designs with improved energy conservation properties. Ensuring that the right HVAC system is installed in a building is critical not only for providing consistent indoor comfort for families, but also for saving energy, which can be wasted by either a too-large or too-small system. Consumers and designers should have an idea of the approximate area a given piece of HVAC equipment might be expected to heat or cool under ideal conditions. For this aim, a correct estimation of the *heating load* (HL) and the *cooling load* (CL) is extremely important to achieve an efficient HVAC design for

* Corresponding author. Tel.: +351 213 828 628; fax: +351 213 872 140.
E-mail addresses: mcastelli@novaims.unl.pt (M. Castelli), leonardo.trujillo@tectijuana.edu.mx (L. Trujillo), lvanneschi@novaims.unl.pt (L. Vanneschi), apopovica@novaims.unl.pt (A. Popovič).

http://dx.doi.org/10.1016/j.enbuild.2015.05.013
0378-7788/

buildings. The HL is the amount of heat energy that would need to be added to a space to maintain the temperature in an acceptable range and the CL is the amount of heat energy that would need to be removed from a space (cooling) to maintain the temperature in an acceptable range. The HL and CL, also known as *thermal loads*, take into account the building's construction and insulation (including floors, walls, ceilings and roof) and the building's glazing and skylights (based on size, performance, shading and overshadowing). Reliable estimations of HL and CL are of paramount importance and can have a serious impact on economy, since mistakes in these estimations may imply a waste of energy. Nevertheless, accurately predicting HL and CL is a difficult task. Most HVAC designs are still nowadays based on the personal advice of an HVAC professional, which has a subjective component and thus may be prone to errors. In such a perspective, reliable computational tools for accurately predicting HL and CL are much in demand, and this is the main motivation for the present work.

As discussed in [1], in the process of designing energy efficient buildings, it is important for architects, engineers and designers to identify which parameters will significantly influence future energy demand. After the identification of these parameters, architects and building designers usually need simple and reliable methods for rapidly estimating building energy performance, so that they can optimize their design plans. In recent years, several methods have been proposed for modeling building energy demand. The proposed methods range from traditional regression methods [7,8], to more complex machine learning techniques like artificial neural networks (ANNs) [9–11]. While the proposed techniques have shown good results, they present problems that often prevent designers and architects from using them. For instance, the performance of ANN models is strongly dependent from the design of the network's architecture, which is usually carried out in an ad-hoc and manual way.

In this paper, we propose a machine learning framework to predict the HL and the CL of a large set of residential buildings. The framework uses data related to eight input variables (relative compactness, surface area, wall area, roof area, overall height, orientation, glazing area, glazing area distribution) to construct predictive models. The dataset used to assess the performance of the proposed method is the same as in [6]. While the methods described in [6] represent the state-of-the-art for the prediction of building energy consumption, we show that it is possible to develop a system able to achieve better results. The proposed approach combines a recently defined variant of genetic programming that integrates semantic awareness in the search process, with a local search method and a linear scaling technique.

The paper is organized as follows: Section 2 presents an overview of standard genetic programming and shows how genetic programming can be used to address a symbolic regression problem. Section 3 presents the geometric semantic genetic programming algorithm used in this work. Sections 4 and 5, respectively, describe the local search method and the linear scaling technique. Section 6 describes the dataset that has been used, the experimental settings and provides a detailed discussion of the obtained results. Furthermore, a comparison between the proposed system and other state-of-the-art methods is presented. Finally, Section 7 concludes the paper summarizing the contributions of this work.

2. Genetic programming

Genetic programming (GP) [12] is a computational method that belongs to the computational intelligence research area called evolutionary computation. GP consists in the automated learning of computer programs by means of a process inspired by biological evolution. Generation by generation, GP stochastically transforms populations of programs into new, hopefully improved, populations of programs. The quality of a solution is expressed by using an objective function (also called fitness function). The search process of GP is graphically depicted in Fig. 1.

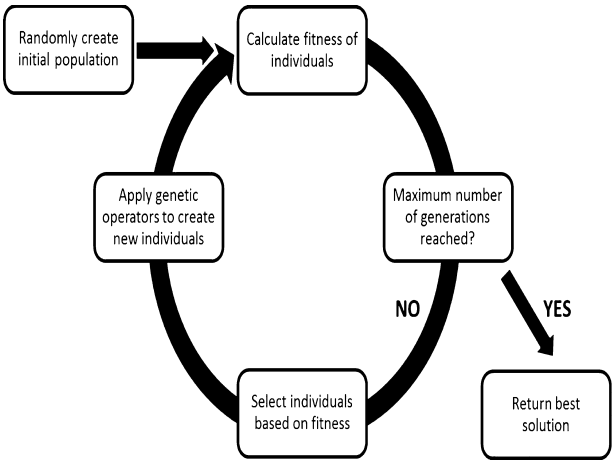

Fig. 1. The GP algorithm.

In order to transform a population into a new population of candidate solutions, GP uses particular search operators called genetic operators. Considering the common tree representation of GP individuals [12], the standard genetic operators (crossover and mutation) act on the structure of the trees that represent the candidate solutions. In other terms, standard genetic operators act on the syntax of the programs. In this paper, we use genetic operators that, differently from the standard ones, are able to act at the semantic level. The definition of semantics used in this work is the one also proposed in [13] and will be discussed in the next section.

