SN Computer Science (2022) 3:119
https://doi.org/10.1007/s42979-021-00998-7

ORIGINAL RESEARCH

An Empirical Study of Progressive Insular Cooperative GP

Karina Brotto Rebuli[1] · **Leonardo Vanneschi**[2]

Received: 30 July 2021 / Accepted: 15 December 2021 / Published online: 7 January 2022

Abstract
Genetic programming (GP) is a general purpose artificial intelligence method, that breeds populations of computer programs to solve a given problem, mimicking the principles of Darwinian evolution. Among the several different uses, it can be employed for supervised machine learning, interpreting the evolving programs as predictive models. With the objective of improving GP for multi-class classification, in this paper we model a feature of biological evolution: the structuring of populations into sub-populations, or demes. In particular, we present the progressively insular cooperative GP (PIC GP), in which classification is performed by applying two stages, in two different co-evolving sub-populations: a population, called population of specialists, aimed at optimizing the learning for the different classes, and a population, called population of teams, in which specialists are joined and the evolution allows us to obtain the final predictive model. By means of three simple parameters, PIC GP can tune the amount of cooperation between specialists of different classes. Preliminary experiments indicate that PIC GP achieves the best performance when the evolution begins with a high level of cooperation between specialists of different classes, and then this type of cooperation is progressively decreased, until only specialists of the same class can cooperate between each other. In this paper, we compare PIC GP with some state-of-the-art classification algorithms on a rich set of test applications. The obtained results show that PIC GP is highly competitive with the best algorithms, outperforming the majority of its competitors on the studied problems.

Keywords Multiclass classification · Genetic programming · Cooperative evolution

Introduction

Genetic programming (GP) [1] breads a population of computer programs (individuals) by mimicking the principles of Darwin's theory of evolution. At each step, stochastic operators of crossover and mutation are applied to generate new individuals, the most promising of which are probabilistically selected for the next generation. This selection step emulates Darwinian natural selection, introducing into GP the ecological relationship of competition. In classification tasks, GP individuals (classifiers) usually compete to generate a discriminating function that best separates the instances of each target class. However, in the Darwin's theory other ecological relationships besides competition are important. Among them, cooperation, a mutually beneficial interaction between species [2], is essential. This paper presents an empirical study of the progressive insular cooperative GP (PIC GP), a modified GP algorithm in which cooperation between solutions is applied to solve multi-class classification (MCC) problems (i.e. for classification tasks with three or more target classes). To use GP for MCC, the very first decision to be made is on how to discriminate the classes, all at once or in pairs. In practical terms, addressing this question means having either a single classifier to handle the entire classification task or to have as many classifiers as the number of classes to be modelled. Very few successful studies have been published so far using the first approach, in which a single solution has to deal with the whole complexity of the classification

This article is part of the topical collection "Genetic Programming" guest edited by Aniko Ekart, Ting Hu and Nuno Lourenço.

✉ Karina Brotto Rebuli
karina.brottorebuli@unito.it

[1] Dipartimento di Scienze Veterinarie, Università degli Studi di Torino, Largo Paolo Braccini 2, 10095 Turin, Italy

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

task. This may be one of the reasons why GP is currently not considered a top algorithm for MCC, as clearly stated, for instance, in [3]. However, if we consider the second approach, the standard GP process should be changed from its basic design to evolve solutions joining more than one program, since one different classifier is needed for each target class. PIC GP uses this approach, having individuals divided in subpopulations according to the target class they are specialised in classifying. We call these individuals *specialists*. Specialists are grouped in *teams*, higher-level solutions that combine specialist outcomes to give the univocal algorithm prediction. Because the evolution of specialists alone is not always enough to produce good teams, the teams themselves are evolved in a separate population. Consequently, in this approach, the GP evolution works in an upgraded two-level design, one level for the specialists and another for the solution that join the outcomes of these classifiers. In addition to the competition relationship, typical of natural selection, the two-level design of PIC GP presents an opportunity for the introduction of *cooperation* between individuals of different specialisations. The cooperation is present in a team-based GP only if specialised individuals are allowed to interact over the evolution process. Simply joining them into a team, in itself, is not a cooperative, but rather a collaborative action, since the specialised individuals work together, but do not benefit individually from it. In PIC GP, the specialists can evolve with different levels of cooperation, which can be modified by tuning three simple parameters that are presented in more detail in Sect. 3.1.4. Following the terminology used in [4], specialist subpopulations are called *demes* when the level of cooperation between specialists is high, and *islands* when it is low or inexistent. The proposed method is called PIC GP because the algorithm begins with demes (high cooperation between specialists) that can be progressively transformed into islands (no cooperation at all). Our objective in this work was to investigate the dynamics of PIC GP, by studying how parameters affect its evolution. The manuscript is organised as follows: Sect. 2 presents a review of previous and related GP methods for MCC. Section 3 explains the functioning of PIC GP. Section 4 describes our experimental study, first presenting the experimental settings and the employed test problems, and then discussing the obtained results. Finally, Sect. 5 concludes the paper and proposes ideas for future research.

GP for Multiclass Classification

In some previous work on MCC, GP was used as a wrapper method for data preprocessing, for posterior classification performed by other algorithms [3, 5–8]. Other strategies use GP directly for dealing with the classification problem, without any posterior classifier procedure. When discriminating three or more classes, the very first decision is how to separate them. Three main possibilities have been investigated so far: all-vs-all, pairwise or one-vs-all.

All-vs-all The all-vs-all strategy is the most simple extension of the binary classification approach. In this strategy, a single GP solution is generated and $K-1$ thresholds are applied to its outcomes for a K classes problem. A single model must therefore be able to discriminate among all classes. For instance, Zhang and Smart [9] proposed a single classifier with $K-1$ thresholds, dynamically evolved during the GP run. The same authors, in [10], used properties of Gaussian distributions of the classes to dynamically define the $K-1$ thresholds of the GP multiclassifier solution. Usually, this approach is less likely to produce good models, since it has to handle all the problem complexity at once.

Pairwise. In the pairwise strategy, the problem of classifying K classes is decomposed into K classifiers, each one trained contrasting one target class with the others, in pairs. Thus, we can imagine the training dataset represents $K \times (K-1)/2$ binary classification problems. The final algorithm result is given by a combination of the predictions of the K classifiers. Examples using GP with this strategy can be found in [11] and [12].

One-vs-all In the one-vs-all category, to which PIC GP belongs, the problem of classifying K classes is decomposed into K binary classification problems, contrasting each class with all others at once, to generate K classifiers, one for each target class. The predictions of these K classifiers are then combined by means of an algorithm, able to output the final classification. In GP, these K classifiers can be evolved using one of the following approaches:

(1) *Independent runs approach.* The GP is simply run K times, one for each class, with the dataset split for the corresponding one-vs-all comparison. For example, Lin et al. [13] used the independent runs approach, proposing a multi-layer system with independent GP multi populations for MCC problems. In the last layer, the solutions obtained in the previous layers were combined in a single population, and a single GP solution was produced. Chien et al. [14] used GP with independent runs to generate a rough classification function in which the fitness of the classifier is a mean of the normalised distance between the output and the threshold, considering both the correct and the incorrect classified data instances.

(2) *Same run, different subpopulations approach.* In this approach, the subpopulations can be totally isolated (in this case they are called islands), or they can exchange their individuals (demes). For example, Chen and Lu [15] used an island subpopulation approach, with the final prediction being decided by majority voting among the different models.

(3) *Same run, same population, independent individuals approach.* In this approach, the individuals evolve all in the same population, as in a standard GP implementation, but at each generation they are evaluated and set to be responsible for classifying one of the target classes. Smart and Zhang [16] used this approach for evolving all classifiers in a single GP run, with solutions evaluated for every target class at each generation. For the GP prediction, the data instance was evaluated by all K solutions and was assigned to the class to which it had the highest probability of belonging to.