However, to understand the differences between the genetic operators used in this work and the ones used in the standard GP algorithm, the latter are briefly described here. The standard crossover operator is traditionally used to combine the genetic material of two parents by swapping a part of one parent with a part of the other. More in detail, after choosing two individuals based on their fitness, the crossover operator selects a random subtree in each parent and swaps the selected subtrees between the two parents. The mutation operator introduces random changes in the structures of the individuals in the population. The traditional and mostly used mutation operator, called sub-tree mutation, works by randomly selecting a point in a tree, removing whatever is currently at the selected point and whatever is below the selected point and inserting a randomly generated tree at that point. This operation is controlled by a parameter that specifies the maximum size (usually measured in terms of tree depth) for the newly created subtree that is to be inserted.

2.1. Symbolic regression with genetic programming

In symbolic regression, the goal is to search for the symbolic expression $T^O : \mathbb{R}^p \rightarrow \mathbb{R}$ that best fits a particular training set $\mathbb{T} = \{(\mathbf{x}_1, t_1), \ldots, (\mathbf{x}_n, t_n)\}$ of n input/output pairs with $\mathbf{x}_i \in \mathbb{R}^p$ and $t_i \in \mathbb{R}$. The general symbolic regression problem can then be defined as

$$(T^O, \boldsymbol{\theta}^{\boldsymbol{O}}) \leftarrow \underset{T \in \mathbb{G}; \boldsymbol{\theta} \in \mathbb{R}^m}{arg\ \min} f(T(\mathbf{x}_i, \boldsymbol{\theta}), t_i) \text{ with } i = 1, \ldots, p \quad (1)$$

where $\mathbb{G}$ is the solution space defined by the primitive set $\mathbb{P}$ (functions and terminals), f is the fitness function based on the distance (or error) between a program's output $T(\mathbf{x}_i, \boldsymbol{\theta})$ and the expected, or target, output t_i, and $\boldsymbol{\theta}$ is a particular parametrization of the symbolic expression T, assuming m real-valued parameters.

In standard GP, parameter optimization is usually not performed explicitly, since GP search operators only focus on syntax. Therefore, the parameters are only implicitly considered. However, recent works have begun to address this issue; for instance in [14] a non-linear numerical optimizer is used to tune the parametrization of the evolved programs, achieving substantial improvements in terms of convergence speed and solution quality. This is the view taken in this work, and while previous works have included parameter optimization for a standard GP search [14], this work applies it to a new GP variant called geometric semantic GP (GSGP), which is described next.

3. Geometric semantic operators

Despite the large number of human-competitive results achieved by GP [15], researchers continue to investigate new methods to improve GP's performance and optimization power. In recent years, one of the emerging ideas is to integrate semantic awareness in the evolutionary process performed by GP. While several studies exist (i.e. [16,17]), the definition of semantics is not unique and this concept can be interpreted in different ways and under different perspectives [18]. In this work, we use the most common and widely accepted definition of semantics in GP literature: the semantics of a program T_i is defined as the vector of outputs $\mathbf{s}_i = [T_i(\mathbf{x}_1), T_i(\mathbf{x}_2), \ldots, T_i(\mathbf{x}_n)]$ obtained after executing the program (or candidate solution) on a set of training data $\mathbb{T}$ [13].

While the semantics of a program determines what a program actually does, traditional genetic operators manipulate programs by only considering their syntax. Hence, traditional GP operators ignore the information regarding the behavior of programs provided by their semantics. The drawback of this choice is that it is difficult (or even impossible) to predict the effect that modifications of the syntax will have on the semantics of programs. To overcome this limitation, new genetic operators that act on the semantics of programs have been recently proposed [13].

Consider the typical GP problem of finding a function that maps input data into known target outputs (as we said, regression and classification are particular cases). The fitness of an individual for this problem is typically a distance between its predicted output values and the expected ones, i.e. an error measure. In this sense, the optimal solution will have an error equal to zero; i.e. the optimal, or target, semantics is the vector of desired outputs $\mathbf{t} = [t_1, t_2, \ldots, t_n]$. Hence, there is a single global optimum toward which the search is expected to progress [19]. Geometric semantic operators define transformations on the syntax of the individuals that have precise consequences on the semantics of the produced offspring. For instance, offspring are guaranteed to have semantics that are contained within a local neighborhood of the semantics of the parent(s). It has been proven that geometric semantic operators are able induce a unimodal fitness landscape [19] on any problem consisting in finding the match between a set of input data and a set of expected target ones. In this paper, we will consider the definition of geometric semantic operators for real functions, since these are the operators we will use in the experimental phase. In particular, *geometric semantic crossover* (GSC) generates the expression

$$T_{XO} = (T_1 \cdot T_R) + ((1 - T_R) \cdot T_2) \tag{2}$$

as the unique offspring of parents $T_1, T_2 : \mathbb{R}^n \to \mathbb{R}$, where T_R is a random real function whose output values range in the interval [0, 1]. Analogously, *geometric semantic mutation* (GSM) returns the expression

$$T_M = T + ms \cdot (T_{R1} - T_{R2}) \tag{3}$$

as the result of the mutation of an individual $T : \mathbb{R}^n \to \mathbb{R}$, where T_{R1} and T_{R2} are random real functions with codomain in [0, 1] and ms is a parameter called mutation step.

Moraglio et al. [13] show that GSC corresponds to geometric crossover in the semantic space (i.e. the point representing the offspring stands on the segment joining the points representing the parents) and GSM corresponds to ball mutation in the semantic space (and thus induces a unimodal fitness landscape). As Moraglio et al. point out, these operators create much larger offspring than their parents and the fast growth of the individuals in the population rapidly makes fitness evaluation unbearably slow. In [20,21], a possible workaround to this problem was proposed, consisting in an implementation of Moraglio's operators that makes them not only usable in practice, but also very efficient. This implementation is based on the idea that, besides storing the initial trees, at every generation it is enough to maintain in memory, for each individual, its semantics and a reference to its parents. As shown in [20], the computational cost of evolving a population of n individuals for g generations is $O(ng)$, while the cost of evaluating a new, unseen, instance is $O(g)$.

The GP algorithm that uses geometric semantic operators to produce new candidate solutions will be called geometric semantic GP (GSGP) from now on. GSGP can be used to build arbitrary function approximations using a series of incremental steps, through crossover and mutation. This contrasts, for instance, with the well-known multi-layered perceptron (MLP), that takes a holistic approach, where the general solution form (topology of the network) is defined a priori, and the learning algorithm has the objective of optimizing the internal connection weights. This is basically the difference between a bottom-up approach of locally optimal incremental improvements (GSGP) and a top-down global approach (MLP). There are of course incremental ways for building MLPs, such as the popular NEAT algorithm [22], but a comparison and analysis of such methods is beyond the scope of this work.

4. Local search in geometric semantic operators

In this work, we integrate a local search (LS) strategy within GSGP. In particular, we include a local searcher within the GSM mutation operator, since previous works have shown that GSGP often achieves its best performance using only mutation [23]. In particular, the GSM with LS (GSM-LS) of a tree T generates an individual:

$$T_M = \alpha_0 + \alpha_1 \cdot T + \alpha_2 \cdot (T_{R1} - T_{R2}) \tag{4}$$

where $\alpha_i \in \mathbb{R}$ and α_2 replaces the mutation step parameter ms defined in Eq. (3). This in fact defines a basic multivariate linear regression problem, which could be solved, for example, by Ordinary Least Square regression (OLS). However, in this case we have n linear equations, the number of fitness cases, and only three unknowns (α_1, α_2 and α_3). This gives an overdetermined multivariate linear fitting problem, which can be solved through SVD.[1] We argue that this should be seen as a LS operator, that attempts to determine the best linear combination of the parent tree and the random trees used to perturb it. In this context, the term *local* is used in the perspective of the linear problem posed by the GSM operator. In some senses, this approach contrasts with previous work [14], that relied on a non-linear local optimizer, since the linear assumption is mostly not satisfied by the expression evolved with standard GP and the corresponding parametrization. On the other hand, in this new approach, it is simple to apply a linear regression optimizer, given that the GSM operator defines a linear expression in the parameter space.

The idea of including a LS method is based on a very simple observation related to the properties of the geometric semantic

[1] In this work, the GNU Scientific Library is used http://www.gnu.org/software/gsl/.

operators: while these operators are effective in achieving good performance with respect to standard syntax-based operators, they require a lot of generations to converge to optimal solutions. Including a local search method, we expect to improve the convergence speed of the search algorithm.

5. Linear scaling

Linear scaling, first introduced for improving GP in symbolic regression in [24], consists in modifying the fitness function by calculating the slope and intercept of the formula coded by a GP individual. Given that $y_i = T(\mathbf{x}_i)$ is the output of the GP individual T on the input $\mathbf{x}_i$, linear scaling calculates the following expressions:

$$b = \frac{\sum_{i=1}^{m}[(y_i - \bar{y})(t_i - \bar{t})]}{\sum_{i=1}^{m}(t_i - \bar{t})} \quad (5)$$

$$a = \bar{t} - b\bar{y} \quad (6)$$

where m is the number of fitness cases (i.e. the number of instances in the training set) [25] and $\bar{y}$ and $\bar{t}$ denote the average output and the average target value, respectively. These expressions respectively calculate the slope and intercept of a set of outputs y, such that the sum of the squared errors between t and $a + by$ is minimized. After this, any error measure can be calculated on the scaled formula $a + by$, for instance the Root Mean Square Error (RMSE):

$$RMSE(t, a + by) = \sqrt{\frac{\sum_{i=1}^{m}(a + by_i - t_i)^2}{m}} \quad (7)$$

If a is different from 0 and b is different from 1, the procedure outlined above is guaranteed to reduce the *RMSE* for any formula $y = T(\mathbf{x}_i)$ [24]. In all the experiments discussed in this paper, a and b have always been calculated using only training data. Then, these values of a and b have been used to calculate the re-scaled RMSE on unseen test data. In this way, the cost of calculating the slope and intercept is linear in the size of the training set. By efficiently calculating the slope and intercept for each individual, the need to search for a and b is removed from the GP run. GP is then free to search for that expression whose shape is most similar to that of the target function. The efficacy of linear scaling in GP for many symbolic regression problems has been widely demonstrated [24].

Table 1
Features and number of their possible values

Features	Description	Number of possible values
X1	Relative compactness	12
X2	Surface area	12
X3	Wall area	7
X4	Roof area	4
X5	Overall height	2
X6	Orientation	4
X7	Glazing area	4
X8	Glazing area distribution	6
Y1	Target variable representing the heating load (HL)	586
Y2	Target variable representing the cooling load (CL)	636

6. Experimental study

In this section we report the experimental settings and the results achieved with the proposed GP framework. We also compare the results obtained with the ones reported in [6], where the same dataset has been considered. Section 6.1 reports a brief description of the dataset. Section 6.2 describes the experimental settings and details to allow practitioners to replicate our work. Finally, Section 6.3, discusses the obtained results.