(4) *Teams approach.* A team can be imagined as a tree, in which the root node combines the results of its members. Each team member is a single threshold binary classifier, specialised in a corresponding class. Both the team and its members evolve in the GP process. Thus, the two-level nature of evolving K classifiers, that are combined into a single GP solution, becomes explicit. Evolving only the specialists can produce strong individuals that perform poorly for the combined prediction. Nevertheless, the specialists should also evolve individually, to be able to improve the output of the team. For that to happen, it is necessary to define their individual evolution criteria, i.e. their individual fitnesses. Therefore, the team's approach creates a new decision requirement, that is to define how the team fitness will be distributed among the team's members. This is called the credit assignment problem [17]. The team outcome will be the class whose specialist member gives a positive result. However, since more than one specialist can give a positive prediction, the team also requires a disambiguation procedure to define which of the positive classes will correspond to its final classification result. Haynes et al. [18] published pioneering work using the team's approach with Strongly Typed GP. Their focus was in the role of crossover in making team populations evolve in coordination. The presented crossover operator essentially controlled if individuals specialised in a target class could exchange genetic material with individuals specialised in other target classes. In a later and more complete publication, Haynes and Sen [19] explored more widely this idea and concluded that the crossover that allowed the exchange of genetic material among random different specialists was advantageous for the studied problem. Muni et al. [20] proposed a Team-based GP for MCC in which the specialists evolve by exchanging parts of their structure, i.e. doing crossover, only with individuals specialised in the same class. Besides that, the algorithm has two interesting features: each specialists has its own threshold and, to speed up the algorithm, the learning phase is made with a step-wise strategy, in which not all data observations are used in each generation.

Some authors have also used hybrid methods, like we do in this work. For instance, in [17], the authors applied the teams approach together with the demes subpopulation approach for two binary classifications and a regression problem with Linear GP. Lichodzijewski and Heywood [21] presented a mixed independent individuals and team approach in a GP that evolves the training subset, the individual binary classifiers and the team, each in a separate evolution process. Soule and Komireddy [22] also presented a mixed independent individuals and teams approach, in which specialist individuals evolved in islands and replaced team's members. Thomason and Soule [23] presented a variation of this idea, in which teams are selected and replace individuals in islands.

Generally speaking, it is expected that the cooperation between specialists will favour the search space exploration and that the competition will favour the search space exploitation. In a traditional GP, the balance between exploration and exploitation is carried on mainly by crossover and mutation rates and the selection pressure. In a cooperative GP, the interaction among specialists can also help to control this balance. In a Linear GP study, Luke and Spector [24] found that restricting the interaction to individuals of the same specialisation improved the algorithm performance. Soule [25] studied a GP regression problem and concluded that heterogeneity among teams is necessary but not sufficient, while individuals' high specialisation, that is related with heterogeneity, is key for improving the algorithm performance. Nevertheless, it is still an open issue how to benefit from the balance between cooperation and the evolution of highly specialised individuals, to properly explore and exploit the search space, and this is the main motivation for our work. In the present study, variations in the level of specialists cooperation over time were explored. Individuals can be distributed in class-based demes, work fully as islands or begin the algorithm in demes and progressively be detached into islands. Thus, by controlling the level of cooperation among specialists, exploration can be favoured in the beginning of the evolution and exploitation can be intensified as the algorithm evolves.

Progressive Insular Cooperative GP

The system studied in this work, Progressively insular cooperative (PIC) GP, is a one-vs-all mixed individuals and teams approach for cooperative MCC. Algorithm 1 describes with pseudocode the main steps of PIC GP. The implementation was made in `Python 3.9`, with the `numpy`, `scipy` and `ski_measures` libraries. Its general structure is exemplified in Fig. 1, for a 3-class classification problem, consisting in categorising data observations into classes C_1, C_2 and C_3.

The system is composed of two main populations, evolving at the same time: the population of specialists and the population of teams. The population of specialists is, in turn,

partitioned into a number of subpopulations equal to the number of target classes. Each one of these subpopulations i contains individuals specialised in the classification of class C_i. The level of cooperation between specialists of different subpopulations is controlled by only two parameters: the cooperation intensity rate (CIR) and the generation in which the algorithm should convert the specialised subpopulations from demes into islands. A third parameter, the CIR decrease rate, adjusts the CIR over the algorithm evolution. For example, the subpopulations of specialists can begin as demes (with high level of cooperation between specialists), but further in the evolution they can become islands (independent and isolated subpopulations, with no cooperation between them). The fact that the level of cooperation among specialists over the algorithm evolution is entirely customisable is one of key features of PIC GP and it is one the main novelties of this work. Additionally, in PIC GP, the selection method of the specialists is different from standard GP, in order to work with two parents at a time and foster cooperation. A team in PIC GP contains one specialist from each of the subpopulations and combines the results of the classifiers of all target classes. The evolution of the teams is also modified from the standard GP. At each generation, as the teams evolve, its population is replaced half by teams' offspring and half by new teams composed of evolved specialists. The following subsections describe in detail how each step of PIC GP works.

Algorithm 1 Pseudocode of PIC GP. At each generation, the specialists population is updated with its offspring, which can be made by parents of the same or different specialisation classes. And the teams population is updated half by its offspring and half by new teams made with specialist individuals that come from the updated population of specialists.

set:
  G # *number of generations*
  N_{sp} # *specialists population size*
  N_{team} # *teams population size*
  K # *target classes*
Initialise pop_{sp}
Initialise pop_{team}
for g in G **do**:
  # *New generation*
  instantiate $population'_{sp}$, $population'_{team}$
  # *Specialists population update*
  while $population'_{sp}$ size $< N_{sp}$ **do**:
    set k # *specilist subpopulation with less individuals*
    select $parent_{sp1}$ from specialisation k
    select $parent_{sp2}$ from the whole $population_{sp}$
    make crossover and/or mutation
    add $offspring_{sp1}$ into $population'_{sp}$
    if $population'_{sp}$ size $< N_{sp}$ **then**:
      add $offspring_{sp2}$ into $population'_{sp}$
    end if
  end while
  apply elitism in $population'_{sp}$
  $population_{sp} \leftarrow population'_{sp}$
  # *Teams population update*
  while $population'_{team}$ size $< int(N_{team}/2)$ **do**:
    select $parent_{team1}$ and $parent_{team2}$ from $population_{team}$
    make crossover and/or mutation
    add $offspring_{team1}$ into $population'_{team}$
    if $population'_{team}$ size $< N_{team}$ **then**:
      add $offspring_{team2}$ into $population'_{team}$
    end if
  end while
  while $population'_{team}$ size $< N_{team}$ **do**:
    instantiate t # *new team object*
    for k in K **do**:
      select sp_k # *a specialist for class k*
      insert sp_k into t
    end for
    add t into $population'_{team}$
  end while
  apply elitism in $population'_{team}$
  $population_{team} \leftarrow population'_{team}$
end for

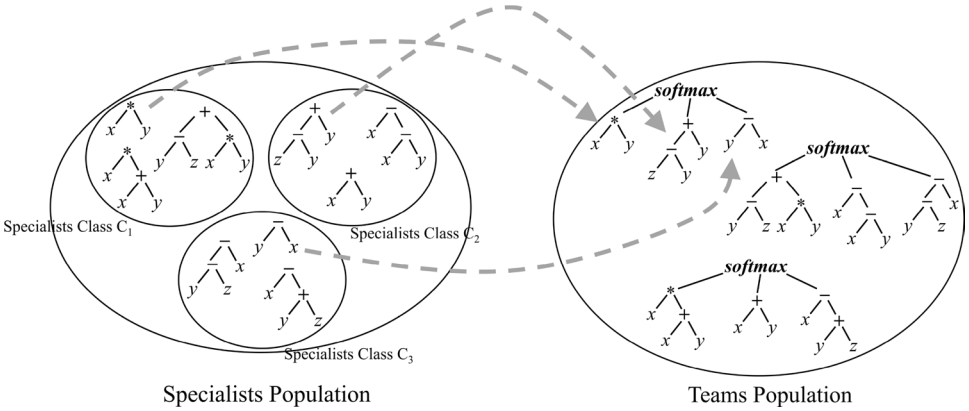

Fig. 1 A simplified graphical representation of the PIC GP system, for an ideal 3-classes classification problem. Two populations (the specialists population and the teams population) are evolved at the same time. The specialists population is, in turn, partitioned into a number of subpopulations equal to the number of classes. Each individual in the teams population contains one specialist coming from each one of these subpopulations.