6.1. Data

To test the proposed GP framework, we used the same data as in [6]. In particular, the dataset consists of eight independent variables (or features) and 768 instances. Each instance is related to a particular building. The features are reported in Table 1.

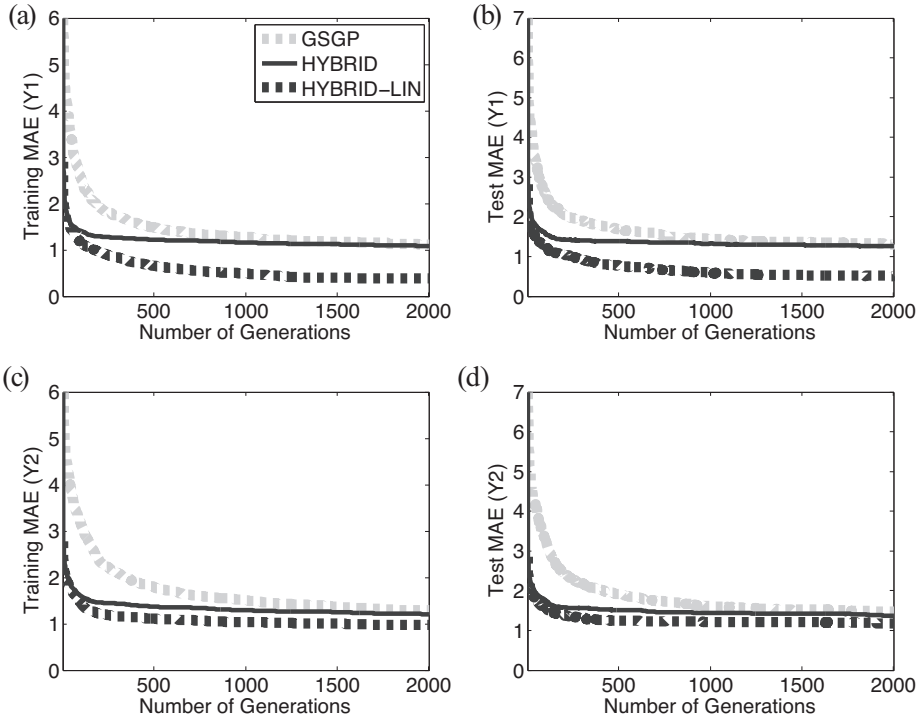

Fig. 2. Training and test MAE for the two targets: Y1 (HL) and Y2 (CL). The plots show the median over 100 runs.

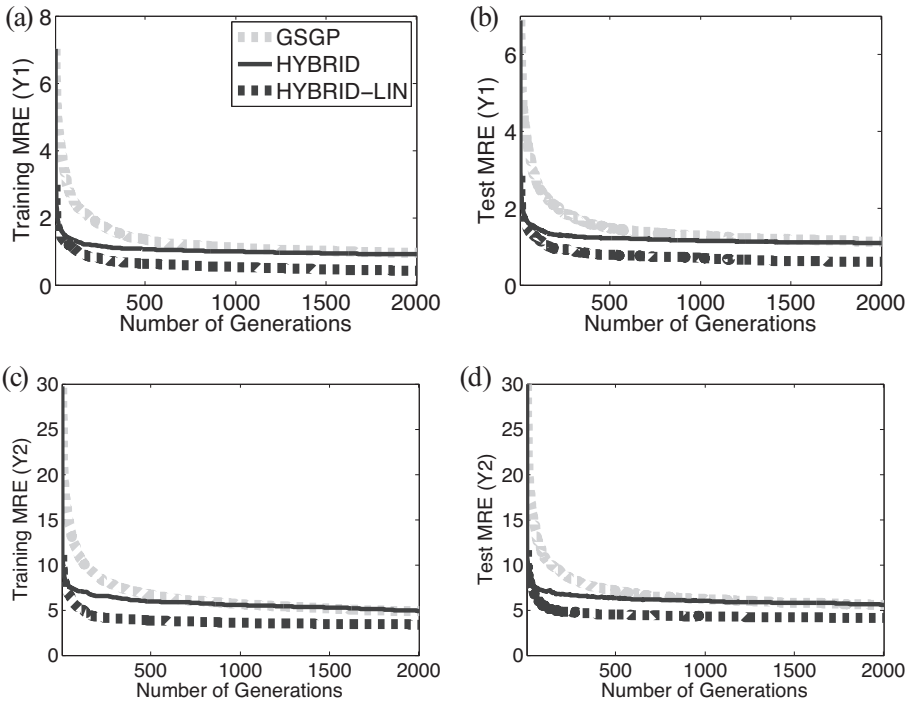

Fig. 3. Training and test MRE for the two targets: Y1 (HL) and Y2 (CL). The plots show the median over 100 runs.

For additional details about the dataset, the reader is referred to [6]. The dataset is freely available at the UCI machine learning repository (http://archive.ics.uci.edu/ml/datasets/Energy+efficiency).

6.2. Experimental settings

Three different GP systems are compared: GSGP, GSGP with local search (HYBRID) and a third system that uses the HYBRID approach integrated with linear scaling (HYBRID-LIN). Moreover, the GP systems were also compared to a set of state-of-the-art machine learning techniques used in [6]. For all the studied systems, we estimate the HL and CL values using a 10-fold cross validation with 100 repetitions. For all the 100 runs, the dataset is randomly split into a training subset with which the learner is trained, and a testing subset used to assess the learner's generalization performance. Typically some percentage of the data is left out for testing the learner, and this is known as *K*-fold cross validation.

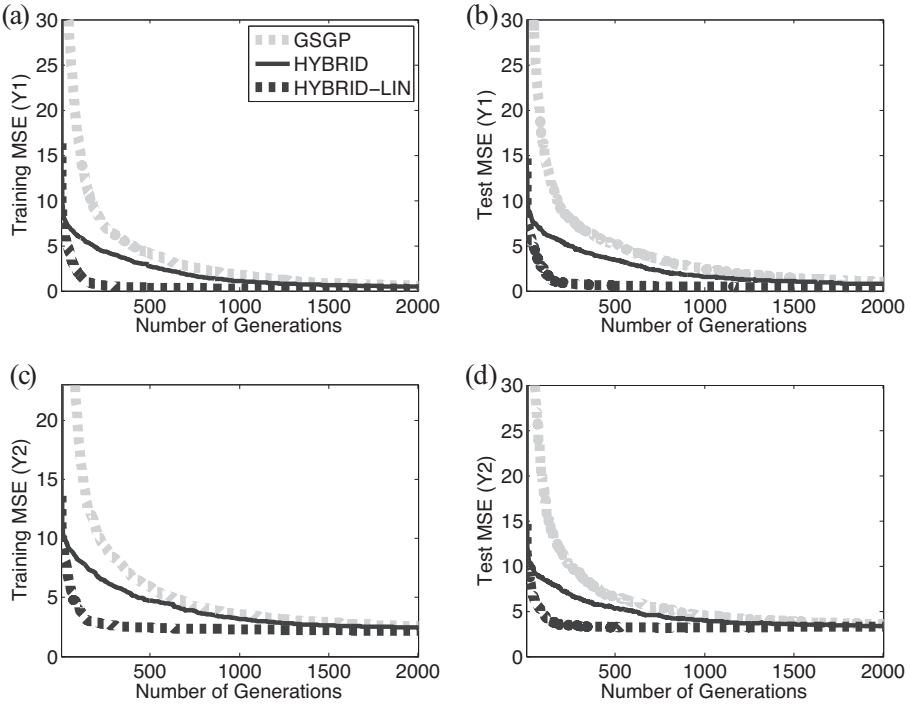

Fig. 4. Training and test MSE for the two targets: Y1 (HL) and Y2 (CL). The plots show the median over 100 runs.

Table 2
Training and test error of the three semantic-based genetic programming systems. Medians over 100 runs are reported.

	Training MAE			Test MAE		
	GSGP	HYBRID	HYBRID-LIN	GSGP	HYBRID	HYBRID-LIN
HL	1.12	1.09	0.38	1.31	1.26	0.51
CL	1.29	1.20	0.97	1.47	1.37	1.18
	TRAINING MRE			TEST MRE		
	GSGP	HYBRID	HYBRID-LIN	GSGP	HYBRID	HYBRID-LIN
HL	0.96	0.91	0.43	1.13	1.09	0.62
CL	4.86	4.92	3.40	5.55	5.58	4.15
	TRAINING MSE			TEST MSE		
	GSGP	HYBRID	HYBRID-LIN	GSGP	HYBRID	HYBRID-LIN
HL	0.60	0.46	0.22	1.02	0.78	0.47
CL	2.60	2.44	2.16	3.61	3.38	3.33

In this study, as in [6], K is equal to 10. Hence, the training set is used to build the predictive model and the test set is used to assess the performance of the model on unseen examples.

Regarding the GP systems, all the runs used populations of 200 individuals [26] allowed to evolve for 2000 generations. Tree initialization was performed with the Ramped Half-and-Half method [12] with a maximum initial depth equal to 6. The function set contained the arithmetic operators, including protected division as in [12]. The terminal set contained a number of variables that corresponds to the number of features in the dataset. Mutation has been used with probability equal to 1 and no crossover was used. Survival from one generation to the other was always guaranteed to the best individual of the population (elitism). For GSM, a random mutation step has been considered in each mutation event, as suggested in [23]. Regarding the HYBRID and HYBRID-LIN systems, GSM-LS has been used in the first 200 generations, while in the remaining generations the standard GSM was employed. We decided to limit the number of generations where the local search has been used in order to prevent overfitting.

Regarding the state-of-the-art machine learning techniques, we referred to the results reported in [6] where the same analysis, on the same dataset, has been performed. In detail, authors of [6] used two machine learning techniques for predicting the HL and the CL of a set of residential buildings. The two machine learning techniques are *iteratively reweighted least squares* (IRLS) and *random forests* (RF). Authors of [6] studied the performance of the machine learning techniques over three different measures of error and, in this work, we also use the same measures. In particular, these three measures are the mean absolute error (MAE), the mean square error (MSE), and the mean relative error (MRE), where:

$$MAE = \frac{1}{N}\sum_{i\in Q}|t_i - y_i| \tag{8}$$

$$MSE = \frac{1}{N}\sum_{i\in Q}|t_i - y_i|^2 \tag{9}$$

$$MRE = 100\cdot\frac{1}{N}\sum_{i\in Q}\frac{|t_i - y_i|}{t_i} \tag{10}$$

where $y_i = T(\mathbf{x}_i)$ is the output of the GP individual T on the input $\mathbf{x}_i$ and t_i is the target value for the instance $\mathbf{x}_i$. N denotes the number of samples in the training or testing subset, and Q contains the indices of that set.