Specialists Evolution Components

Specialists and Their Fitness

The solutions that we call *specialists* are trees. Each tree is labelled with an attribute that represents the class that this individual should classify in a binary one-vs-all classification task. Let C be that class label. To classify instances, the tree has a logistic function ($S(x) = 1/(1 + e^{-x})$) at its root node and uses the threshold 0.5 for classes discrimination (instances with an output greater than 0.5 are classified as belonging to class C, while the other instances are identified as not belonging to class C), as it is usual, for instance, in Perceptron artificial neural networks. The chosen fitness function for the specialists was the *f-score* measure. This is the harmonic mean of precision and recall rates of a target class, being the precision the proportion of true positive over all positive classifications and the recall, the proportion of true positive over all positive observations in the data. This measure was chosen because, contrarily to accuracy, it is a reliable measure of performance also in presence of unbalanced data. The definition of the class in which individuals are specialised can be done in three ways: (i) it can be simply defined as the class for which the individual has a higher fitness, (ii) it can be assigned by the algorithm to balance the number of specialists in the population, or (iii) it can be the specialisation class of the parents of the solution if the algorithm is in the islands phase.

Specialists Initial Population

Like in traditional GP, the specialists population is initialised using the ramped half-and-half method [1]. Specialists are firstly assigned to the class label for which they work better. Then, to ensure that there will be specialists of all classes in the initial population, individuals in classes with exceeding specialists are relocated. For each class, only the N/K best individuals are kept, where N is the size of the entire specialists population and K is the number of class labels in the dataset. If there are more than N/K individuals specialised in a class, the weaker are randomly changed to other specialisation classes, in which the number of individuals is less than N/K.

Specialist Solutions Selection

If the algorithm is in islands phase, the selection is made with roulette wheel or tournament selection, as in a standard GP. If the algorithm is in demes phase, the specialists selection algorithm works with two individuals at a time. Algorithm 2 shows the PIC GP specialists' selection method for the demes phase. To keep the balance of specialists in the population, the first parent is selected with roulette wheel or tournament selection from a specific deme. The second parent is selected with roulette wheel or tournament over the entire population, but the fitnesses of the specialists are weighted by means of the *cooperation intensity rate* parameter (see the paragraph below). This parameter controls the quantity of cooperation between individuals from different specialisations.

It is worth mentioning that the selection does not determine the class of the second parent. Moreover, it is not guaranteed that the offspring individuals will belong to the same specialisation class as the parents. Consequently, in the end of a generation, the proportion of individuals in each specialisation may change. Despite this, it is enough to control the class of the first parent to keep the number of individuals in specialisation groups approximately balanced.

Algorithm 2 Specialists selection method for demes algorithm phase.

Set:
  k *# the class of the subpopulation with less individuals.*
  `P` *# the specialists population*
  $\texttt{P}_k$ *# the subpopulation of individuals with specilisation class* k.
  `selection_method_1` *# the selection method of the first parent.*
  `selection_method_2` *# the selection method of the second parent.*
  $\texttt{ts}_1$ *# the tournament size for parent 1.*
  $\texttt{ts}_2$ *# the tournament size for parent 2.*
Select one individual from $\texttt{P}_k$ with `selection_method_1`
if `selection_method_2` is roulette wheel **then**:
    for `individual` in the `P` **do**:
        if specialisation class of `individual` is $\neq k$ **then**:
            Recalculate its fitness:
            $f' = f \times \eta$
        end if
    end for
else
    Select $\texttt{ts}_2$ individuals from the entire population
    for `individual` in tournament **do**:
        if specialisation class of `individual` is $\neq k$ **then**:
            Recalculate its fitness:
            $f' = f \times \eta$
        end if
    end for
    Select the individual with higher f' in the tournament
end if

PIC GP Parameters: CIR, CIR Decrease Rate and Phase Change

The main parameter to control the intensity of the cooperation among specialists is the cooperation intensity rate (CIR). It is indicated by η and assumes values in the [0, 1] interval. When the algorithm is in the demes phase, as mentioned in Sect. 3.1.3, the second step of specialists selection selects individuals from the entire specialists population. The CIR is used to lower the fitness of individuals from other specialisations according to Eq. 1.

$$f_i' = \begin{cases} f_i \times \eta_{dec} & \text{if } k_i \neq k_1 \\ f_i & \text{otherwise} \end{cases} \quad (1)$$

where f_i' is the new fitness of specialist i, f_i is the usual fitness of specialist i, η_g is the CIR ate generation g, k_i is the specialisation class of the specialist i and k_1 is the specialisation class of the first parent. Therefore, if CIR is equals to 1.00, the cooperation between different specialists is maximum. As CIR decreases, the fitnesses of the specialists of other classes become smaller and they will have a lower probability of being selected to make crossover with the parent that was selected in the first step of the selection method.

The CIR can be decreased over the evolution by the CIR decrease rate parameter, according to Eq. 2.

$$\eta_g = \eta_{g-1} \times \eta_{\text{dec}}, \quad (2)$$

where η_g is the CIR at generation g, η_{g-1} is the CIR at generation $g-1$ and $\eta_{\text{dec}} \in [0, 1]$ is the CIR decrease rate. The decrease rate reduces constantly, at each generation, the rate of cooperation among specialists. However, if using tournament selection, even with $\eta = 0$ this process cannot convert the algorithm into an island approach. As the first step in tournament selection is purely random, it can happen that only individuals with fitness equal to zero are chosen to take part in the tournament, and so even a zero-fitness individual can be selected. In other words, decreasing the fitness of an individual to zero does not guarantee that the individual will not be selected. Consequently, it is not guaranteed that with $\eta = 0$ individuals with different specialisation will not cooperate. Therefore, to transform the specialists subpopulations from demes to islands, a *phase change* parameter is needed. The value of this parameter corresponds to the generation in which the specialists subpopulations should be transformed from demes to islands. Notice that this parameter also allows a transformation of the demes into islands without having the CIR equals to zero.

With these three parameters (CIR, CIR decrease rate and phase change), the specialised subpopulations can begin the evolution with a defined level of cooperation, that is reduced over the generations, up to a moment in which they do not cooperate anymore.

Specialist Solutions Genetic Operators

For specialists crossover and mutation, in this work the PIC GP uses one point crossover with two offspring and one-point mutation [1]. However, the method is general and, in principle, any existing tree-based GP genetic operator can be used. When the algorithm is in demes phase, the offspring will be assigned to the class for which it works better. So, it does not depend on the parents specialisation class(es). When the algorithm is in islands phase, the specialisation class of the offspring is automatically the same as the specialisation class of the parents.

Teams Evolution Components

Teams Structure and Their Fitness

In PIC GP, a team is a tree with a prediction function at its root node, with arity equal to K, being K the number of target classes in the dataset, and with one specialist of each class (the team members) in each one of the root's subtrees.

The teams fitness used in the present work is the accuracy of the final algorithm classification (Eq. 3).

$$\text{acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{3}$$

where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives and FN is the number of false negatives. The accuracy was chosen to make it possible to compare the PIC GP with the results of other classification algorithms from the literature. Nonetheless, any metric that assesses the performance of a classification algorithm, like the *f-score* used here as the fitness of the specialists, can be used.

Teams Initialisation

The teams population starts with one special team, created deterministically with the elite of each specialists subpopulations (i.e. the best individual in each specialists' subpopulation). The other teams are created with specialists selected with a roulette wheel selection from the specialists' subpopulations.

Teams Genetic Operators

The crossover of teams exchanges entire specialists between parents. At each crossover operation, one class is randomly selected with uniform distribution and the team members of this class are exchanged between the team parents. Mutation substitutes the weaker specialist of the team, i.e. the team member with the lowest fitness, by an individual with the same specialisation selected from the specialists population with a roulette wheel selection.