Table 3
Comparison between errors obtained on unseen examples with different techniques. Three different error measures and two prediction tasks are considered. For all the considered techniques we reported 10th, 25th, 50th, 75th and 90th percentile. **Bold** is used to denote the best (i.e. lower) median value among the considered techniques.

	GSGP	HYBRID	HYBRID-LIN	IRLS	RF
MAE HL					
10th	1.19	0.91	0.44	2.14	0.49
25th	1.26	1.14	0.49	2.14	0.49
50th	1.31	1.26	**0.51**	2.14	**0.51**
75th	1.34	1.35	0.56	2.22	0.57
90th	1.47	1.40	0.67	2.52	0.64
MAE CL					
10th	1.31	1.12	1.08	2.19	1.31
25th	1.38	1.24	1.14	2.20	1.40
50th	1.47	1.37	**1.18**	2.21	1.42
75th	1.61	1.48	1.24	2.25	1.46
90th	1.73	1.61	1.33	2.50	1.61
MRE HL					
10th	1.02	0.87	0.50	9.84	0.99
25th	1.07	1.00	0.52	9.86	1.02
50th	1.13	1.09	**0.62**	9.87	1.03
75th	1.20	1.12	0.65	9.96	1.08
90th	1.25	1.19	0.72	10.09	1.26
MRE CL					
10th	5.01	4.83	4.02	10.67	6.12
25th	5.30	5.26	4.05	11.05	6.43
50th	5.55	5.58	**4.15**	11.46	6.59
75th	5.94	5.97	4.45	11.66	7.06
90th	6.42	6.57	4.66	12.09	7.17
MSE HL					
10th	0.75	0.63	0.33	9.30	1.85
25th	0.82	0.69	0.37	9.74	2.07
50th	1.02	0.78	**0.47**	10.09	2.18
75th	1.27	0.94	0.49	10.69	2.24
90th	1.45	1.17	0.53	10.99	2.61
MSE CL					
10th	2.82	2.84	2.71	8.58	4.14
25th	3.41	3.06	3.079	9.08	4.51
50th	3.61	3.38	**3.33**	9.41	4.62
75th	3.86	3.67	3.50	10.08	4.810
90th	4.27	3.94	3.69	10.93	5.19

6.3. Experimental results

In this section, we report and discuss the obtained results. In particular, we start by comparing the performance of the three GP systems. The objective of this study is to check whether the inclusion of a local search algorithm and the use of linear scaling are beneficial for improving the performance of the algorithm. Results are reported in Figs. 2–4. In all the plots we report the median error calculated over the 100 independent runs. As it is possible to see, for all the error measures, HYBRID is able to outperform GSGP on both training and test instances. In particular, HYBRID requires a smaller number of generations to reach good quality solutions. Moreover, HYBRID-LIN is even better that HYBRID. Hence, using linear scaling in conjunction with local search and geometric semantic operators is an appropriate approach for the problem at hand.

Results shown in Figs. 2–4 are summarized in Table 2.

To analyze the statistical significance of these results, a set of tests has been performed on the median errors. As a first step, the Kolmogorov–Smirnov test has shown that the data are not normally distributed and hence a rank-based statistic has been applied. Successively, the Wilcoxon rank-sum test for pairwise data comparison (with Bonferroni correction) has been used under the alternative hypothesis that the samples do not have equal medians. According to this test, results produced by HYBRID-LIN are always consistently statistically better (i.e. with lower errors) than the ones produced