Teams Prediction

When working with teams for MCC, if only one of its members gives the positive prediction, the specialisation class of this member becomes the team prediction. However, when there is more than one positive result among team members or if none of them provides a positive prediction, a disambiguation procedure is needed. In the present work, the team prediction is given by one of the following options in such cases:

- *softmax*: the logistic outcomes of the team members are standardised in a probability distribution with the softmax function (Eq. 4). The class with highest probability is the team outcome.

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}, \quad j = 1, 2, \ldots, K \tag{4}$$

  where $\mathbf{z}$ is the K-dimensional vector of the specialists logistic outcomes for each i=1,2,...,K.
- *weighted softmax*: the logistic outcomes of the team members are weighted according to the quality of their fitness (Eq. 5).

$$l'_i = \left[(l_i - 0.5) \times \frac{f_i}{\sum_{k=1}^{K} f_k} \right] + 0.5 \tag{5}$$

  where l' is the weighted logistic value of the i_{th} member of the team, l_i is its original logistic outcome, f_i its fitness and sf is the sum of the fitnesses of all team members. These weighted values are standardised in a probability distribution with the softmax function and the class with highest probability is the team outcome. In few words, this team prediction method takes into consideration that not all the team members have the same quality in their predictions. Therefore, before using the specialists logistic outcomes in the softmax function, the logistic outcomes of the team members are weighted by their respective fitness, which is a measure of the quality of their individual prediction.
- *hierarchical*: the binary predictions of the team members are used sequentially, giving precedence to individuals with higher fitnesses, until the first positive result is achieved. The team outcome is the specialisation class of its member with the first positive outcome. This team prediction method also takes into consideration that not all the team members have the

Table 1 Summary of the benchmark datasets from UCI machine learning repository used in the experiments

Dataset	Number of classes	Number of features	Training set size	Validation set size	Test set size
Iris (IRS)	3	4	120	15	15
Page block (PGB)	5	10	4378	547	548
Shuttle (SHT)	7	9	46400	5800	5800
Thyroid (THY)	3	21	5760	720	720
Wine (WNE)	3	13	142	18	18
Yeast (YST)	10	8	1187	148	149

same quality in their predictions. However, it gives more importance to the individual fitness of the team members than the weighted softmax method.

Experimental Study

Test Problems and Experimental Settings

All experiments were run 30 times, each with a different data partition, in which 80% of the observations were selected randomly with uniform distribution to form the training set, 10% to form the validation set and 10% the test set. The validation set was used for parameter tuning (the parameter setting chosen for the final experiments was the one that returned the best results on the validation set), while the (unseen) test set was used to report the final results. This is exactly the same data partitioning scheme as the one used in [26]. Table 1 shows the six MCC datasets from the UCI Machine Learning Repository [27] that were studied.

The IRS was used to explore the PIC GP dynamics and how it is influenced by its parameters. IRS is a widely studied dataset, with 3 target classes (three flower species) and the simplicity of the data structure in this dataset is beneficial to the comprehension of the algorithm dynamics. The target class *setosa* is the easiest to classify, since it is linearly separable from the other two species, based on petal length and width. The *versicolor* is the hardest to classify, because the values of its features are overlapped by the values of the other two classes. The PGB, SHT, THY, WNE and the YST datasets were used to compare PIC GP with other state-of-the-art classification algorithms.

In all runs, the trees were initialised with an initial maximum depth equal to 3. Specialists elitism and teams elitism (i.e. copy of the best individual in these populations, into the next generation, without modification) were always used. The primitive functions used to build the specialists were $+$, $-$, $\times$ and protected $\div$ (the denominator was replaced by the constant $10e^{-6}$ when it was zero). The terminal set was composed by ephemeral constants

Table 2 PIC GP base settings used in the experiments. T2 and T3 are Tournament selection, respectively with sizes 2 and 3, and RW is the Roulette Wheel selection method

PIC GP settings	IRS	PGB	SHT	THY	WNE	YST
Specialists evolution						
Trees maximum depth	6	24	12	10	12	10
Population size	90	250	240	90	200	120
Parent 1 selection	T3	T2	T2	T3	T2	T3
Parent 2 selection	T3	T2	T2	T2	T2	RW
Crossover probability	0.8	0.2	0.2	0.8	0.2	0.8
Mutation probability	0.2	0.8	0.8	0.2	0.8	0.2
Maximum generations	250	250	500	250	500	300
Phase change	200	200	400	200	375	240
Initial CIR	1.00	1.00	1.00	1.00	1.00	1.00
CIR decrease	0.00	$1e^{-4}$	$1e^{-4}$	0.00	$1e^{-4}$	0.00
Teams evolution						
Population size	0	12	24	0	25	0
Crossover probability	0	0.5	0.5	0	0.3	0
Mutation probability	0	0.5	0.5	0	0.7	0
Prediction method	Sfm	Sfm	Sfm	Sfm	Hrc	Sfm

Sfm is the team prediction method called Softmax. Hrc is the team prediction method called Hierarchical

from [0, 1), in addition to the dataset features. The other default parameter settings are presented in Table 2. They were decided in a preliminary experimental tuning phase, choosing the configuration that was able to return the best fitness on the validation set. The modified settings for each experiment on the IRS dataset are presented in the corresponding section.

Experimental Results

Dataset Imbalance

To train the specialists of each class, the full dataset was divided into two parts, the positive cases (with instances that belong to the specialisation class of the individual to be evaluated) and the negative cases (with the instances that do not belong to that target class). One problem that can arise from this procedure is that it may produce training data with significant imbalance between the positive and negative classes. In this scenario, the class with fewer observations is more likely to be misclassified than the class with more observations [28].

When working with teams, the data imbalance created or increased by the one-vs-all approach does not impact directly the final algorithm prediction ability, since the algorithm prediction results from the combination of the specialists outcomes. Only if the data imbalance worsen significantly the specialists performance, it will have an effect on the quality of the final algorithm prediction.

To evaluate this issue, the individual datasets used to train the specialists of each target class were balanced back with undersampling. The experiment was run with all default parameters presented in Table 2. The data balancing procedure did not improved the final algorithm accuracy. Without balancing the individual datasets, the mean accuracy of the 30 runs was 0.967(±0.042) in the test set and with the balanced datasets it was 0.964(±0.049). In addition to the fact that the individual datasets imbalance were not expected to impact the accuracy of the team, two other factors may have contributed to this result: (i) the imbalance in the one-vs-all datasets was modest (33.33% of positive and 66.67% of negative instances) and (ii) the fitness of the specialists used in all experiments was the *f-score*. This measure is the harmonic mean of precision and recall of a target class (see Sect. 3.1.1) and, therefore, it makes the specialists more robust to dataset imbalance.

Specialists Selection Methods

In this set of experiments, we are interested in comparing between each other several different selection methods for the specialists populations. The methods used in these experiments were: tournament of size 3 for the first parent and roulette wheel for the second (T3_R); roulette wheel for the first parent and tournament of size 3