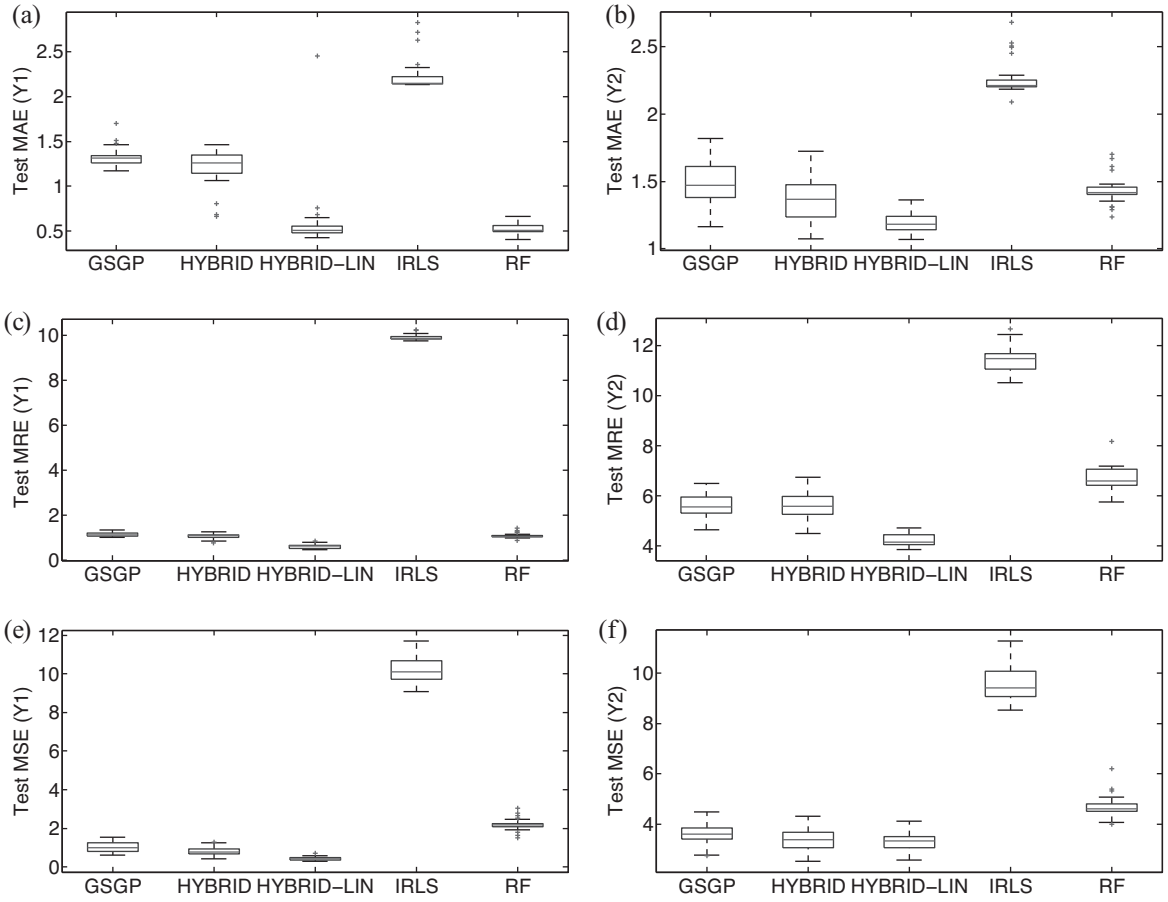

Fig. 5. Test error for the two targets: Y1 (HL) and Y2 (CL). The plots show the median over 100 runs. On each box, the central mark is the median, the edges of the box are the 25th and 75th percentiles, and the whiskers extend to the most extreme data points not considered outliers.

by GSGP on both training and test instances. Regarding the HYBRID system, the performance is comparable to or better than the one of GSGP. In detail, results on unseen instances are comparable except for the prediction of CL using MSE as error measure. In this case, the HYBRID system outperforms GSGP. On the training instances, the two techniques perform in a comparable way, except for the prediction of HL when MAE and MSE are the error measures. In these cases, the HYBRID system outperforms GSGP. Finally HYBRID-LIN produces results that are statistically better than the ones achieved with the HYBRID system, except for the prediction of CL on unseen instances using MAE and MSE as error measures (p-value = 0.066 and p-value = 0.305, respectively).

To summarize, the proposed idea for improving the performance of GSGP is effective. In particular, the use of a local search method allows GSGP to converge in a smaller number of generations, while the integration of linear scaling and local search allows GSGP to converge in a small number of generations and also to produce quantitatively better results.

To conclude this section, we report in Table 3 the results achieved on the out-of-sample instances with the HYBRID-LIN system and the results obtained with the machine learning techniques studied in [6]. Finally, Fig. 5 reports the boxplots of the error on unseen instances for each one of the considered error measures and for all the studied techniques. On each box, the central mark is the median, the edges of the box are the 25th and 75th percentiles, and the whiskers extend to the most extreme data points not considered as outliers. As it is possible to see, the proposed HYBRID-LIN system produces better results than the other considered techniques for all the studied error measures. This is an important results, considering that the techniques used in [6] represent the state-of-the-art for the prediction of HL and CL.

7. Conclusions

Optimizing energy performance is considered a crucial step for policy makers in various economies across the globe. In this work, an artificial intelligence system for the prediction of energy performance of residential buildings has been proposed. The system (called geometric semantic genetic programming, or GSGP) is based on a variant of the genetic programming (GP) algorithm. In particular, GSGP uses genetic operators, called geometric semantic operators, that exploit semantic awareness. In this work, we have integrated GSGP with two additional components: a local searcher and linear scaling. The utilization of a local searcher is motivated by the attempt to improve the convergence speed of GSGP. The integration of the local searcher with linear scaling is motivated by the attempt to achieve greater effectiveness [24]. To assess the performance of the resulting system (called HYBRID-LIN) a large set of experiments has been performed. In particular, the same dataset used in the previous study [6] has been used. The objective of the predictive models, built with the proposed system, is the prediction of the heating load and cooling load of a set of residential buildings. Experimental results, where three different error measures have been considered, have shown the suitability of the proposed system. In particular, the proposed system outperforms existing state-of-the-art techniques. To summarize, the paper provides two contributions: from the point of view of the energy performance prediction, a system able to outperform existing state-of-the-art

techniques has been defined; from the machine learning perspective, this case study has shown that integrating a local searcher and linear scaling in GSGP can speed up the convergence of the search process.

Acknowledgment

The authors Mauro Castelli and Leonardo Vanneschi gratefully acknowledge project MassGP (PTDC/EEI-CTP/2975/2012), FCT, Portugal.