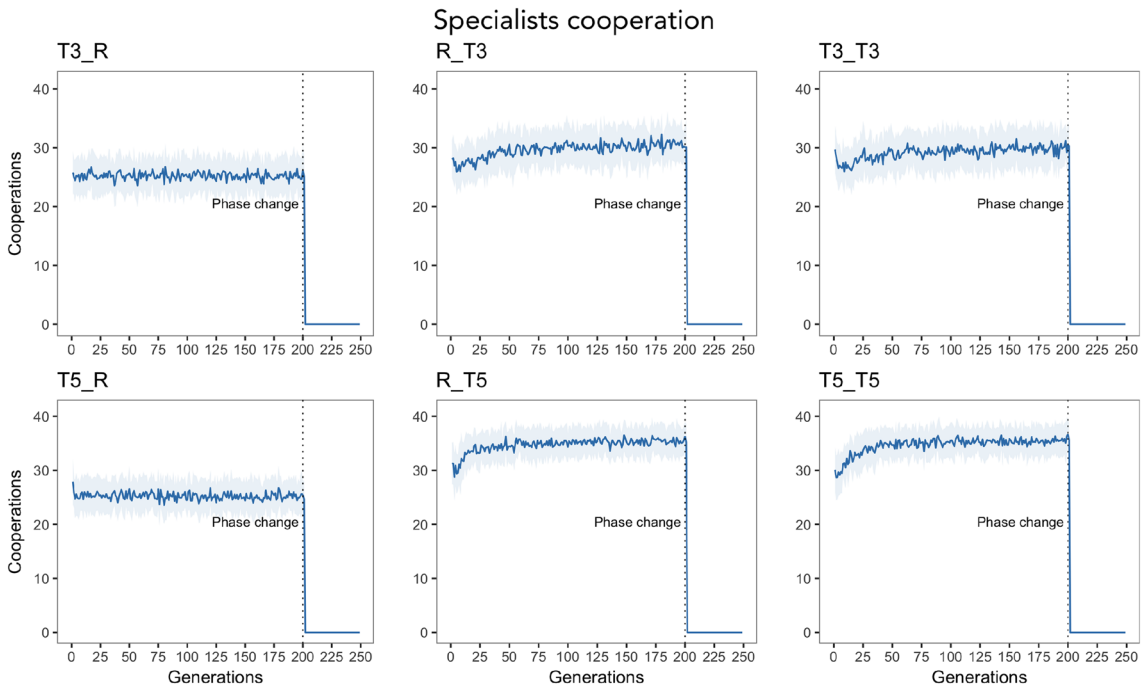

Fig. 2 Effect of the selection method of the second parent on the evolution of mean and standard deviation of the amount of the specialists cooperation (i.e. crossovers between specialists of different classes) against generations, for each specialists selection method

for the second (R_T3); tournament of size 3 for both parents (T3_T3); tournament of size 5 for the first parent and roulette wheel for the second (T5_R); roulette wheel for the first parent and tournament of size 5 for the second (R_T5); and tournament of size 5 for both parents (T5_T5). For all these selection methods, the best accuracy for the test partition was 1.000. The accuracy mean for the test partition was 0.978 ± (0.036sd) for T5_T5, 0.967 ± (0.042 sd) for R_T5 and T3_T3, 0.953 ± (0.056 sd) for R_T3, 0.953 ± (0.047 sd) for T3_R and 0.951 ± (0.068 sd) for T5_R. The differences were not significant (p value 0.267 for a one-way ANOVA test).

The selection methods for first and second parents have different effects in PIC GP. The selection of the first parent just controls the selection pressure inside the subpopulation of one single specialisation class. Besides the selection pressure, the selection of the second parent also controls the cooperation among specialists of different classes. This can be seen in Fig. 2, which shows the evolution of the amount of cooperation between specialists from different classes, for the tested selection methods. More in particular, this figure shows the average and standard deviation of the number of events in which individuals belonging to two different specialists subpopulations are selected for crossover.

Independently of the selection method used for the first parent, the amount of cooperation among specialists of different classes was the same for the same selection method used in the second parent (T3_R is very similar to T5_R; R_T3 to T3_T3; and R_T5 to T5_T5). Moreover, the bigger the tournament for the second parent, the more cooperation among different specialists happened in each generation. The first step of tournament selection is completely random, i.e. it is not related with the individuals' fitnesses. However, the number of individuals in each class subpopulation affects the selection pressure, favouring individuals of the more abundant subpopulation. In addition to the fact that the first parent is chosen to balance the number of individuals among the specialisation classes, favouring the more abundant class in the second parent selection increased the cooperation among classes. Since the *setosa* class is the easiest to discriminate, its specialists tend to have higher fitness and, hence, to be predominant in the population. This can be seen in Fig. 3, which shows the mean of the number of individuals in each class subpopulation for each generation of the T3_T3 and T5_T5 experiments.

These plots show that in the demes phase of the algorithm, for both tournament sizes, the number of *versicolor* specialists tended to decrease, while the number of *setosa* specialists tended to increase. The number of *virginica* specialists tended to decrease for the tournament size 3 and to decrease in the beginning of the evolution process but to increase afterwards for the tournament size 5. So, in a situation under higher selection pressure, the difference in the fitnesses of *versicolor* and *virginica* specialists was more decisive for the selection method outcome.

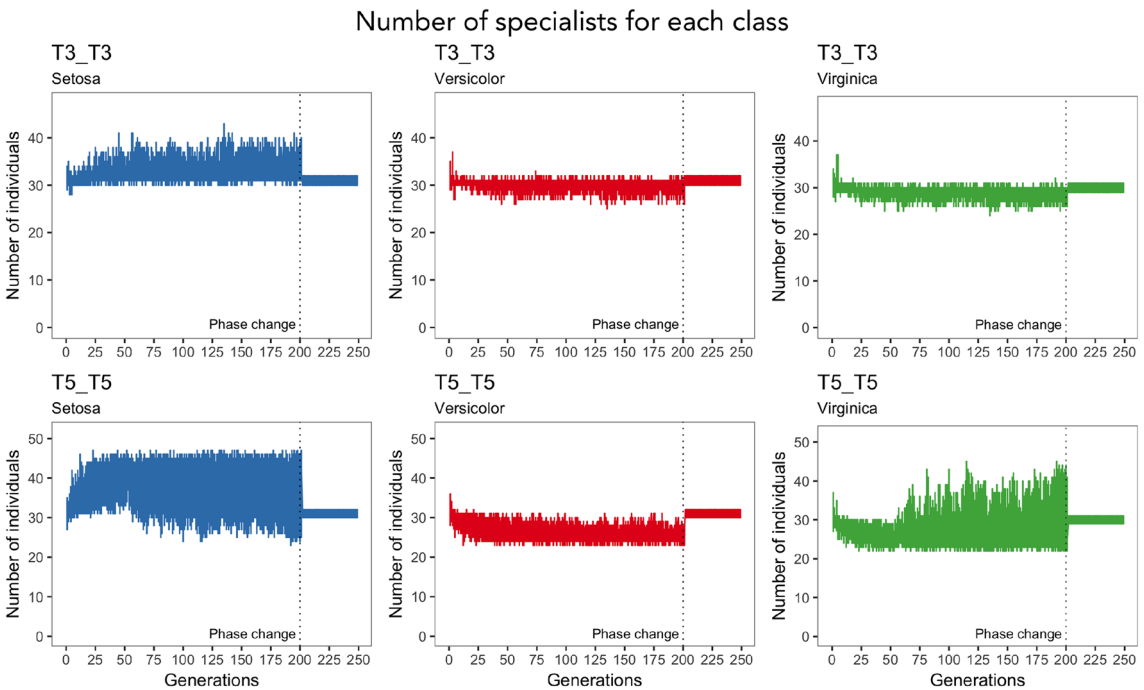

Fig. 3 Evolution of the number of specialised individuals in each class subpopulation against generations for each specialists selection method

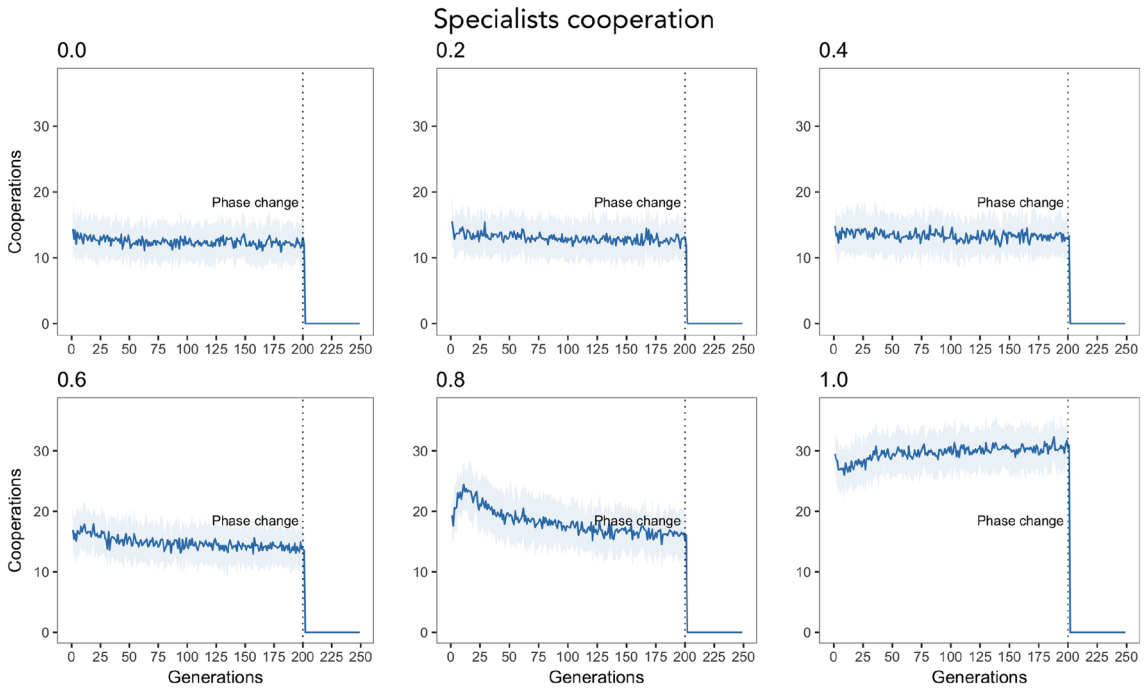

Fig. 4 Evolution of the mean and one standard deviation of the specialists interactions against generations for each CIR values experiment

Cooperation Intensity Rate

The objective of this second part of our experimental study is to understand the influence of the CIR parameter on the dynamics of PIC GP. The following values of CIR were tested: 0.0, 0.2, 0.4, 0.6, 0.8 and 1.0. For all experiments, the CIR was kept constant for the entire PIC GP evolution. The best runs of all CIR experiments achieved accuracy of 1.000 for the test set for CIR0.6 and CIR 0.8. The highest mean accuracy in the test set was obtained for CIR 0.6, 0.973 ± (0.041 sd). Nevertheless, the differences among the accuracies obtained in the CIR experiments were not statistically significant (p value 0.843 for a one-way ANOVA test).

As shown in Fig. 4, the amount of specialists cooperation presented different behavior for different CIR values. The specialists cooperation decreased slightly along the evolution in experiments with CIR from 0.0 to 0.4. For CIR equal to 0.6 and 0.8, they reached a maximum in the early generations, decreasing afterwards. For CIR equal to 1.0, in contrast, they steadily increased along the evolution. In general, the specialists cooperation increased with the increase of CIR. However, this correlation was not linear, because it also depends on the number of individuals in each class specialisation, since tournament was used as selection method. The bigger the difference in the number of individuals among the specialised subpopulations, the more the individuals of different specialisations cooperate when using the tournament selection for the second parent. For CIR values from 0.0 to 0.4, the number of individuals in each specialisation subpopulation was stable and balanced. For CIR

Table 3 Mean of differences between the mean fitnesses of the class subpopulations 50 generations before and 50 generations after the phase change

Class	CIR	Before	After	Diff	CIR	Before	After	Diff
Setosa		0.845	0.913	0.086		0.873	0.897	0.024
Versicolor	0.0	0.671	0.757	0.077	0.6	0.754	0.793	0.039
Virginica		0.741	0.818	0.064		0.780	0.818	0.038
Setosa		0.856	0.910	0.054		0.873	0.893	0.020
Versicolor	0.2	0.743	0.791	0.049	0.8	0.752	0.780	0.028
Virginica		0.780	0.816	0.037		0.802	0.816	0.013
Setosa		0.843	0.902	0.059		0.893	0.892	−0.001
Versicolor	0.4	0.756	0.794	0.038	1.0	0.799	0.811	0.012
Virginica		0.763	0.799	0.035		0.796	0.809	0.014

equal to 0.60 and 0.8m the *setosa* specialists started to prevail at the expense of *versicolor* and *virginica* specialists. But this pattern tended to smooth with the algorithm evolution, more intensely for CIR equal to 0.6 and less for CIR equal to 0.8. For CIR equal tp 1.0, the prevalence of *setosa* individuals lasted for the entire demes phase. The CIR value was also important for the mean fitness of class subpopulations during the evolution: with higher CIR values, i.e. with more cooperation between specialists of different classes, the mean fitness of the class subpopulations increased earlier for all classes. Furthermore, the mean fitness of the specialised subpopulations increased more with the phase change for smaller values of CIR, as Table 3 shows.

Before the phase change, the subpopulations of specialists of the classes *setosa* and *versicolor* had higher mean fitness for CIR equal to 1.0 and the *virginica* class for CIR equal to 0.8. Comparing only the values before the phase change, the weakest class subpopulation (*versicolor*, which is the hardest to separate) presented the greatest difference (0.128) between the mean fitness with CIR equal to 0.0 and with CIR equal to 1.0. For the *virginica* class subpopulation, this difference was 0.055 and for the *setosa*, it was 0.048. After the phase change, *setosa* had the higher mean fitness with CIR equal to 0.0 (0.913), *versicolor* with CIR equal to 1.0 (0.811) and *virginica* with CIR equal to 0.0 and 0.6 (0.818).

Table 3 also shows that the smaller the CIR value, the bigger the difference between the subpopulations mean fitness before and after the phase change. This is due to both smaller mean fitness values before the phase change and higher values after the phase change. Although with CIR equal to 0.0 the mean fitness of the specialised subpopulations increased more with the phase change, for the weakest classifier subpopulation (*versicolor*) the highest mean fitness was reached with CIR equal to 1.0, after phase change. This was not observed for the strongest classifier subpopulation (*setosa*), for which the highest mean fitness was reached with CIR equal to 0.0, after phase change. Thus, a higher CIR value favoured an improvement of the subpopulation of weaker classifiers.

Two interesting conclusions can be drawn from the results presented above. First, before the change of phase (i.e. in the demes phase) the cooperation among the specialists of different classes was helpful, especially for the weaker classifiers. Second, the islands phase is also important for subpopulations to evolve. Both demes and islands seem to be important for achieving the best algorithm performance.

Cooperation Intensity Decrease Rate

Now, we study the influence of the CIR decrease rate on the PIC GP dynamics. The following values of the CIR decrease rates were tested: $8.2e^{-4}$, $5.3e^{-4}$ and $2.6e^{-4}$, to decrease the CIR from 1.00 respectively to 0.85, 0.90 and 0.95 at the end of the demes phase. The constant decrease of the CIR worsened the algorithm final accuracy from 0.967 with no CIR decrease, to 0.920 with final CIR equal to 0.85 (p value equal to 0.017 in a Tukey HSD test for this pair of means). For final CIR values equal to 0.90 and 0.95, the mean was also smaller than for the experiments without CIR decreasing (0.931 and 0.942, respectively), but the difference was not statistically significant.

These results indicate that for the IRS dataset, the maximum cooperation among specialists over the entire demes phase was beneficial.

Phase Change

We now present the experiments aimed at understanding the influence of the phase change on the dynamics of PIC GP. Four values of the phase change were tested:

Fig. 5 Evolution of team fitness using three different teams prediction methods. The solid line is the mean fitness (accuracy) over 30 runs and the envelope is the region of the mean plus and minus one standard deviation

0 (full islands evolution), 125, 200 and 250 (full demes). The accuracy mean was 0.980 ± (0.036 sd) for the phase change at generation 125, 0.967 ± (0.042 sd) for the phase change at generation 200, 0.962 ± (0.049 sd) for the phase change at generation 0 (full islands) and 0.960 ± (0.057 sd) for the phase change at generation 250 (full demes). These results indicate that combining the demes and islands phases can be convenient.

Teams Prediction Method

The teams prediction method defines how the specialists will collaborate to give the final algorithm prediction. Three methods were studied: hierarchical, weighted softmax and softmax, as described in Sect. 3.2.4. They all performed similarly for the IRS dataset: the mean test accuracies using the hierarchical, the weighted softmax and the softmax methods were 0.933 ± (0.129 sd), 0.960 ± (0.045 sd) and 0.967 ± (0.042 sd), respectively. The differences were not significant (p value 0.258 for a one-way ANOVA test). However, it is interesting to notice that the standard deviation of the hierarchical prediction method was much higher than those of the weighted softmax and softmax methods. This can be seen also in Fig. 5, which shows the mean and standard deviation of the fitness of the team solution over the algorithm evolution. It is clear from these plots that the hierarchical prediction gave more unstable results, while the two methods using the softmax function were more consistent, with the pure softmax prediction method providing the most stable results.