References

[1] Z. Yu, F. Haghighat, B.C. Fung, H. Yoshino, A decision tree method for building energy demand modeling, Energy Build. 42 (10) (2010) 1637–1646.
[2] W. Cai, Y. Wu, Y. Zhong, H. Ren, China building energy consumption: situation, challenges and corresponding measures, Energy Policy 37 (6) (2009) 2054–2059.
[3] UNEP, United Nations Environment Programme, Sustainable Building and Climate Initiative. URL http://www.unep.org/publications/
[4] H. Chappells, E. Shove, Debating the future of comfort: environmental sustainability, energy consumption and the indoor environment, Build. Res. Inf. 33 (1) (2005) 32–40.
[5] European Commission, Directive 2002/91/EC of the European Parliament and of the Council of 16th December 2002 on the energy performance of buildings, 2003.
[6] A. Tsanas, A. Xifara, Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools, Energy Build. 49 (0) (2012) 560–567.
[7] T. Catalina, J. Virgone, E. Blanco, Development and validation of regression models to predict monthly heating demand for residential buildings, Energy Build. 40 (10) (2008) 1825–1832.
[8] C. Ghiaus, Experimental estimation of building energy performance by robust regression, Energy Build. 38 (6) (2006) 582–587.
[9] L. Zhou, F. Haghighat, Optimization of ventilation system design and operation in office environment, Part I: Methodology, Build. Environ. 44 (4) (2009) 651–656.
[10] L. Magnier, F. Haghighat, Multiobjective optimization of building design using TRNSYS simulations, genetic algorithm, and artificial neural network, Build. Environ. 45 (3) (2010) 739–746.
[11] J. Zhang, F. Haghighat, Development of artificial neural network based heat convection algorithm for thermal simulation of large rectangular cross-sectional area earth-to-air heat exchangers, Energy Build. 42 (4) (2010) 435–440.
[12] J.R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, Cambridge, MA, USA, 1992.
[13] A. Moraglio, K. Krawiec, C.G. Johnson, Geometric semantic genetic programming, in: C.A. Coello Coello, V. Cutello, K. Deb, S. Forrest, G. Nicosia, M. Pavone (Eds.), Parallel Problem Solving from Nature, PPSN XII (Part 1), Vol. 749, of Lecture Notes in Computer Science, Springer, 2012, pp. 21–31.
[14] E. Z-Flores, L. Trujillo, O. Schuetze, P. Legrand, Evaluating the effects of local search in genetic programming, in: A.-A. Tantar, et al. (Eds.), EVOLVE – A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation V, No. 288 in Advances in Intelligent Systems and Computing, Springer, 2014, pp. 213–228.
[15] J.R. Koza, Human-competitive results produced by genetic programming, Genet. Program. Evol. Mach. 11 (3–4) (2010) 251–284.
[16] M. Castelli, L. Vanneschi, S. Silva, Semantic search-based genetic programming and the effect of intron deletion, IEEE Trans. Cybern. 44 (1) (2014) 103–113.
[17] L. Beadle, C.G. Johnson, Semantically driven mutation in genetic programming, in: A. Tyrrell (Ed.), 2009 IEEE Congress on Evolutionary Computation, IEEE Computational Intelligence Society, IEEE Press, Trondheim, Norway, 2009, pp. 1336–1342.
[18] L. Vanneschi, M. Castelli, S. Silva, A survey of semantic methods in genetic programming, Genet. Program. Evol. Mach. 15 (2) (2014) 195–214.
[19] P. Stadler, Towards a theory of landscapes, in: R. Lpez-Pea, H. Waelbroeck, R. Capovilla, R. Garca-Pelayo, F. Zertuche (Eds.), Complex Systems and Binary Networks, Vol. 461 of Lecture Notes in Physics, Springer, Berlin, Heidelberg, 1995, pp. 78–163.
[20] L. Vanneschi, M. Castelli, L. Manzoni, S. Silva, A new implementation of geometric semantic GP and its application to problems in pharmacokinetics, in: K. Krawiec, A. Moraglio, T. Hu, A.S. Uyar, B. Hu (Eds.), Proceedings of EuroGP, LNCS, Springer, 2013, pp. 205–216.
[21] M. Castelli, S. Silva, L. Vanneschi, A C++ framework for geometric semantic genetic programming, Genet. Program. Evol. Mach. 16 (1) (2015) 73–81.
[22] K.O. Stanley, R. Miikkulainen, Evolving neural networks through augmenting topologies, Evol. Comput. 10 (2) (2002) 99–127.
[23] L. Vanneschi, S. Silva, M. Castelli, L. Manzoni, Geometric semantic genetic programming for real life applications, in: R.L. Riolo, J.H. Moore, M.E. Kotanchek (Eds.), Genetic Programming Theory and Practice XI, Springer, 2013, pp. 191–209.
[24] M. Keijzer, Improving symbolic regression with interval arithmetic and linear scaling, in: C. Ryan, T. Soule, M. Keijzer, E. Tsang, R. Poli, E. Costa (Eds.), Genetic Programming, Proceedings of EuroGP'2003, Vol. 2610 of LNCS, Springer-Verlag, Essex, 2003, pp. 70–82.
[25] M. Giacobini, M. Tomassini, L. Vanneschi, Limiting the number of fitness cases in genetic programming using statistics, in: J. Guervs, P. Adamidis, H.-G. Beyer, H.-P. Schwefel, J.-L. Fernndez-Villacaas (Eds.), Parallel Problem Solving from Nature PPSN VII, Vol. 2439 of Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2002, pp. 371–380.
[26] R. Poli, N.F. McPhee, L. Vanneschi, The impact of population size on code growth in GP: analysis and empirical validation, in: M. Keijzer, et al. (Eds.), GECCO '08: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation, ACM, 2008, pp. 1275–1282.