The collaboration among team member solutions is a type of ensemble of algorithms. The hierarchical method is the one that prioritises the most the stronger members of the team. Conversely, the other two give the algorithm the chance to create a greater collaboration amongst the team members. It is interesting to observe that the lower the degree of collaboration among team members (hierarchical prediction << weighted softmax < softmax), the highest the variability of the team fitness. These results are expected considering that the collaboration among the team members works like an ensemble of the specialist classifiers and ensembles make the system more robust [29].

Performance Indicators

In addition to the experiments to study the behaviour of the PIC GP in relation to its hyperparameters, in this section we present some indicators of the algorithm performance. Using the IRS dataset as a benchmark, the average time of the train phase was 17 min and 55 s with standard deviation of 2 min and 35 s. The running time of population-based optimisation algorithms tend to be high exactly because there are many solutions to be evaluated over the optimisation process, and this is the case of PIC GP. In addition, the one-vs-all approach increases the number of individuals needed to the optimisation. However, the demes phase allows the algorithm work with less individuals. Thus, even with a high computational cost, the PIC GP can represent a good strategy to improve the GP performance for a one-vs-all approach in MCC problems.

Considering the data of all experiments, the average of trees sizes in the whole population at the last generation ranged from 32.48 to 171.86, with mean 63.04 and standard deviation 15.56. The genotype diversity, evaluated for each subpopulation as the number of different tree structures in a population divided by the number of individuals in the population, ranged from 0.53 to 0.99, with mean 0.83 and standard deviation 0.08. The phenotype diversity, evaluated for each subpopulation as the variance of the fitnesses of the individuals of the whole population, ranged from 0.01 to 0.15, with mean 0.07 and standard deviation 0.03.

Comparison of PIC GP with Other Machine Learning Algorithms

The objective of this final part of our experimental study is to assess the competitiveness of PIC GP, by comparing its performance with a set of state-of-the art classification algorithms. In this experiments, the best accuracies of PIC GP for the PGB, SHT, THY, WNE and YST datasets were compared to other 11 machine learning algorithms

Table 4 Best accuracy for PIC GP and the achieved accuracy for each classifier reported in [26] for the PGB, SHT, THY, WNE and YST datasets

Algorithm	PGB	SHT	THY	WNE	YST
PIC GP	0.978	0.997	0.992	0.941	0.642
SCR	$0.907^{(-)}$	$0.982^{(-)}$	$0.903^{(-)}$	$0.944^{(+)}$	$0.574^{(-)}$
GDBT	$0.889^{(-)}$	$0.995^{(-)}$	$1.000^{(+)}$	$1.000^{(+)}$	$0.622^{(-)}$
KNN	$0.907^{(-)}$	$0.986^{(-)}$	$0.903^{(-)}$	$0.833^{(-)}$	$0.574^{(-)}$
RF	$0.926^{(-)}$	$0.995^{(-)}$	$1.000^{(+)}$	$0.944^{(+)}$	$0.622^{(-)}$
LR	$0.926^{(-)}$	$0.972^{(-)}$	$0.931^{(-)}$	$0.889^{(-)}$	$0.621^{(-)}$
ELM	$0.870^{(-)}$	$0.990^{(-)}$	$0.903^{(-)}$	$0.722^{(-)}$	$0.649^{(+)}$
AB	$0.889^{(-)}$	$0.995^{(-)}$	$0.931^{(-)}$	$0.889^{(-)}$	$0.412^{(-)}$
SVM	$0.944^{(-)}$	$0.959^{(-)}$	$0.903^{(-)}$	$0.944^{(+)}$	$0.629^{(-)}$
NB	$0.907^{(-)}$	$0.940^{(-)}$	$0.903^{(-)}$	$0.944^{(+)}$	$0.595^{(-)}$
C4.5	$0.926^{(-)}$	$0.995^{(-)}$	$0.986^{(-)}$	$1.000^{(+)}$	$0.513^{(-)}$
DL	$0.870^{(-)}$	$0.756^{(-)}$	$0.903^{(-)}$	$0.278^{(-)}$	$0.331^{(-)}$

$^{(+)}$ indicates the algorithms that performed better than PIC GP, $^{(-)}$ those that performed worse to PIC GP

for MCC studied by Zhang et al. [26]. The algorithms are: stochastic gradient boosting decision trees (GBDT), random forests (RF), extreme learning machine (ELM), support vector machine (SVM), C4.5, sparse representation based classification (SRC), KNN, logistic regression (LR), AdaBoost (AB), NB and deep learning (DL). Zhang et al. [26] do not provide the values of the hyperparameters used in their experiments. However, they give information regarding the tuning of the hyperparameters. For GDBT, they tuned all combinations of learning rate with values in [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0] and of the number of nodes in each individual regression tree with values in [1, 2, 3, 4]. For RF, they kept the number of trees fixed to 100 and they tuned the number of features used by the individual classifiers with values from 1 to the number of features of the dateset. For ELM, they tested values of the regularization coefficient from 1 until 100 and of the kernel parameter from 1 until 150, both ranges with step of 1. For SVM, they have used they precomputed kernel, which does not require other hyperparameters tuning. For KNN, they tried different values for the number of clusters. For the other algorithms (C4.5, SRC, LR, AB, NB, DL), they used their default parameters.

As Table 4 shows, PIC GP outperformed all other methods for SHT and PGB datasets. The difference for the second best algorithm was small in both problems (0.02 and 0.034, respectively). Regarding the THY and YST datasets, the performance of PIC GP was also excellent: it outperformed, respectively, 9 and 10 of the 11 algorithms. It is worth to mention that for the THY dataset, it was just slightly outperformed only by GDBT and RF. With the WNE dataset, PIC GP was outformed by 6 of the algorithms. Again, the difference between the best accuracy of PIC GP and the algorithms by which it was outperformed was small (0.056 with GDBT and C4.5 and 0.003 with SCR, RF, SVM and NB).

PIC GP outperformed some other results found in the literature. For example, Tsakonas [30] tested four grammar-guided GP configurations on the THY dataset: with decision trees, with fuzzy rule-based training, with fuzzy petri-nets and with neural networks. The best obtained accuracies were respectively 0.976, 0.941, 0.940 and 0.940 for the test set. Ionita and Ionita [31] also compared methods of machine learning for this dataset. They found that the best runs for Naive Bayes (NB), decision trees, multilayer perceptron and radial basis function network achieved classification accuracies of 0.917, 0.969, 0.951 and 0.960, respectively. Concerning the experiments on the YST dataset, the accuracy achieved by PIC GP is comparable with some results found in the literature, again confirming the robustness of the algorithm. Muñoz et al. [3], for instance, found a median accuracy of 0.562 for this dataset, using a MCC GP wrapper algorithm.

Conclusions

This work presented an empirical study of PIC GP, a robust and accurate GP system for MCC. PIC GP combines the advantages of evolving both individuals that are strongly specialised in the classification of the single target classes, and teams that combine the results of those individuals to obtain the final prediction for MCC problems. Specialists and teams are evolved at the same time, in two independent populations, and the specialists population is further partitioned into subpopulations, one for each different target class. The algorithm is named progressively insular cooperative (PIC) GP because its key feature is the possibility to control the level of cooperation between specialised individuals from a high level of cooperation (where the specialists subpopulations are demes), to a complete separation of the specialists subpopulations (islands). The modifications made in the standard GP for the evolution of specialist individuals were the introduction of new parameters, changes in the selection step and modifications in the individuals' fitness measures. The new parameters that were introduced are the cooperation intensity rate (CIR), the rate of CIR decrease over the algorithm evolution and the generation in which the algorithm starts working with totally separated specialised subpopulations (islands). The selection step was modified to control the interaction between specialists of different classes, using the introduced parameters. The modifications made in the team evolution were in the prediction of the team individual, the team mutation and crossover operators and the teams population replacement. For the latter, instead of replacing the teams population only by their offspring, at each generation a new team population was created with a combination of teams' offspring and new teams formed by evolved specialists. Even though the idea of using mixed teams and specialised subpopulations is not new, PIC GP contains, at least, the following elements of novelty: (1) the combination of the demes and islands phase in different stages of the evolution; (2) the parametrisation to control the cooperation intensity; and (3) the functioning of the teams population, in which, at each generation, the teams evolution is combined with the input of new evolved specialists.

An empirical study was carried out with the Iris dataset to explore the impact of the PIC GP parameters on the dynamics of the system. The Iris dataset is well-known and its characteristics favour the exploration of the role of the algorithm parameters on its evolution because one of its target classes is linearly separable while another is quite hard to discriminate. Therefore, it was possible to understand better how competition among individuals and cooperation among specialists can boost the algorithm

evolution. Importantly, this work clarified a major question from literature: team-based GP can benefit both from the cooperation among specialists of different classes and from a more restricted process, where only an interaction among individuals specialized in the same class is allowed. The contribution of each approach to the algorithm's performance will depend on the performance of each group of specialised individuals. A demes approach helps weaker groups of specialist classifiers, because they may benefit from receiving crucial genetic material from stronger groups. An island approach, on the other hand, allows strong classifiers to evolve to their best potential. The presented results indicate that the combination of both approaches may be the best strategy, at least for the studied test problems. Starting with a demes approach is important to improve the weaker performers. Later, when all groups are strong, the algorithm can change to an islands approach, to allow all the specialised classifiers to reach their best performance.

Additionally, PIC GP was compared to a set of state-of-the-art classification algorithms using the Page Blocks, Shuttle, Thyroid, Wine and Yeast datasets (these five datasets are publicly available in the UCI machine learning repository [27]). The results presented in this paper show that PIC GP outperforms the majority of its competitors.

In the near future, we are planning to begin a vast testing phase of PIC GP on numerous and more complex datasets, implementing more sophisticated genetic operators for the specialists and for the teams and selection methods, for the specialists which are expected to improve the PIC GP performance. The main limitation of PIC GP is its computational cost, but the demes phase can help to reduce it. Thus, we are planning to study how to properly chose the size of the specialists' subpopulations, combined with the *phase change* parameter. This study also will take into account that subpopulations with higher average fitness may need less individuals. Therefore, the size of the subpopulations could be dinamically adjusted over the algorithm evolution to reduce the number of individuals of the stronger subpopulations and at the same time to increase the number of individuals in the weaker subpopulations. This is expected to speed up the learn phase of PIC GP. Besides that, a version of the algorithm for regression problems will be implemented.

Acknowledgements This work was partially supported by FCT, Portugal, through funding of projects BINDER (PTDC/CCI-INF/29168/2017) and AICE (DSAIPA/DS/ 0113/2019).

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Koza JR. Genetic programming: on the programming of computers by means of natural selection. Cambridge: MIT Press; 1992.
2. Boucher D. Mutualism. Integr Comp Biol. 2016;56(2):365–7.
3. Muñoz L, Silva S, Trujillo L. M3GP—multiclass classification with GP. In: Machado P, et. al., editors. Genetic programming, vol 9025. Springer International Publishing; 2015. p. 78–91.
4. Wilson DS. Structured demes and the evolution of group-advantageous traits. Am Nat. 1977;111(977):157–85.
5. Raymer ML, Punch WF, Goodman ED, Kuhn LA. Genetic programming for improved data mining—application to the biochemistry of protein interactions. In: Genetic programming 1996: proceedings of the 1st annual conference. 1996. p. 375–80.
6. Tan X, Bhanu B, Lin Y. Fingerprint classification based on learned features. IEEE Trans Syst Man Cybern Part C (Applications and Reviews). 2005;35(3):287–300.
7. Al-Madi N, Ludwig SA. Improving genetic programming classification for binary and multiclass datasets. In: IEEE symposium on computational intelligence and data mining (CIDM). 2013. p. 166–73.
8. Bi Y, Bing X, Zhang M. Genetic programming with image-related operators and a flexible program structure for feature learning in image classification. IEEE Trans Evol Comput. 2021;25(1):87–101.
9. Zhang M, Smart W. Multiclass object classification using genetic programming. In: Raidl GR, et. al., editors. Applications of evolutionary computing, vol 3005. Berlin: Springer; 2004. p. 369–78.
10. Zhang M, Smart W. Using Gaussian distribution to construct fitness functions in genetic programming for multiclass object classification. Pattern Recognit Lett. 2006;27:1266–74.
11. Kishore JK, Patnaik LM, Mani V, Agrawal VK. Application of genetic Programming for multicategory pattern classification. IEEE Trans Evol Comput. 2000;4(3):242–58.
12. Silva S, Tseng YT, et al. Classification of seafloor habitats using genetic programming. In: Giacobini M, et al., editors. Applications of evolutionary computing, vol. 4974. Berlin: Springer; 2008. p. 315–24.
13. Lin JY, Ke HR, Chien BC, Yang WP. Designing a classifier by a layered multi-population genetic programming approach. Pattern Recognit. 2007;40(8):2211–25.
14. Chien B, Yang J, Lin W. Generating effective classifiers with supervised learning of genetic programming. In: Proceedings of 5th international conference, DaWak 2003. 2003. p. 192–201.
15. Chen Z, Lu S. A genetic programming approach for classification of textures based on wavelet analysis. In: IEEE international symposium on intelligent signal processing. 2007. p. 1–6.
16. Smart W, Zhang M, et al. Using genetic programming for multiclass classification by simultaneously solving component binary classification problems. In: Keijzer M, et al., editors. Genetic programming, vol. 3447. Berlin: Springer; 2005. p. 227–39.
17. Brameier M, Banzhaf W. Evolving teams of predictors with linear genetic programming. Genet Program Evolvable Mach. 2001;2:381–407.
18. Haynes T, Sen S, Schoenefeld D, Wainwright R. Evolving a team. Working Notes for the AAAI symposium on genetic programming. 1995. p. 23–30.
19. Haynes T, Sen S. Crossover operators for evolving a team. In: Proceedings of the second annual conference genetic programming. 1997. p. 1–5.

20. Muni PD, Pal NR, Das J. Evolving teams of predictors with linear genetic programming. IEEE Trans Evol Comput. 2004;8(2):183–90.
21. Lichodzijewski P, Heywood MI. Managing team-based problem solving with symbiotic bid-based genetic programming. In: Proceedings of the 10th annual conference on genetic and evolutionary computation—GECCO '08, vol 363. 2008. p. 363–70.
22. Soule T, Komireddy P. Orthogonal evolution teams: a class of algorithms for evolving teams with inversely correlated errors. Genetic Program Theory Pract IV. 2007;17:79–95.
23. Thomason R, Soule T. Novel ways of improving cooperation and performance in ensemble classifiers. In: Proceedings of the 9th annual conference on genetic and evolutionary computation—GECCO '07. 2007. p. 1708–15.
24. Luke S, Spector L. Evolving teamwork and coordination with genetic programming. In: Genetic Programming 96 (GP96) conference proceedings. 1996. p. 150–56.
25. Soule T. Heterogeneity and specialization in evolving teams. In: Proceedings of the genetic and evolutionary computation conference (GECCO-2000). 2000. p. 778–85.
26. Zhang C, Liu C, Zhang X, Almpanidis G. An up-to-date comparison of state-of-art classification algorithms. Expert Syst Appl. 2017;8(2):128–50.
27. https://archive.ics.uci.edu/ml/
28. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: synthetic minority over-sampling technique. J Artif Intell Res. 2002;16:321–57.
29. Waring J, Lindvall C, Umeton R. Automated machine learning: review of the state-of-the-art and opportunities for healthcare. Artif Intell Med. 2020;104:101822–35.
30. Tsakonas A. A comparison of classification accuracy of four genetic programming-evolved intelligent structures. Inf Sci. 2006;176(6):691–724.
31. Ionita I, Ionita L. Prediction of thyroid disease using data mining techniques. Broad Res Artif Intell Neurosci. 2016;7(3):115–24.